**Implementação de sistema especialista Fuzzy para determinar condições para SURF em determinado local**

**Gabriel Civinski; William Lohn**

1. **Introdução**

Neste relatório será detalhada a implementação de um Sistema Especialista Fuzzy utilizando a ferramenta FuzzyClips. A finalidade do sistema é determinar as condições para SURF em um local específico, considerando como variáveis auxiliares a altura média das ondas e a velocidade do vento. As variáveis linguísticas de entrada e saída são mostradas na Tabela 1:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **VELOCIDADE VENTO** | | | |
| **ALTURA ONDA** | **Fraco** | **Médio** | **Forte** | **Muito forte** |
| **Baixa** | ruim | ruim | ruim | ruim |
| **Média** | boa | razoavel | razoavel | ruim |
| **Alta** | muito\_boa | boa | razoavel | ruim |
| **Muito alta** | muito\_boa | razoavel | ruim | ruim |

Tabela 1: variáveis linguísticas para determinar as condições de surf

Na primeira coluna estão relacionados os valores referentes à altura média das ondas, enquanto na primeira linha, estão os valores para determinar a velocidade/força do vento. Cruzando os valores, temos definida a condição para surf.

1. **Implementação e testes**

Para cada variável linguística foi definido um *template*, conforme a seguir. Para a variável linguística “altura\_onda” teremos uma função pré-definida (z) e três trapezoidal.

(deftemplate altura\_onda

0 300 altura\_onda

((baixa (z 0 50))

(media (51 0)(51 1)(130 1)(130 0))

(alta (131 0)(131 1)(200 1)(200 0))

(muito\_alta (201 0)(300 1))

)

)

Figura 1. Template da variável “altura\_onda”

Na figura 2 temos a representação gráfica dos possíveis valores para a altura da onda, definidos em seu template (valores em cm).

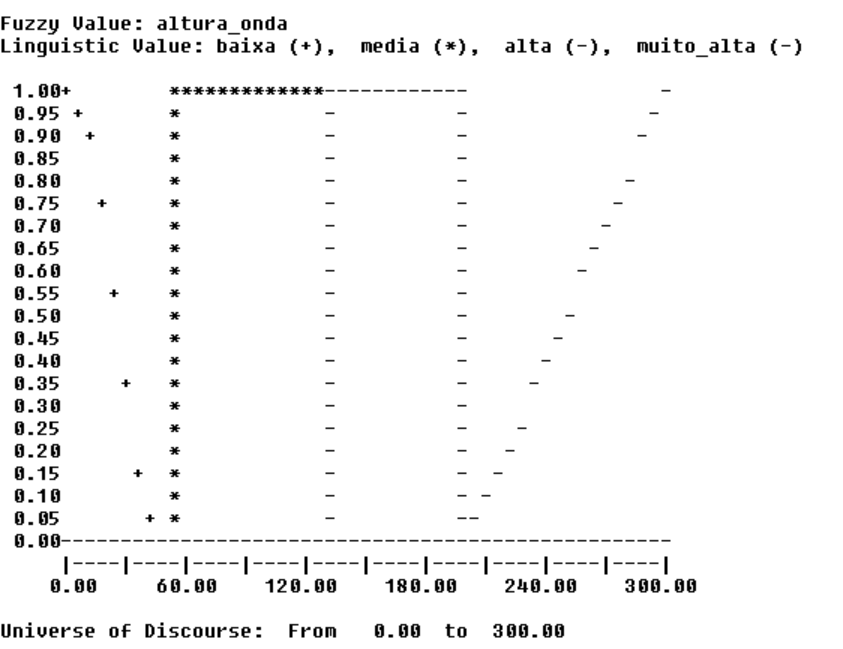


Figura 2. Gráfico para varável “altura\_onda”

Na figura 3 podemos observar o template definido para a força do vento.

(deftemplate vento

0 50 vento

((fraco (z 0 10))

(medio (10 0)(10 1)(18 1)(18 0))

(forte (19 0)(29 1)(30 1)(30 0))

(muito\_forte (30 0)(50 1))

)

)

Figura 3. Template para variável vento

E a seguir, na figura 4, podemos observar a representação gráfica deste template (valores em km/h).

