Aplicação de lógica Fuzzy em um Sistema de Interface Humano-Computador baseado em Visão Computacional.

Cristiano Fraga G. Nunes[1,3], Paulo Vitor de C. Souza[1,2]

[1]Secretaria de Governança da Informação
Centro Federal de Educação Tecnológica de Minas Gerais (CEFET-MG)
Av. Amazonas, 5253 - Nova Suíça - CEP 30421-169 - Belo Horizonte, MG, Brasil

[2]Faculdade de Engenharia e Sistemas de Informação Faculdade UNA de Betim
Av. Gov. Valadares, 640 - Centro, Betim - MG, 32510-010

[3]Programa de Pós-Graduação em Modelagem Matemática e Computacional
Centro Federal de Educação Tecnológica de Minas Gerais (CEFET-MG)
Belo Horizonte, MG – Brasil

{cristiano,paulovitor}@cefetmg.br

Abstract. *This work proposes the implementation of a system for human-computer interface (HCI) based on computer vision using fuzzy logic. From this work we present an algorithm which will control the functions of a computer only with the use of body motions captured by one camera. The results of this work are presented relevant to a system of human-computer interface.*

Computer vision, Fuzzy, HCI, AI

Abstract. *Este trabalho propõe a implementação de um sistema de interface humano- computador (IHC) baseado em visão computacional utilizando lógica fuzzy. A partir deste trabalho apresentaremos um algoritmo que permitirá controlar funções de um computador ape- nas com a utilização de movimentos corporais captados por uma câmera. Os resultados obtidos neste trabalho se apresentam relevantes para um sistema de interface humano-computador.*

visão computacional, fuzzy, IHC, IA

1. Introdução

A interação entre humanos e máquinas acontece através de uma interface formada por software e hardware. Ela é utilizada, por exemplo, para a manipulação de periféricos de computadores e grandes máquinas [Prates and Barbosa 2003].

Associadas à visão computacional e técnicas de inteligência artificial, a interface humano- computador pode se tornar mais simples e agradável, de modo a tornar a máquina acessível a qual- quer pessoa. Segurança e controle de acesso tornam-se cada dia necessários para a melhor adequação e utilização de sistemas computacionais, garantindo a integridade e identificação de usuários em tarefas rotineiras [de Campos Souza 2015].

Com base nestes conceitos, este trabalho propõe o desenvolvimento de um sistema de interface humano-computador baseado em visão computacional com decisões de comandos que utiliza lógica fuzzy [Zadeh 1965] para avaliar a intensidade de atuação na máquina alvo.

A figura 1ilustra uma visão geral do sistema proposto identificando os elementos utilizados, a forma de detecção e o sistema fuzzy aplicado. Nela estão presentes elementos de hardware e software que auxiliam na identificação das imagens.

2. Caracterização do problema

O problema discutido neste artigo trata-se da codificação de um sistema de interface humano- computador. Tal sistema tem como objetivo facilitar a interação com máquinas e propor maneiras alternativas de manipulação, de modo a possibilitar uma pessoa a movimentar o cursor do sistema opera- cional apenas com movimentos faciais, por exemplo [Prates and Barbosa 2003]. Esse tipo de abordagem também pode auxiliar na segurança de sistemas computacionais, permitindo que a imagem captada seja utilizada para liberar recursos fundamentais e restritos dos sistemas computacionais [de Campos Souza 2015].

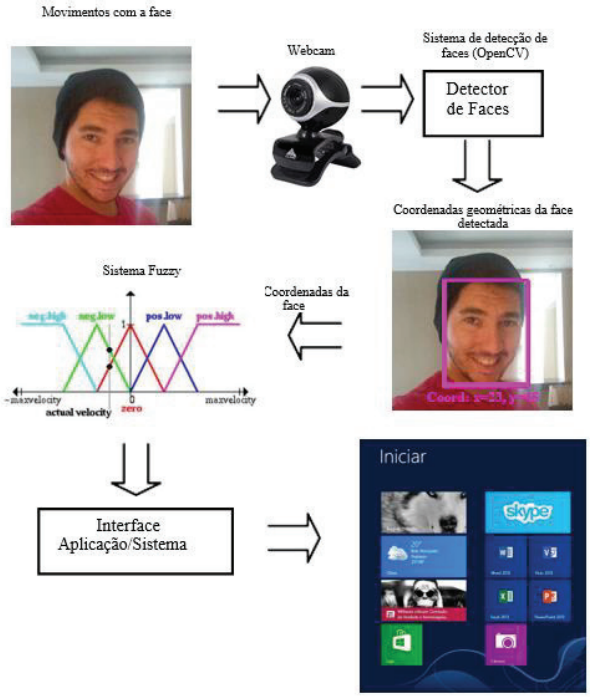

Figura 1. Visão geral do sistema proposto

3. Objetivos

3.1. Objetivo geral

O objetivo geral deste trabalho trabalho é propor o desenvolvimento um sistema de interface humano- computador baseado em visão computacional.

