Integração dos Ambientes Livres WEKA e ImageJ na Construção de Interfaces Guiadas por Sinais Visuais

Hemerson Pistori[1] **, Mauro Conti Pereira**[1]

[1]Centro de Ciência Exatas e Tecnológicas – Universidade Católica Dom Bosco
79117-900 Campo Grande, MS, Brasil

pistori@ec.ucdb.br, maurocp@ucdb.br

Abstract. *WEKA and ImageJ are free-softwares, not well-known in Brazil, but that offer a great variety of resources for, respectivelly, machine learning and digital image processing algorithms implementations. The present work, besides presenting these packages, describes how they have been integrated to create two prototypes of systems with a humam-computer interface based on visual-input. Two differents visual signs were used: eyes-gaze and hand-gestures, however, the proposed soluction may be generalized to any system with input captured from webcams.*

Resumo. *Os pacotes livre WEKA e ImageJ, ainda pouco conhecidos no Brasil, oferecem uma grande variedade de recursos para a implementação de algoritmos de aprendizagem de máquina e algoritmos de processamento digital de imagens, respectivamente. Além de apresentar esses pacotes, este trabalho descreve como os pacotes foram utilizados, de forma integrada, na criação de dois protótipos de sistemas com interface guiada por sinais visuais. Dois tipos de sinuais visuais foram utilizados, um relacionado com a direção do olhar e o outro com gestos realizados através das mãos. A solução proposta pode ser generalizada para qualquer tipo de sistema em que se pretende utilizar imagens capturadas através de uma webcam, na interação homem-máquina.*

1. Introdução

O baixo custo dos dispositivos de captura de imagens, juntamente com o aumento significativo na capacidade de processamento dos computadores pessoais, abrem espaço para o desenvolvimento de novos tipos de interfaces homem-máquina. A substituição dos tradicionais teclados e mouses por sinais visuais, a serem apresentados ao computador através de uma filmadora digital, oferece uma variada gama de aplicações que podem viabilizar e facilitar a utilização de computadores por segmentos da sociedade que apresentam determinadas deficiências físicas. Editores de texto e navegadores para Internet acionados através de movimentos do globo ocular, por exemplo, poderiam ser utilizados por pessoas tetraplégicas. Surdos também poderiam se beneficiar deste tipo de sistema, com a introdução de programas de computador capazes de interpretar línguas de sinais.

Stiefelhagen [Stiefelhagen et al., 1997] relata ter obtido uma precisão entre 1.3 e 1.9 graus em um sistema que permite o controle do ponteiro do mouse através do olhar. Gee e Cipolla [Gee and Cipolla, 1994], descrevem um sistema que infere a direção do olhar a partir da reconstituição de um modelo simplificado da cabeça, a partir da imagem projetada. Neste modelo, busca-se valorizar a posição relativa dos olhos, boca e nariz, que, segundo os autores, são atributos que variam pouco em relação a diferentes sujeitos. Wang e Sung [Wang and Sung, 2001] modelam a direção do olhar como sendo a direção da normal ao plano de suporte de cada íris. Aproximando

a forma da íris através de circunferências e assumindo raios, distâncias focais e fatores de escala conhecidos, é possível estimar a normal em questão a partir da projeção dos círculos.

Outro elemento visual utilizado na interface homem-máquina é a mão. Além de sua importância nas línguas de sinais para surdos, sistemas para rastreamento ($tracking$) das mãos humanas possuem aplicações que incluem jogos controlados por movimentos de mão, aparelhos de televisão acionados por sinais manuais [Freeman et al., 1996] e dispositivos que substituem o $mouse$ na interação com computadores.

Neste trabalho, mostraremos como alguns pacotes livres existentes puderam ser integrados na criação de dois protótipos de sistemas guiados por sinais visuais. Todos os pacotes utilizados são escritos em Java, o que facilita a portabilidade das soluções para novas plataformas. As novas classes que precisaram ser criadas para permitir a integração dos pacotes e o controle de webcams, a partir de um programa em Java, estão disponíveis na Internet [1]. Na próxima seção, descreveremos cada um dos pacotes livres utilizados no projeto. Na seção 3, apresentaremos a arquitetura de um sistema guiado por sinais visuais, bem como os problemas e as soluções encontradas, para que os pacotes existentes pudessem ser utilizados, em conjunto, na construção desse tipo de sistema. Os dois protótipos desenvolvidos e as conclusões são apresentadas nas duas últimas seções.

