Relatório de I.A.: Sistemas Fuzzy (Trabalho 4)

Cauê Baasch de Souza
João Paulo Taylor Ienczak Zanette

9 de Novembro de 2018

1 O "Fuzzy Truck"

No problema "Fuzzy Truck", é necessário definir parâmetros, saídas e regras de um sistema Fuzzy para fazer um caminhão, andando de ré, estacionar em uma doca considerando um espaço 2D sem obstáculos. Como restrições do problema, o caminhão possui velocidade constante (com exceção de que ele para quando está suficientemente próximo do espaço que delimita a vaga) e suas únicas ações possíveis são: girar o volante para a esquerda (indicado pelo valor -1), manter a direção atual do caminhão (valor 0) e girar o volante para a direita (valor 1).

O espaço por onde o caminhão anda é delimitado pelas coordenadas X e Y variando de 0 a 1 com um pequeno limiar extra (com exceção de valores de $Y > 1$). Ao se passar dessa delimitação, a simulação é encerrada e uma pontuação é dada considerando o número de passos dados, o ângulo final do caminhão e a distância até a doca, que está centralizada no ponto $(0.5, 1)$.

Para simulação, é utilizado um programa gráfico simples (Figura 1), escrito em Java, que simula o sistema em duas instâncias. O objetivo é que cada instância represente um competidor, que irá rodar seu sistema de inferência Fuzzy e enviar comandos ao programa. A comunicação é feita via Sockets em um host e portas pré-definidos (`localhost` para o Host, e 4321 e 4322 para as portas de cada jogador). Todo o código da solução realizada neste trabalho está disponível em [1].

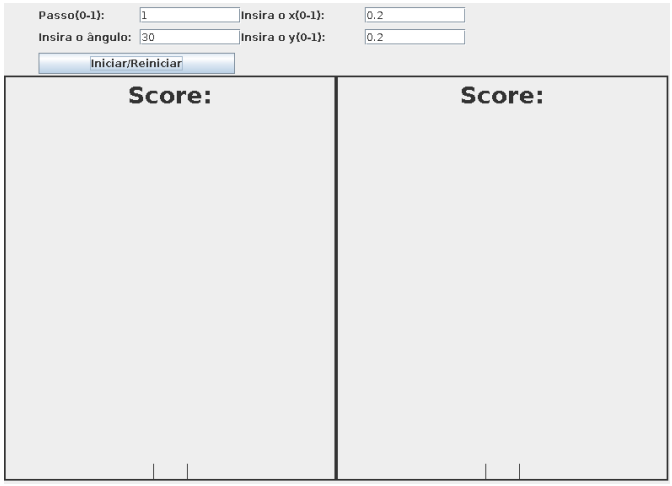

Figura 1: Tela inicial do simulador para o Fuzzy Truck.

Entrada	Símbolo	Tipo	Intervalo esperado
X	`x`	Real	$[0, 1]$
Y	`y`	Real	$[0, 1]$
Ângulo	`dir`	Real	$[0, 360]$

Tabela 1: Entradas utilizadas para o sistema em questão. Cada entrada pode receber um valor fora de seu intervalo esperado, porém o comportamento resultante não pode ser previsto nesse caso.

2 Resolvendo o problema do "Fuzzy Truck"

2.1 Entradas utilizadas e Conjuntos Fuzzy

Foram utilizadas três entradas simples, descritas na Tabela 1, em que X e Y simbolizam as coordenadas do caminhão.

Os conjuntos fuzzy (visualizáveis na Figura 2) foram definidos pensando em:

- O quão longe o caminhão está horizontalmente, separando em: "muito para a esquerda", "muito para a direita" e "centralizado";
- O quão longe o caminhão está verticalmente, separando em: "muito longe" e "próximo";
- A direção da traseira do caminhão, separada em: norte, sul, leste e oeste.

2.2 Regras utilizadas e Defuzzificação

O sistema modelado dá como saída uma variável chamada de "Ação" (`action` na Figura 2), que pode ser:

- Virar o volante para a direita (valores de -1 a 0);
- Virar o volante para a esquerda (valores de 0 a 1);