Este sistema agregou um módulo de reconhecimento de faces (previamente implementado pela biblioteca OpenCV) e utilizará lógica *fuzzy* para definir a atuação da interface no sistema operacional da máquina alvo. A atuação do sistema codificado se dará a partir dos movimentos da face humana.

O sistema visa propor métodos alternativos de se utilizar máquinas computacionais. Seja através da face, através dos olhos ou de qualquer outro meio. É proposta uma maneira mais natural de se utilizar tais sistemas, de modo que não seja

necessário a manipulação de um equipamento. Almeja-se tornar possível o uso de tais máquinas apenas com o próprio corpo [Prates et al. 2000].

3.2. Objetivos secundários

Como objetivo secundário, este trabalho pretende servir de apoio à dispositivos de entradas utilizados em jogos computacionais ou até mesmo aplicativos e sistemas para pessoas que apresentam dificuldades de utilizar os periféricos amplamente utilizados na atualidade, como por exemplo, pessoas com deficiência física [Prates et al. 2000].

4. Metodologia

Para a implementação do sistema, foi utilizado a biblioteca `OpenCV`[1] [Bradski 2000] para o reconhecimento de faces, e mapeado, através da lógica fuzzy, o movimento no sistema computadorizado provocado pelo movimento corporal. A Figura 2 representa o sistema em partes, demonstrando os passos essenciais para a realização do trabalho.

O algoritmo do sistema é exibido de acordo com o algoritmo 1.

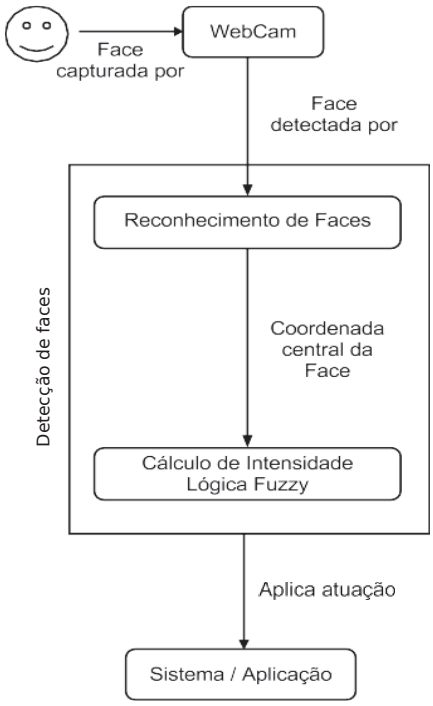

Figura 2. Visão do sistema em partes

Algoritmo 1: Algoritmo do sistema proposto

```
sair ← false
while sair = true do
    quadro ← CapturaQuadroCamera() face ← DetectaPrimeiraFace(quadro)
    coordenadas ← ExtraiCoordenadas(face) incremento ← Fuzzy(coordenadas)
    cursor ← cursor + incremento
end
```

[1] A biblioteca OpenCV (*Computer Vision Open Source*) é uma biblioteca de visão computacional popular que foi iniciada pela Intel em 1999. A biblioteca multi-plataforma define seu foco em tempo real de processamento de imagem e inclui patente livre de implementações dos algoritmos de visão computacional mais recentes

4.1. Sistema de detecção de faces

A detecção de faces é uma técnica que determina os locais e tamanhos de rostos humanos em imagens digitais. Ela detecta traços faciais e ignora qualquer outra coisa, como prédios, árvores e corpos [Turk and Pentland 1991].

Durante essa fase do projeto foi desenvolvida a parte do sistema responsável pela detecção de faces em um vídeo, sendo utilizado o algoritmo de Viola e Jones [Viola and Jones 2001], baseado em filtros de Haar [Tu et al. 2008] em cascata.

A biblioteca utilizada nesse projeto, OpenCV, já possui uma implementação da detecção de faces, bem como um classificador para detectar faces.

