Universidade Federal do Rio Grande do Norte

Centro de Tecnologia

Departamento de Computação e Automação

Engenharia de Computação

Desviando de obstaculos com fuzzy e aprendizagem por reforço

Samuel Cavalcanti

Orientador: Professor: Sérgio Natan Silva

Natal

14 de junho de 2019

Sumário

1 Introdução

Recentemente navegação de robôs moveis tem se tornado um grande objeto de estudo dentro da robótica e no campo da inteligência artificial. O problema da navegação robótica é o robô escolher a decisão correta de conseguir completar a tarefa de encontrar o seu destino sem colidir com obstáculos, de acordo com a informação do ambiente captada pelos sensores (DUAN et al., 2005).

Aprendizagem por reforço é uma técnica de aprendizagem de máquina que aprende com a interação com o ambiente. Duas de suas principais características comparada outras técnicas de aprendizagem de máquina é busca por tentativa e erro e sua recompensa atrasada (SUTTON; BARTO, 2018). A busca é realizada a medida que o agente interage com o ambiente e após uma ação ou um conjunto delas o agente recebe uma recompensa. A aprendizagem por reforço é recomendada quando não se é possível obter bons exemplos para todas as situações. Então nesses casos o agente tem que aprender pela própria experiência (ZHANG,).

Um dos grandes desafios do aprendizado por reforço é o gerenciamento do quanto o agente deve tentar ações ainda não conhecidas e quando ele vai executar ações que maximiza a sua recompensa. Para resolver esse desafio além do agente contar com o próprio conhecimento, foi utilizado aprendizagem por demonstração.Com a aprendizagem por demonstração uma sequência de estado-ação é aprendido a partir de um exemplo ensinado por um professor. Um exemplo é um comportamento, uma sequência de estado-ação que foi gravado durante a demonstração do professor (ARGALL et al., 2009).

Outro desafio da aprendizagem por reforço é a definição da função recompensa, uma vez que os algoritmos visam maximizar essa função o que não necessariamente ira gerar o comportamento desejado. Abordagem utilizada para gerar a função recompensa foi a logica nebulosa. A teoria dos conjuntos nebulosos, quando utilizada em um contexto lógico, como o de sistemas baseados em conhecimento, é conhecida como lógica nebulosa, lógica difusa ou lógica “fuzzy”(SANDRI; CORREA, 1999). Um

controlador logico fuzzy é um sistema especialista baseado em regras de se-então, que busca representar a linguagem natural humana (DUAN et al., 2005).

Esse relato está contextualizado no problema de navegação de robôs moveis utilizado técnicas de aprendizagem por reforço para encontrar a melhor politica ou estado-ação para desviar de obstáculos. Onde a função de reforço é dada por um controlador logico fuzzy e para reduzir o tempo de aprendizagem, foi feito uma única demonstração para o robô de uma possível política que ele poderia seguir para desviar de obstáculos. A utilização de aprendizagem por reforço e controlador logico fuzzy para resolver esse problema não é novidade, um sistema utilizando essas duas técnicas já foi proposto por (DUAN et al., 2005), onde ele utilizou uma Q(γ)-learning e um controlador fuzzy, para resolver esse problema. A utilização de aprendizagem por demonstração também não é novidade , onde (ARGALL et al., 2009) utilizou aprendizagem pro demonstração para ensinar uma política a um robô.

2 Abordagem Teórica

Nesse relato foi utilizado um robô simulado, implementado o deep Q-learning com algumas alterações necessárias para o problema com a biblioteca keras e criado a função de recompensa com a biblioteca skfuzzy.