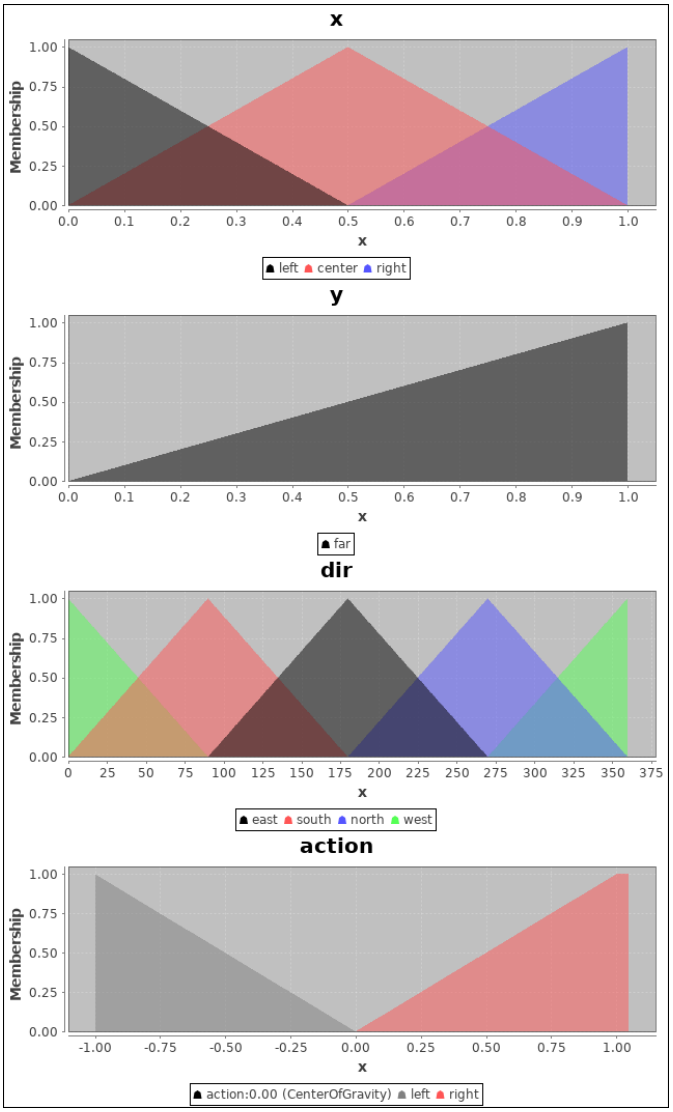

Figura 2: Conjuntos Fuzzy para variáveis de entrada e saída do "Fuzzy Truck".

Por consequência da inferência do sistema Fuzzy, "0" corresponde a "Não virar". Anteriormente havia sido definido um intervalo específico para essa ação, centralizado em "0", porém ele se mostrou desnecessário e ineficiente, porque uma decisão que deveria ter um alto grau de pertinência para "Não virar" já terá, de acordo com as regras, um baixo grau de pertencimento aos conjuntos respectivos às outras ações. Consequentemente, tal decisão já terá um valor próximo do zero esperado.

Por simplicidade, as regras definidas foram:

- Muito para a esquerda $\wedge$ Norte $\rightarrow$ Virar para a direita
- Muito para a esquerda $\wedge$ Muito longe $\wedge$ Leste $\rightarrow$ Virar para a direita
- Muito para a esquerda $\wedge$ Muito longe $\wedge$ Sul $\rightarrow$ Virar para a esquerda
- Centralizado $\wedge$ Oeste $\rightarrow$ Virar para a esquerda
- Centralizado $\wedge$ Norte $\rightarrow$ Virar para a esquerda
- Centralizado $\wedge$ Leste $\rightarrow$ Virar para a direita
- Muito para a direita $\wedge$ Norte $\rightarrow$ Virar para a esquerda
- Muito para a direita $\wedge$ Muito longe $\wedge$ Oeste $\rightarrow$ Virar para a esquerda
- Muito para a direita $\wedge$ Muito longe $\wedge$ Sul $\rightarrow$ Virar para a direita

A ação é defuzzificada dos conjuntos da saída "Ação" para um valor no intervalo $[-1, 1]$ utilizando o método do **Centro de Gravidade**.

3 Resultados obtidos

Diversos testes com diferentes valores de entrada foram feitos. Serão expostos aqui apenas três deles, escolhidos da forma: utilizando as configurações iniciais do programa de simulação (Figura 3), utilizando um caso trivial (Figura 4) e utilizando um *corner-case* (Figura 5). A Tabela 2 descreve os parâmetros utilizados em cada cenário.

Teste	Passo	X_0	Y_0	Ângulo	Pontuação
Padrão	1	0.2	0.2	30	9904.47
Trivial	1	0.5	0.2	90	9996.00
Corner-case	1	0.8	0.8	160	9821.47

Tabela 2: Parâmetros e pontuação para cada um dos casos de teste expostos.

Nos dois primeiros casos, o sistema conseguiu responder bem aos estímulos e levar o caminhão ao destino correto. No último, porém, a reação foi levar o caminhão apenas para baixo. Houveram tentativas, conforme mencionado mais à frente nas dificuldades encontradas para a modelagem, de resolver casos semelhantes com regras mais específicas (como, por exemplo, definir que quando o caminhão está muito próximo de $Y = 1$ deve contornar pelo outro lado), porém isso degradava outros resultados e nem sempre fazia o caminhão chegar ao seu destino.

4 Dificuldades encontradas

4.1 Ferramental

Devido à vontade de aventura, os alunos decidiram utilizar o gerador de código C++ a partir de FCL (linguagem

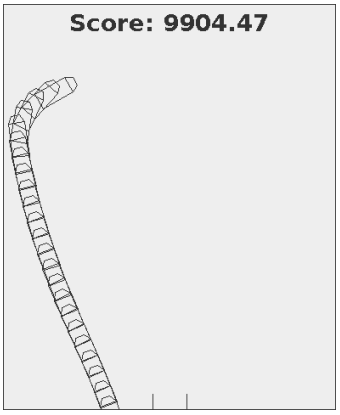

Figura 3: Trajetória do caminhão no caso padrão, utilizando as configurações iniciais do programa de simulação.

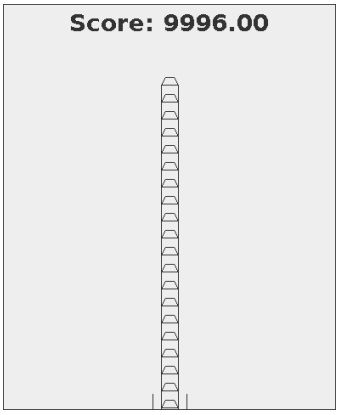

Figura 4: Trajetória do caminhão no caso trivial, em que o caminhão deve ir apenas reto.

de descrição de sistemas Fuzzy) pelo jFuzzyLogic (biblioteca e programa para funcionalidades relacionadas a lógica Fuzzy, incluindo geração de gráficos). Porém, um pouco fora do esperado, isso acarretour na necessidade de passos extras como: melhoria do código gerado, separação das declarações em header, e implementação simples de uma API Socket em C++. Porém, tendo esses passos prontos, os ajustes e criação das regras e parâmetros do sistema para o FuzzyTruck foram mais triviais. Como FCL obriga a enumeração de regras, para melhorar a produtividade criou-se um pequeno script em Python que gerava o arquivo FCL a partir de regras definidas em uma lista de `Rule`s (uma tupla com os valores esperados e o valor de saída para a regra).

4.2 Modelagem da solução

Da parte da definição das regras, diferentes abordagens foram tomadas tentando ser o mais restritivo possível: utilização de pontos colaterais (totalizando 8 direções) e definindo regras para cada caso (muito longe, no centro ou perto em Y, combinando com se está à esquerda,

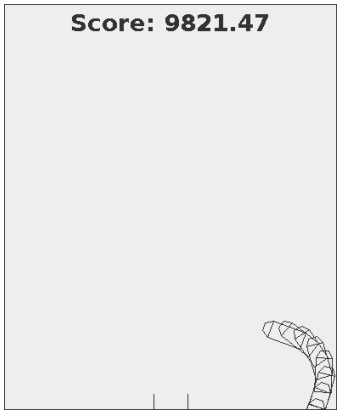

Figura 5: Trajetória do caminhão no *corner-case*. É evidente que o sistema não resolve esse caso.

centro ou direita em X, combinando com cada uma das direções possíveis). Para cada mudança nas regras, o caminhão se comportava com desempenho muito bom (às vezes melhor do que a versão final deste trabalho) para um mesmo conjunto de valores de entrada, porém em outro o desempenho era péssimo. Depois de vários testes e regras específicas criadas, decidiu-se remodelar as entradas bem como seus conjuntos fuzzy e então redefinir regras simples e diretas. Com essa última mudança, os resultados se tornaram muito mais plausíveis para uma boa parte dos casos não extremos (como iniciar-se muito próximo da doca em Y, porém muito longe em X). Tentativas de adicionar novas regras voltaram ao mesmo comportamento de antes: melhoravam um pouco em alguns pontos, pioravam muito mais em outros.

Sendo assim, é possível (e mais aconselhável) confiar no sistema Fuzzy para escolher as ações em resultados intermediários, se preocupando apenas com pontos estratégicos dos conjuntos Fuzzy em vez de deixá-los densos e extremamente detalhistas.

Referências

[1] Repositório oficial desta solução para o Fuzzy Truck. `https://github.com/jptiz/fuzzytruck`.