O processo de detecção de faces exige primeiro que se faça a conversão da imagem para tons de cinza e em seguida equalize o histograma da imagem para então efetuar o processo de detecção [Turk and Pentland 1991]. Esses passos são exibidos na figura 3.

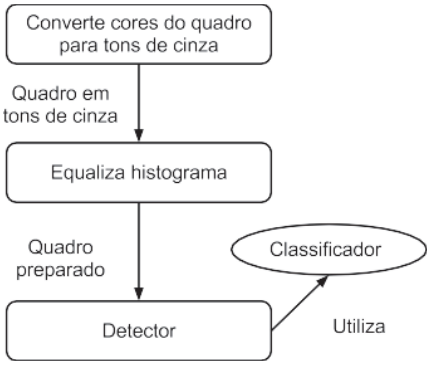

Figura 3. Sistema de detecção de faces.

4.2. Sistema Fuzzy

Para a implementação do sistema de controle *fuzzy*, foi utilizada a técnica de Mamdani proposta em [Castellano et al. 2003] e implementada em C++ [Love].

Toda a modelagem e simulação do sistema *fuzzy* foi feita através do software MATLAB [MATLAB 2010]. O sistema *fuzzy* utiliza como entrada as coordenadas **x** e **y** da face. A figura 4 exibe a modelagem do sistema *fuzzy* apresentada pela interface do *toolbox fuzzy* do MATLAB. Nele são apre- sentados o processo de entrada dos dados, a fuzzificação e a defuzzificação gerando as funções de pertinência triangulares que representam a imagem no espaço *fuzzy*.

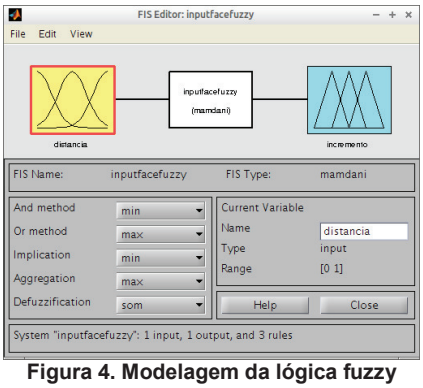

Figura 4. Modelagem da lógica fuzzy

Como pode ser observado na figura 4, o método de defuzzificação escolhido foi o "Menor dos Máximos" [Fortemps and Roubens 1996] com a finalidade de obter uma resposta linear na saída e reproduzir uma região de inatividade do cursor.

As curvas de pertinência foram ajustadas de modo a obter uma resposta confiável à aplicação hospedeira. As curvas de pertinência de entrada e saída são exibidas pelas figuras5e6respectivamente.

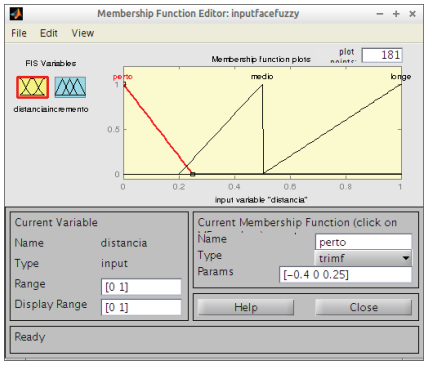

Figura 5. Curvas de pertinência da entrada Fuzzy

As regras do sistema fuzzy para as curvas de pertinência foram definidas de acordo com o algoritmo2.

A figura 7 mostra a curva de superfície da lógica fuzzy que foi gerada após a modelagem.

A curva de superfície da lógica fuzzy foi mapeada em um vetor de 1000 posições e inserida no código do sistema para efetuar o controle fuzzy sem a necessidade de avaliar as etapas fuzzy a cada iteração do algoritmo.

4.3. Região de atividade

Durante a detecção de movimentos, definiu-se uma zona de inatividade, que não provoca movimento no sistema computadorizado (área delimitada pela circunferência azul conforme a figura 8). Nas demais regiões, a pertinência nas coordenadas x e y foram proporcionais às componentes correspondentes do vetor OB com um limite máximo. A direção do movimento possui a mesma direção do vetor OB.