2.1 aprendizado por reforço

Aprendizado por reforço é o aprendizado de como mapear situações para ações de modo que maximize uma função recompensa. O aprendiz não sabe a priore quais ações deve realizar, ele deve descobrir quais ações maximizam a função recompensa a partir da tentativa e erro. O fato mais interessante é que as ações não só afetam a recompensa imediata como também afetam a recompensa das próximas situações. Aprendizado por reforço é um conjunto de soluções que visam resolver problemas oriundos da teoria de sistemas dinâmicos e otimização de controle de processos markovianos (SUTTON; BARTO, 2018). dentre esse conjunto de soluções o algoritmo deep Q-learning with experience replay foi escolhido 1. O deep Q network (DQN) é uma rede neural de múltiplas camadas que recebe um estado s e lhe dá como saída um vetor de ações $Q(s, \cdot; \theta)$, onde θ são os parâmetros da rede. Para um espaço $n-$dimensional e o espaço m de ações. A rede neural é uma função do $\mathbb{R}^n$ para $\mathbb{R}^m$. Dois importantes passos desse algoritmo proposto por (MNIH et al., 2015) foi o uso do rede alvo e uma adaptação chamada de memória ou experience replay. Rede alvo é usar a própria saída da rede como parte do vetor de amostras com a diferença que o nodo vencedor será atualizado seguindo a equação 2.1.

$$y_j^{DQN} = r_j + \gamma \text{argmax}_{\acute{a}} \hat{Q}(\theta_{j+1}, \acute{a}; \bar{\theta}) \tag{2.1}$$

Já a memória é uma estrutura onde é armazenado por um determinado período de tempo as transições observadas pelo agente. A partir desse banco de memórias é

que a rede neural será atualizada. Tanto a rede alvo quando o a memória melhoram a performance do algoritmo (HASSELT; GUEZ; SILVER, 2016) , (MNIH et al., 2015)

O algoritmo de aprendizado por reforço usado foi o deep Q-learning com o uso da memória (MNIH et al., 2015). O seu pseudo código pode ser encontrado aqui 1

Algorithm 1 deep Q-learning with experience replay

inicialize a memoria D com capacidade N
inicialize a função ação-valor Q com pesos randômicos θ
inicialize a função de valor de destino $\hat{Q}$ com pesos $\bar{\theta} = 0$
for episódio =1, M **do**
    s_t = valores dos sensores
    Com probabilidade ϵ selecione uma ação randômica a_t
    ou selecione $a_t = \text{argmax}_a Q((s_t), a; \theta)$
    Execute a ação a_t no simulador e observe a recompensa r_t e o estado s_{t+1}
    armazene a transição (s_j,a_j,r_j,s_{j+1}) em D
    recupere um mine pacote de amostras de transições (s_t,a_t,r_t,s_{t+1}) de D
    if o episódio acabar no passo $j + 1$ **then**
        $y_j = r_j$
    else
        $y_j = r_j + \gamma \text{argmax}_{\acute{a}} \hat{Q}(\theta_{j+1}, \acute{a}; \bar{\theta})$
    Use o gradiente descendente em $(y_j - Q(s_j, a_j; \theta))^2$ no parâmetros θ da rede neural

2.2 Controlador logico nebuloso

As técnicas de controle nebuloso originaram-se comas pesquisas e projetos de (MAMDANI, 1976) ,(MAMDANI; PROCYK; BAAKLINI, 1976) e ganharam espaço como área de estudo em diversas instituições de ensino, pesquisa e desenvolvimento do mundo, sendo até hoje uma importante aplicação da teoria dos conjuntos nebulosos.Ao contrário dos controladores convencionais em que o algoritmo de controle é descrito analiticamente por equações algébricas ou diferenciais, através de um modelo matemático, em controle nebuloso utilizam-se de regras lógicas no algoritmo de controle, com a intenção de descrever numa rotina a experiência humana, intuição e heurística para controlar um processo (SANDRI; CORREA, 1999).

2.3 aprendizagem por demonstração

O princípio de aprendizagem por demonstração (learning from demonstration-LfD) se baseia em ensinar novas tarefas a robôs sem a necessidade de programação. Tomando em conta um cenário clássico de programação, se faz necessário primeiramente programar todas as tarefas que se deseja que o robô realize, sendo necessário cobrir todas as possibilidades e adversidades que possam ocorrer nesse cenário. Esse processo porém, envolve muitas etapas e testes, e caso erros ou novas circunstâncias ocorram depois da implementação no robô, muitas e em alguns casos todas as etapas do processo precisam ser refeita (EKVALL; KRAGIC, 2008).Técnicas e métodos de aprendizagem por demonstração permitem ao usuário final comandar e especificar tarefas a serem realizadas pelo robô, sem nenhuma necessidade de programação, apenas demonstrando fisicamente como realizá-las. Dessa forma, quando algum erro ou adversidade ocorrer, será necessário apenas fornecer mais demonstrações para o robô, evitando assim a necessidade de reprogramação do mesmo. (MOTTA, 2016).

3 Descrição da Proposta

Para avaliar o sistema que desvia de obstáculos, foi feito uma simulação que a partir dela foi observado os valores do sensores para a formação das regras do controlador fuzzy. Por ultimo criado uma estratégia de treinamento envolvendo aprendizagem por demonstração. Para facilitar o leitor esse capítulo foi dividido em 3 partes: a primeira irá detalhar a simulação, a segunda será sobre o controlador fuzzy e a ultima contará a estratégia de treinamento.

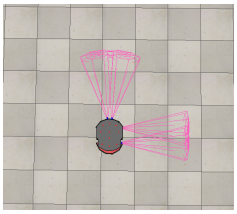
3.1 Simulação

O simulador utilizado foi o V-rep. Um simulador de robótica que provê uma quantidade razoável de robôs já prontos e boa documentação. A partir desse simulador foi selecionado o robô Pioneer 1a a qual possui dezesseis sensores ultrassônicos e tração diferencial 1b. Como o número de regras cresce exponencialmente com o número de entradas do controlador fuzzy. Foi então decidido limitar o movimentação do robô para frente e para a esquerda e retirar doze sensores para simplificar o problema. Nesse relato foi utilizado duas cenas, a primeira 1c é a mais simples utilizada para treinamento do algoritmo, é um único cômodo fechado a quatro paredes, a segunda 1d , simula um apartamento com três cômodos e um corredor, essa cena foi utilizada para avaliar o sistema de controle.

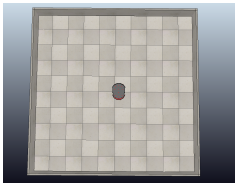
3.2 Controlador fuzzy

O controlador fuzzy funcionava como uma função que recebe a média dos sensores da frente e a média dos sensores da direita como entrada, e retornava um valor entre menos um e um. Para chegar nessa função foi necessário duas logicas fuzzy, a primeira mensurava o quão distante estava uma das media dos sensores até o obstáculo ,a segunda mensurava a recompensa a partir da primeira. A primeira logica transformava a distancia das médias dos sensores em um grau de pertinência

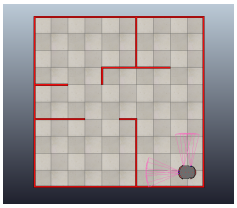
(a) Robô Pioneer



(b) sensores do Pioneer



(c) cena de treinamento



(d) cena de validação

de três classes: perto, bom , longe. A classe perto é um trapézio definido nos pontos: $(-1, 2, 0, 0.2, 0.3)$, a classe bom é um triangulo com os pontos: $(0.2, 0.4, 0.7)$e a classe longe é outro trapézio com os pontos $(0.6, 0.8, 1, 1.2)$. Ambas as médias dos sensores passam por essa mesma lógica que pode ser melhor compreendida no gráfico 2. Para a segunda lógica foi necessário criar cinco regras que mapeasse o grau de pertinência de cada classe para trés tipos recompensa: ruim, neutra e boa. As cinco regras foram:

- **Se** a média dos sensores da frente **OU** a média dos sensores da direita for perto **então** a recompensa é ruim
- **Se** a média dos sensores da frente for boa **&** a média dos sensores da direita for boa **então** a recompensa é boa
- **Se** a média dos sensores da frente for boa **&** a média dos sensores da direita for longe **então** a recompensa é boa
- **Se** a média dos sensores da frente for longe **&** a média dos sensores da direita for boa **então** a recompensa é boa
- **Se** a média dos sensores da frente for longe **&** a média dos sensores da direita for longe **então** a recompensa é neutra

Depois das regras foi definido o formato da função de cada classe da logica da recompensa. A classe ruim é um triangulo com os pontos: $(-1, -1, 0)$, classe neutra é um triangulo com os pontos: $(-0.5, 0, 0.5)$ e a classe boa é um triangulo com os pontos: $(0, 1, 1.2)$ essa funções podem ser melhor compreendida no gráfico 3, 4, 5. O processo de defuzzificação utilizado é o centroide.

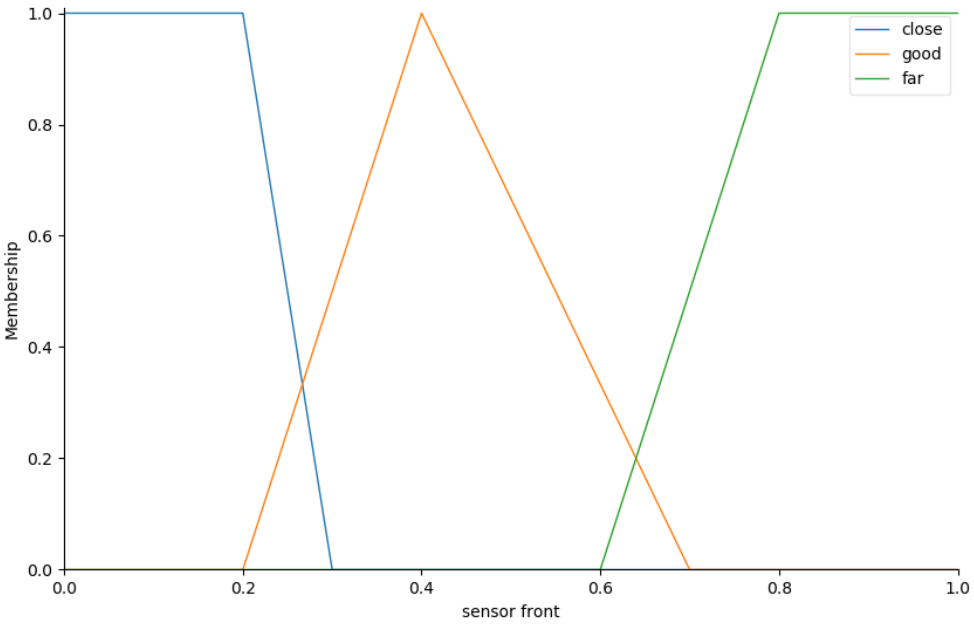


Figura 2: logica fuzzy da distancia de um sensor até o obstáculo

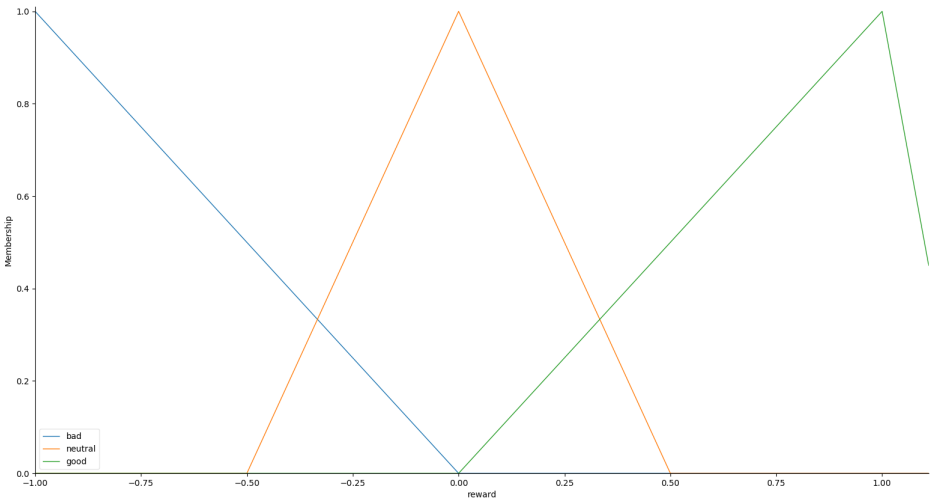


Figura 3: logica fuzzy da recompensa

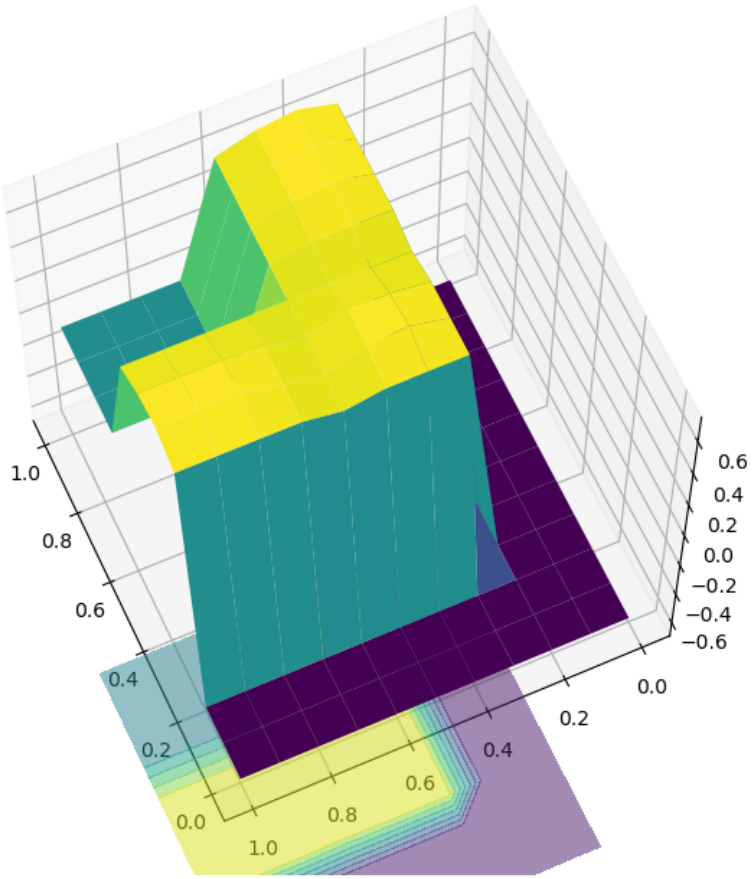


Figura 4: superfície de controle

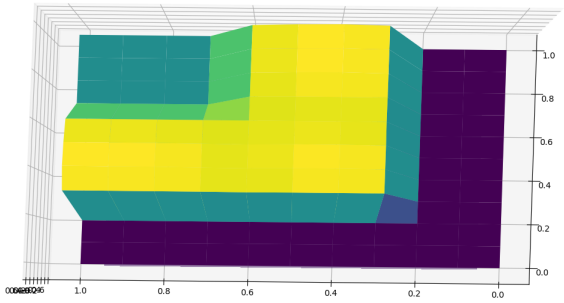


Figura 5: superfície de controle vista de cima

3.3 estratégia de treinamento

O aprendizado por reforço é uma busca por tentativa e erro (SUTTON; BARTO, 2018), ou seja dependendo da complexidade do problema o custo computacional fica bastante elevado. Para mitigar um pouco desse custo foi utilizado uma técnica chamada aprendizagem por demonstração, nela um professor demonstra uma política que é gravada e apresentada ao agente. O objetivo dessa estratégia é dar um conhecimento base ao robô que seria refinado a medida que ele interagisse com o ambiente. Então a estratégia de treinamento ficou:

- robô aprende a única demonstração feita
- robô fica em contado com o ambiente simples 1c até melhorar a política demonstrada
- robô fica em contado com o ambiente de validação 1d para mensurar a política aprendida e caso fique tempo o suficiente, refina ainda mais a política aprendida na cena simples 1c.

A cada duzentos movimentos do robô a simulação era reiniciada.

4 Desenvolvimento e Resultados

Para avaliar o sistema foi feito uma avaliação visual do robô e gráficos da recompensa por movimento do robô. A avaliação visual é verificado se algum momento o robô colide com a parede e pode-se encontrar os vídeos dessa avaliação abaixo:

- **Primeiro vídeo** link primeiro vídeo
- **Segundo vídeo** link segundo vídeo
- **Terceiro vídeo** link terceiro vídeo

nó primeiro vídeo é mostrado desempenho do robô com apenas o conhecimento base, ou seja treinado com a única demonstração. podemos observar que o robô aprendeu que tem que desviar da parede, ele vai chegando perto e bate na parede e sua função de recompensa por movimento reflete esse comportamento 6. O segundo vídeo é mostrado o desempenho do robô após algumas interações no primeiro cenário 1c. Novamente o gráfico da função recompensa reflete o seu comportamento, assim como o terceiro vídeo e o terceiro gráfico 8, no entanto vale lembrar que o segundo cenário possui mais obstáculos logo a recompensa por movimento oscila mais.

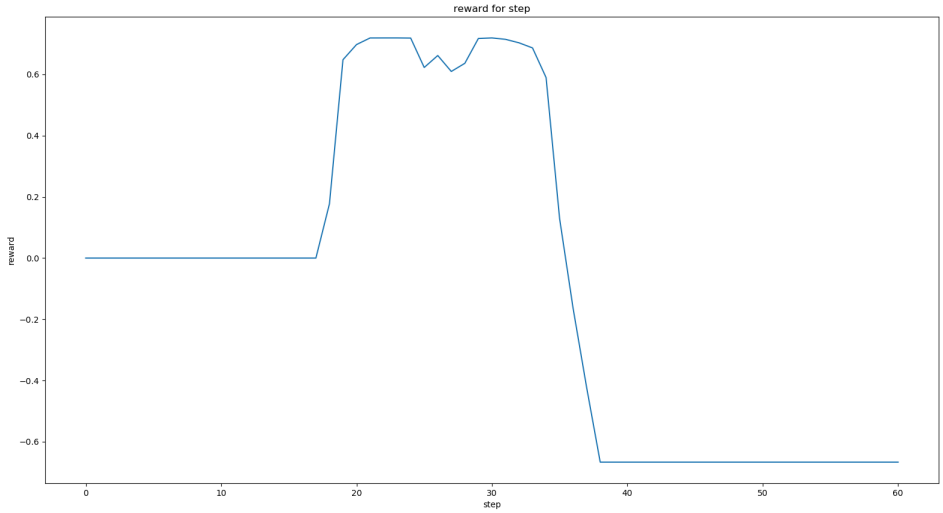


Figura 6: gráfico da recompensa do conhecimento base no primeiro cenário 1c

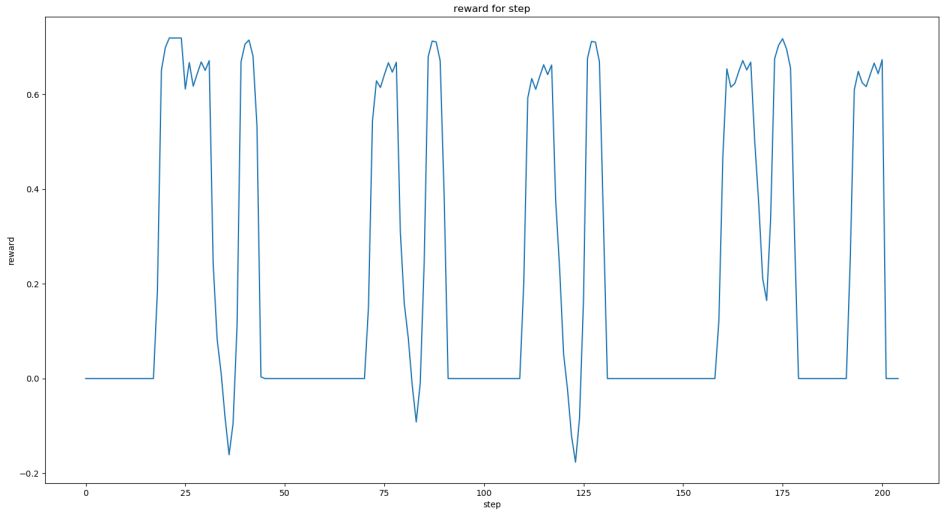


Figura 7: gráfico da melhor recompensa no primeiro cenário 1c

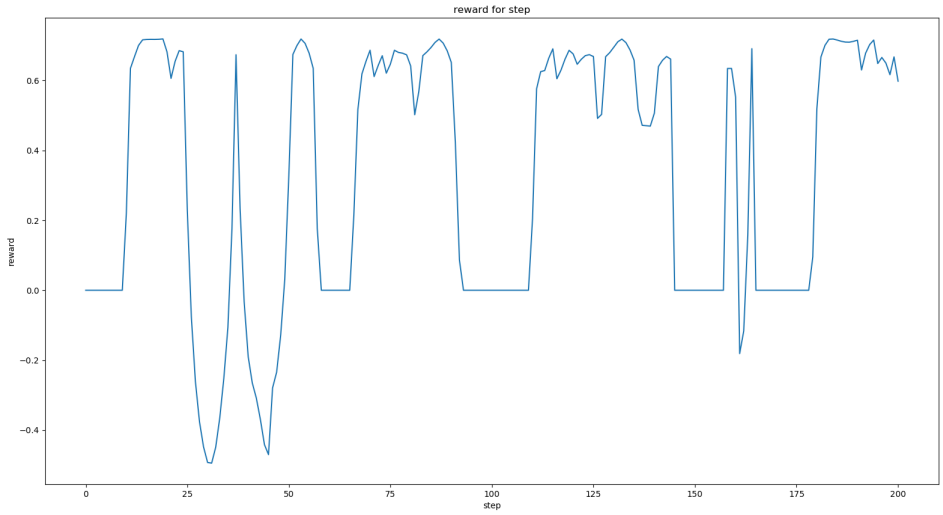


Figura 8: gráfico da recompensa no segundo cenário 1d

5 Conclusão

A rede convergiu e foi capaz de aprender a desviar de obstáculos. Mostrando que a função de recompensa foi capaz de gerar uma politica ou comportamento satisfatório para o problema. No entanto a função de recompensa foi projetada de modo que recompensa-se o robô a ficar próximo do obstáculo e o robô tende a fugir completamente do obstáculo. Esse comportamento é possivelmente um mínimo local, onde uma das possíveis causas desse fato tenha sido a simplificação do problema, talvez permitindo o agente a se mover para direita e com isso aumentar o número de sensores é possível que o robô consiga sair desse mínimo e consiga melhores resultados

Referências

ARGALL, B. D. et al. A survey of robot learning from demonstration. *Robotics and autonomous systems*, Elsevier, v. 57, n. 5, p. 469–483, 2009. 2, 3

DUAN, Y. et al. Fuzzy reinforcement learning and its application in robot navigation. In: IEEE. *2005 International Conference on Machine Learning and Cybernetics*. [S.l.], 2005. v. 2, p. 899–904. 2, 3