2. Descrição dos Pacotes Utilizados

Para implementar e testar técnicas de visão computacional e de processamento digital de imagens, utilizamos o pacote ImageJ [2], que é uma versão multiplataforma, ainda em desenvolvimento, do software NIH Image, para Macintosh. Entre os recursos oferecidos pelo pacote destacamos a disponibilidade, com programas-fonte abertos, de diversos algoritmos para: manipulação dos mais variados formatos de arquivo de imagens, detecção de bordas, melhoria de imagens, cálculos diversos (áreas, médias, centróides) e operações morfológicas. Esse software disponibiliza também um ambiente gráfico que simplifica a utilização de tais recursos, além de permitir a extensão através de *plugins* escritos em Java. Um outro fator importante para a sua escolha é a existência de uma grande comunidade de programadores trabalhando em seu desenvolvimento, com novos plugins sendo disponibilizados freqüentemente. Um desses plugins, o *HoughCircles*, foi desenvolvido pelo nosso grupo, durante a execução deste projeto, e está disponível na página do ImageJ.

O segundo pacote utilizado neste projeto foi o WEKA[3] (Waikato Environment for Knowledge Analysis) [Witten and Frank, 2000], também escrito em Java e com programas-fonte abertos. O WEKA é um ambiente bastante utilizado em pesquisas na área de aprendizagem de máquina, pois oferece diversos componentes que facilitam a implementação de classificadores e agrupadores (*clustering tools*). Além disto, esse ambiente permite que novos algoritmos sejam comparados a outros algoritmos já consolidados na área de aprendizagem, como o C4.5, o Backpropagation, o KNN e o naiveBayes, entre outros. Podemos com esse pacote obter facilmente resultados estatísticos comparativos da execução simultânea de diversos programas de aprendizagem em domínios variados, tais como o reconhecimento de caracteres e diagnóstico médico.

Por fim, utilizamos o pacote *Java Media Framework* [4] para captura de sinais temporais em tempo real. Este pacote permite que programas em Java possam acessar, por exemplo, imagens capturadas através de uma $webcam$ acoplada a um computador pessoal. O software abstrai detalhes das arquiteturas e interfaces de diferentes dispositivos de captura de imagem, facilitando assim a portabilidade dos aplicativos.

[1] http://www.ec.ucdb.br/ pistori/projetos/sigus/index.html
[2] http://rsb.info.nih.gov/ij/
[3] http://www.cs.waikato.ac.nz/ ml/weka/
[4] http://java.sun.com/products/java-media/jmf/

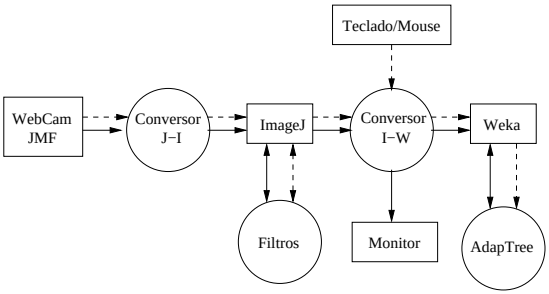

Figura 1: Sistema guiado por sinais visuais

3. Integração dos Pacotes

A figura 1 apresenta, utilizando uma notação gráfica similar à de um diagrama de fluxo de dados, a arquitetura básica de um sistema guiado por sinais visuais e aprendizagem. As linhas pontilhadas resumem o fluxo de informação relacionado com o treinamento do sistema, que acontece da seguinte forma: uma imagem do sinal (e.g. uma mão humana realizando o sinal manual para a letra p) a ser aprendido é capturado através de um $webcam$, com auxílio do JMF, juntamente com uma classificação para o sinal, que é fornecida através do teclado ou mouse (e.g. digitação da letra p). A imagem é transmitida, após ser devidamente convertida (Conversor J-I), para o módulo de processamento digital de sinais (ImageJ). Nesse módulo são realizadas transformações na imagem que podem envolver filtros adicionais (Filtros), indisponíveis no pacote ImageJ. O resultado dessa transformação é convertido e repassado, juntamente com a classificação do sinal, para o módulo de aprendizagem (Weka), que aciona o algoritmo de aprendizagem, neste caso, o AdapTree-E [Pistori, 2003], baseado em tecnologia adaptativa [Neto, 2001].

Depois que os sinais são aprendidos, o sistema entra no modo de funcionamento representado pelas linhas cheias da figura 1. Nesse modo, a entrada consiste apenas das imagens capturadas pela $webcam$ (sem as informações de treinamento). As imagens são processadas através dos mesmos filtros utilizados no treinamento e as informações resultantes são passadas para o módulo de aprendizagem. O módulo de aprendizagem deve agora oferecer uma resposta à entrada, com base no modelo abstraído a partir dos exemplos fornecidos anteriormente. Esta resposta é apresentada ao usuário através de algum dispositivo de saída, como por exemplo, o monitor do computador.