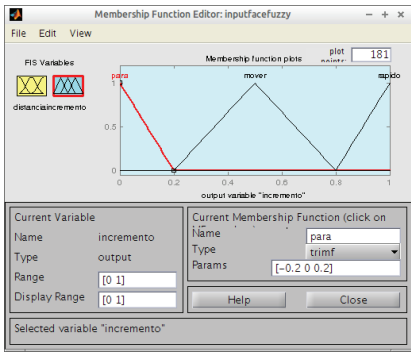

Figura 6. Curvas de pertinência da saída Fuzzy

Algoritmo 2: Regras de inferência do sistema Fuzzy

```
if distancia is perto then
| incremento ← para
end
if distancia is medio then
| incremento ← mover
end
if distancia is longe then
| incremento ← rapido
end
```

4.4. Integração com outros sistemas

O sistema proposto foi implementado de tal forma a ser compatível com qualquer programa que receba dispositivos de entrada do sistema operacional, podendo ser: jogos ou aplicativos específicos.

Para a implementação deste módulo, foram utilizadas bibliotecas específicas do sistema operacional Windows™ para envio de comandos traduzidos do sistema *fuzzy* para os demais aplicativos.

5. Resultados

Os resultados experimentais são descritos a seguir:

5.1. Execução do programa

Para a execução o programa é necessário que a pessoa esteja sobre iluminação direta, para que a câmera captura a face frontal para a detecção de face ocorrer. Se o projeto com o código fonte estiver aberto em alguma IDE[2] , basta compilar e rodar o mesmo. Caso contrário, execute o aplicativo.

Durante a execução do aplicativo uma janela de tamanho 640x480 pixels mostrará as imagens capturada pela webcam. Um quadrado é desenhado sobre a face detectada. É importante que apenas uma face esteja no campo de gravação da webcam. Nesse contexto do programa, o software deve identificar somente uma face. Caso nenhuma face seja detectada, o movimento do cursor é interrompido.

Uma vez que o sistema esteja em pleno funcionamento, basta ajustar a captura de vídeo de forma a centralizar seu rosto no vídeo. Para mover o cursor do sistema operacional, basta mover a face.

[2] IDE, do inglês *Integrated Development Environment* ou Ambiente Integrado de Desenvolvimento, é um programa de computador que reúne características e ferramentas de apoio ao desenvolvimento de software com o objetivo de agilizar este processo.

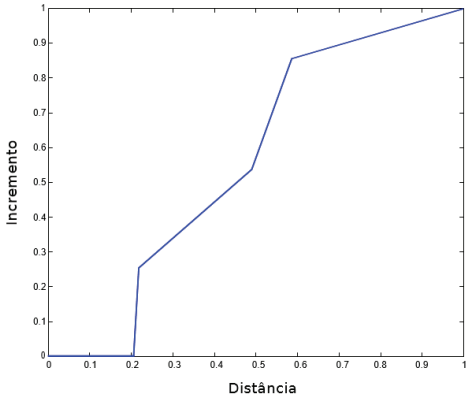

Figura 7. Curva de superfície produzida pela lógica fuzzy

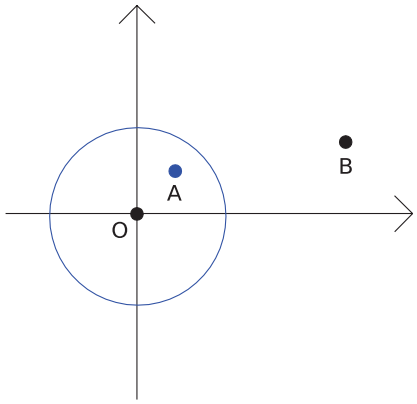

Figura 8. Região de inatividade do sistema

5.2. Desempenho

Através da figura 9 verificamos que o sistema é sensível ao tempo. Isso ocorreu nas primeiras fases do desenvolvimento, quando a imagem capturada da webcam era de alta resolução e seu processamento era mais lento. Nessas fases, a atualização de quadro do vídeo era de aproximadamente 1 segundo. Como efeito imediato disso, a posição do cursor só era atualizada, aproximadamente, a cada 1 segundo. A solução para o problema anterior foi reduzir a resolução e, consequentemente, o tamanho do quadro utilizado. Isso fez com que a imagem fosse processada em menos tempo e aumentou a frequência com que o cursor era atualizado, mas não tornou seu movimento suave o suficiente para que se assemelhasse ao movimento provocado por dispositivos utilizados atualmente (mouse ou touchpad).

A figura 9 exibe as duas curvas: a curva de azul representa a movimentação ideal do cursor e a curva de vermelho representa o resultado obtido com o sistema. Observa-se então um certo atraso de processamento.