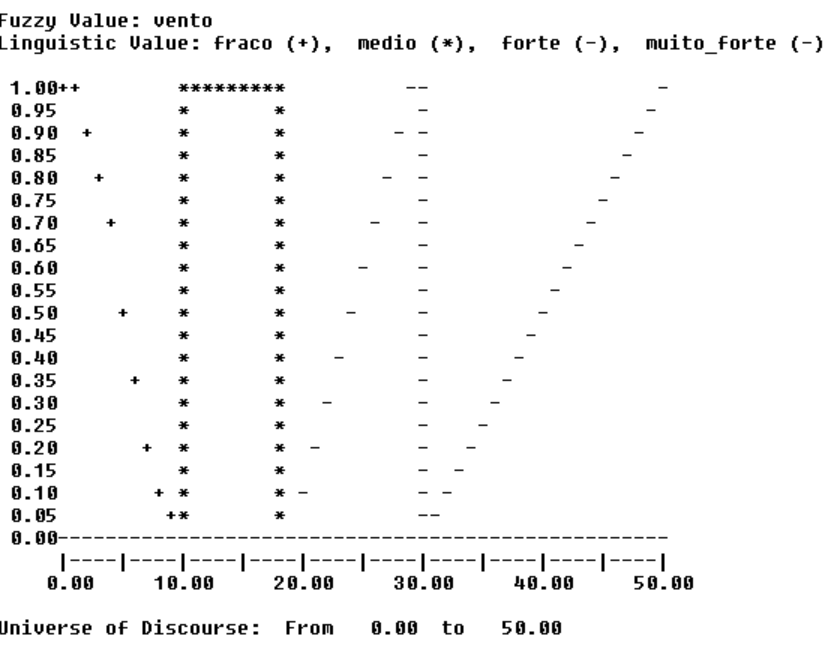


Figura 4. Gráfico para variável “vento”

Na figura 5 podemos observar o template definido para as condições de surf

(deftemplate condicoes

0 4

((ruim (0 0)(0.5 1)(1 0))

(razoavel (1.1 0)(1.5 1)(2 0))

(boa (2.1 0)(2.5 1)(3 0))

(muito\_boa (3.1 0)(3.5 1)(4 0))

)

)

Figura 5. Template para as condições

Abaixo, na figura 6, podemos observar a representação gráfica do mesmo.

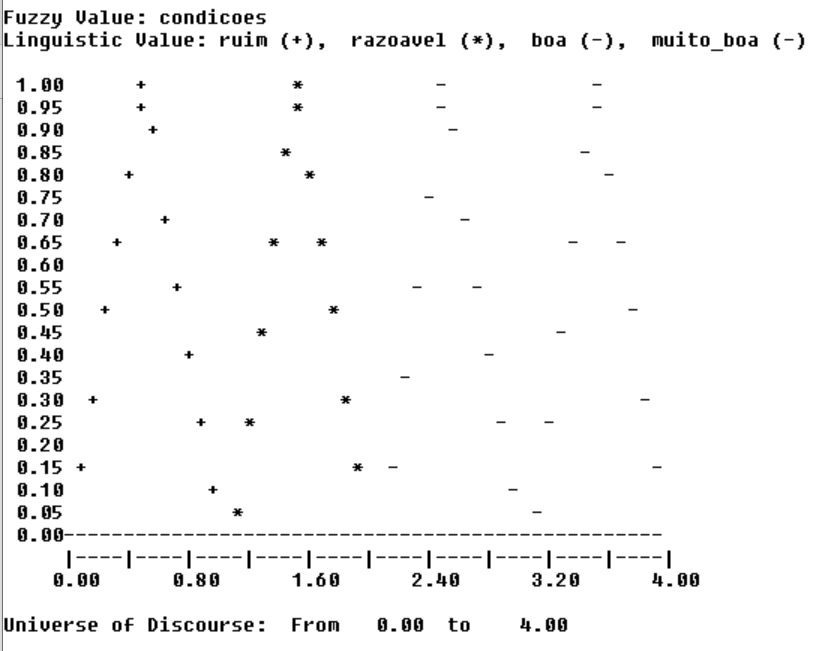


Figura 6. Representação gráfica do template “condicoes”

As regras que vão definir a condição para surf foram codificadas em quatro defrule’s distintos, um para cada variável linguística e condição. A utilização da declaração “*salience*” foi a solução adotada para garantir que essas regras fossem executadas antes da regra de defuzzificação.

(defrule ruim

        (declare (salience 10))

        (or(and (altura\_onda baixa)(vento fraco))

         (and (altura\_onda baixa)(vento medio))

(and (altura\_onda baixa)(vento forte))

(and (altura\_onda baixa)(vento muito\_forte))

(and (altura\_onda media)(vento muito\_forte))

         (and (altura\_onda alta)(vento muito\_forte))

(and (altura\_onda muito\_alta)(vento muito\_forte))

(and (altura\_onda muito\_alta)(vento forte))

        )

        =>

        (assert (condicoes ruim))

    )

(defrule razoavel

        (declare (salience 10))

        (or(and (altura\_onda media)(vento forte))

         (and (altura\_onda media)(vento medio))

(and (altura\_onda muito\_alta)(vento medio))

(and (altura\_onda alta)(vento forte))

        )

        =>

        (assert (condicoes razoavel))

    )

(defrule boa

        (declare (salience 10))

         (or(and (altura\_onda alta)(vento medio))

(and (altura\_onda media)(vento fraco))

        )

        =>

        (assert (condicoes boa))

    )

(defrule muito\_boa

        (declare (salience 10))

         (or(and (altura\_onda muito\_alta)(vento fraco))

(and (altura\_onda alta)(vento fraco))

        )

        =>

        (assert (condicoes muito\_boa))

    )

Figura 7. Código fonte para definição das condições

Para a *defuzzificação* foi definida uma variável global (g\_resultado) e foi criada uma outra regra (defrule defuzifica) com *salience 0*, para que dessa forma, ela seja executada após todas as demais regras declaradas com *salience 10*, para assim, podermos ter o resultado das condições.

(defglobal *?*\*g\_resultado\* = 0)

Figura 8. Definição da varável g\_resultado

(defrule defuzifica

(declare (salience 0))

*?v\_tmp* <- (condicoes *?*)

=>

(bind *?*\*g\_resultado\*(moment-defuzzify *?v\_tmp*))

(plot-fuzzy-value t "\*" nil nil *?v\_tmp*)

(retract *?v\_tmp*)

(printout t "Condição para surf: ")

(printout t *?*\*g\_resultado\* crlf)

(printout t ">>> TERMINOU <<<" crlf)

)

Figura 9. Definição da rule “defuzifica”

Para testes, definimos fatos fixos com a função “deffacts”

(deffacts condicoes

(altura\_onda baixa)

(vento fraco)

)

Figura 10. Fatos definidos para testes

A seguir, podemos ver o resultado gerado pela aplicação.

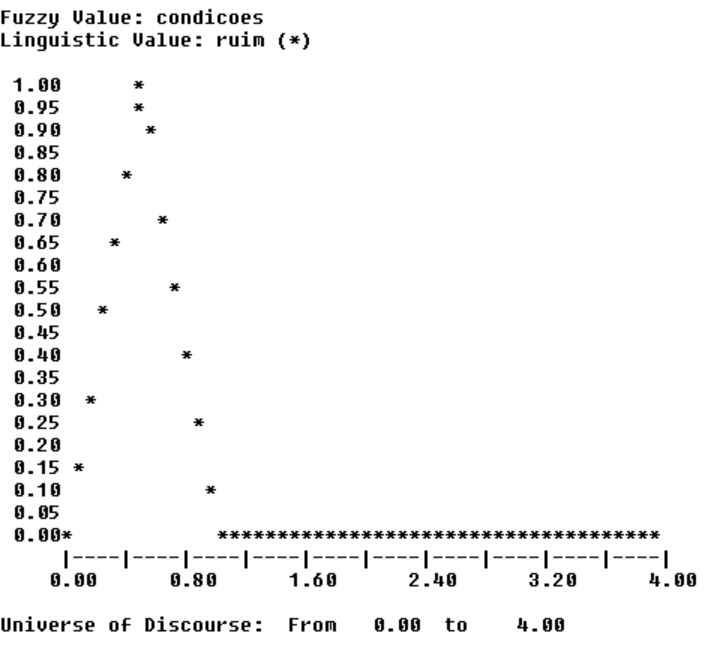


Figura 11. Condição para surf definida após executar aplicação

Podemos observar que a condição foi categorizada como “RUIM”, pelo fato de a altura das ondas ser baixa.

1. **Conclusão**

Uma implementação como essa pode ser implementada em aplicações simples, voltadas para surfistas. Em um futuro com melhorias também é possível adicionar outras variáveis para a execução de um cálculo mais preciso, como o clima (seco, chuvoso), força e altura da maré, etc.