EKVALL, S.; KRAGIC, D. Robot learning from demonstration: a task-level planning approach. *International Journal of Advanced Robotic Systems*, SAGE Publications Sage UK: London, England, v. 5, n. 3, p. 33, 2008. 6

HASSELT, H. V.; GUEZ, A.; SILVER, D. Deep reinforcement learning with double q-learning. In: *Thirtieth AAAI Conference on Artificial Intelligence*. [S.l.: s.n.], 2016. 5

MAMDANI, E.; PROCYK, T.; BAAKLINI, N. Application of fuzzy logic to controller design based on linguistic protocol. *Discrete Systems and Fuzzy Reasoning*, London, UK: Queen Mary College, University of London, p. 125–149, 1976. 5

MAMDANI, E. H. Advances in the linguistic synthesis of fuzzy controllers. *International Journal of Man-Machine Studies*, Elsevier, v. 8, n. 6, p. 669–678, 1976. 5

MNIH, V. et al. Human-level control through deep reinforcement learning. *Nature*, Nature Publishing Group, v. 518, n. 7540, p. 529, 2015. 4, 5

MOTTA, B. d. C. Aprendizagem por demonstração baseada em redes neurais artificiais aplicada à robótica móvel. 2016. 6

SANDRI, S.; CORREA, C. Lógica nebulosa. *Instituto Tecnológico da Aeronáutica–ITA, V Escola de Redes Neurais, pp. C073-c090, São José dos Campos*, 1999. 2, 5

SUTTON, R. S.; BARTO, A. G. *Reinforcement learning: An introduction*. [S.l.]: MIT press, 2018. 2, 4, 12

ZHANG, C. Reinforcement learning for robot obstacle avoidance and wall following. 2

Procure citar todas as referências utilizadas no projeto.