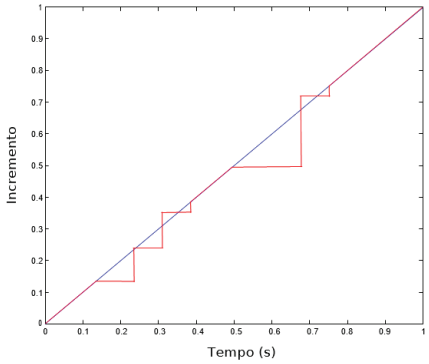

Figura 9. Curva ideal e curva real de movimentação

5.3. Sensibilidade

Durante a execução do programa, a face só é detectada se estiver de frente para a câmera. Mesmo de frente, sua detecção falha algumas vezes e a eficiência da detecção varia de acordo com a luminosidade e discrepância da face para com o fundo do vídeo. Quando a face não é detectada, o movimento do cursor é interrompido.

6. Conclusão

O sistema proposto mostrou-se eficaz no que se diz respeito à manipulação de máquinas através de movimentos corporais. Além de fornecer meios de facilitar a interação com a máquina, a modelagem do sistema se torna escalável e extensível, uma vez que não é necessário alterar a estrutura do programa: basta apenas remodelar a lógica fuzzy.

Para aplicações complexas, a lógica fuzzy é, sem dúvida, indicada para o controle e mapeamento de entradas e saídas.

A interface humano computador é facilitada por um sistema de reconhecimento facial eficiente e interpretativo, permitindo que as imagens sejam reconhecidas com melhor eficiência.

Como trabalho futuro poderá ser implementado uma resposta linear aos estímulos de entrada.

Dessa forma, a transição dos movimentos do cursor se torna mais suave.

Uma abordagem também se pensar é a implementação de uma suíte de interfaces para tornar possível o uso de computadores sem as mãos. Nesta interface, poderíamos selecionar características específicas do usuário, tornando o software adaptável às características físicas do mesmo.

Como o sistema proposto não possui um meio de calibração do ponto zero, poderá também ser implementado um sistema de calibração da câmera.

Pode-se ainda deixar como opcional a exibição das imagens capturadas pela webcam, a fim de melhorar o desempenho da aplicação, ou até mesmo paralelizar essa parte do resto da aplicação. Essas imagens poderiam ainda ser utilizadas para calibrar o sistema, ou verificar se a luminosidade é boa o suficiente para o sistema de identificação de faces.

Em relação à eficácia da detecção das faces novos trabalhos podem trabalhar na sua melhoria através do reconhecimento de imagens com treinamentos de faces.

Referências

[Bradski 2000] Bradski, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.

[Castellano et al. 2003] Castellano, G., Fanelli, A. M., and Mencar, C. (2003). Design of transparent mamdani fuzzy inference systems. In Abraham, A., Koppen, M., and Franke, K., editors, *Design and Application of Hybrid Intelligent Systems, HIS03, the Third International Conference on Hybrid In- telligent Systems, Melbourne, Australia, December 14-17, 2003*, volume 105 of *Frontiers in Artificial Intelligence and Applications*, pages 468–477. IOS Press.

[de Campos Souza 2015] de Campos Souza, P. V. (2015). O marco civil da internet: impactos e tecnolo- gias na proteção de sistemas governamentais. *Fonte*.

[Fortemps and Roubens 1996] Fortemps, P. and Roubens, M. (1996). Ranking and defuzzification methods based on area compensation. *Fuzzy sets and systems*, 82(3):319–330.

[Love] Love, T. CUED C++.

[MATLAB 2010] MATLAB (2010). *version 7.10.0 (R2010a)*. The MathWorks Inc., Natick, Massachu- setts.

[Prates and Barbosa 2003] Prates, R. O. and Barbosa, S. D. J. (2003). Avaliação de interfaces de usuário– conceitos e métodos.

[Prates et al. 2000] Prates, R. O., de Souza, C. S., and Barbosa, S. D. (2000). Methods and tools: a method for evaluating the communicability of user interfaces. *Interactions*, 7(1):31–38.

[Tu et al. 2008] Tu, Z., Narr, K. L., Dollár, P., Dinov, I., Thompson, P. M., and Toga, A. W. (2008). Brain anatomical structure segmentation by hybrid discriminative/generative models. *IEEE transactions on medical imaging*, 27(4):495–508.

[Turk and Pentland 1991] Turk, M. A. and Pentland, A. P. (1991). Face recognition using eigenfaces. In *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR'91., IEEE Computer Society Conference on*, pages 586–591. IEEE.

[Viola and Jones 2001] Viola, P. and Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. pages 511–518.

[Zadeh 1965] Zadeh, L. A. (1965). Fuzzy sets. *Information and Control*, 8:338–353.