Nosso trabalho incluiu o desenvolvimento dos módulo de conversão J-I (JMF para ImageJ) e I-W (ImageJ para Weka), além é claro, da implementação do mecanismo de aprendizagem, AdapTree-E, e de filtros específicos para as duas aplicações. O resultado é um grupo de classes, escritas em Java, que podem ser especializadas ou modificadas na implementação de outros sistemas com características semelhantes.

4. Protótipos Implementados

Para realizar alguns dos nossos experimentos com interfaces baseadas na direção do olhar, implementamos um simples jogo da velha. Existem apenas nove regiões de interesse para o usuário nesse jogo: as posições do tabuleiro onde podemos marcar os círculos ou as cruzes. Fazendo com que a apresentação visual do tabuleiro ocupe grande parte do monitor, conseguimos relaxar o grau de precisão na detecção da direção do olhar. Além disto, é bastante natural a transposição da interface por mouse para a interface pelo olhar.

Inicialmente, existe um período de aprendizagem, em que usuário deve olhar para cada uma das nove regiões do tabuleiro. A captura de imagens de treinamento é iniciada e finalizada clicando-se o mouse. Depois que uma quantidade pré-definida de exemplos é capturada, o sistema infere uma primeira árvore de decisão, e o usuário pode começar a jogar "com o olhar". O módulo de aprendizagem não trabalha diretamente sobre a imagem obtida, mas sobre um conjunto de parâmetros dela extraídos pelo módulo de processamento digital de imagens. Uma descrição mais detalhada sobre as técnicas utilizadas pode ser encontrada em [Pistori, 2003].

Nós implementamos também um protótipo de um editor para símbolos alfabéticos da LIBRAS. Primeiramente, ocorre a fase de treinamento, quando o usuário deve executar os sinais manuais, com uma das mãos, e utilizar a outra para digitar a letra correspondente. Diversas imagens da mão são capturadas, até que uma tecla qualquer seja pressionada. Este procedimento é repetido, diversas vezes, para todas as letras desejadas: normalmente, quanto mais exemplos forem colhidos na fase de treinamento, maior será a precisão obtida na fase de reconhecimento. Quando uma tecla especial, reservada, é pressionada, o sistema induz a árvore de decisão adaptativa inicial, e o usuário pode então iniciar a edição do texto utilizando sinais manuais.

5. Conclusões

Apresentamos neste artigo um conjunto de pacotes e bibliotecas, alguns desenvolvidos por nós, outros disponíveis gratuitamente na Internet, que podem ser utilizados na implementação de sistemas guiados por sinais visuais. A integração dos pacotes pré-existentes não foi uma tarefa trivial, uma vez que, além de complexos, estes pacotes não foram projetados para trabalhar em conjunto. Um objetivo para o futuro é a construção de um ferramenta com interface gráfica que permita mais facilmente a utilização destes pacotes e bibliotecas, que hoje é feita basicamente através da alteração de programas-fonte escritos em Java. Também seria interessante buscar uma alternativa ao pacote JMF, que embora gratuito, não é livre (os programas-fonte não estão disponíveis).

Referências

Freeman, W. T., Tanaka, K., Ohta, J., and Kyuma, K. (1996). Computer vision for computer games. In *2nd International Conference on Automatic Face and Gesture Recognition*, pages 100–105, Killington, VT, USA.

Gee, A. H. and Cipolla, R. (1994). Determining the gaze of face in images. Technical Report CUED/F-INFENG/TR 174, Cambridge University, Trumpington Street, Cambridge CB2 1PZ, England.

Neto, J. J. (2001). Adaptative rule-driven devices - general formulation anda case study. In *CIAA'2001 Sixth International Conference on Implementation and Application of Automata*, pages 234–250, Pretoria, South Africa.

Pistori, H. (2003). *Tecnologia Adaptativa em Engenhara de Computação: Estado da Arte e Aplicações*. PhD thesis, Universidade de São Paulo, São Paulo, Brasil.

Stiefelhagen, R., Yang, J., and Waibel, A. (1997). Tracking eyes and monitoring eye gaze. *Proceedings of the Workshop on Perceptual User Interfaces (PUI'97).*, pages 98–100.

Wang, J. G. and Sung, E. (2001). Gaze determination via images of irises. *Image and Vision Computing*, 19(12):891–911.

Witten, I. H. and Frank, E. (2000). *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann.