Visão Computacional e Reconhecimento de Comandos de Voz Aplicados na Interação com Jogos e Ambientes 3D

Edirlei E. Soares de Lima[1] Cristina Ruby[2] Cesar T. Pozzer[3] Cassia Neves da Silva[4]

[1] PUC-Rio, Depto. de Informática, Brasil
[2] UFSC, Depto. de Informática e Estatística, Brasil
[3] UFSM, Depto. de Eletrônica e Computação, Brasil
[4] UEG, Depto. de Contabilidade, Brasil

Resumo

Os jogos de computador e vídeo games têm passado por diversas evoluções nos últimos anos, principalmente no que diz respeito à qualidade gráfica e realismo de suas cenas. Porém a maneira como o jogador interage com estes não sofreu grandes mudanças. Apenas recentemente que novas formas de interação vêm sendo apresentadas ao publico em geral. Este artigo apresenta uma proposta de um sistema de interação com ambientes virtuais 3D e jogos de computador através do uso de técnicas de visão computacional e reconhecimento de comandos de voz.

Keywords: Interação, Jogos, Visão Computacional.

Contato:
[1] elima@inf.puc-rio.br
[2] cristina.ruby@gmail.com
[3] pozzer@inf.ufsm.br
[4] cassiapnn@gmail.com

1. Introdução

Nas últimas décadas houve um crescimento gradativo das inovações presentes nos computadores, porém, até hoje, é normal que toda a interação entre o usuário e a máquina seja feita através do teclado, mouse ou joystick. Atualmente, buscam-se alternativas para estas formas convencionais de interação, principalmente no que diz respeito aos jogos eletrônicos. Desde o surgimento dos primeiros jogos, a maneira como usuário interage com eles evoluiu notoriamente. Entretanto, ainda hoje é comum que para esta interação seja necessário um contato entre o usuário e uma parte física do computador ou vídeo game. Por isso, novas formas de interação vêm sendo pesquisadas [Paula *et al.* 2006] [Teixeira *et al.* 2006] [Prihatmanto 2007] [Kuo e Chirravuri 2009], muitas destas apontam as técnicas de visão computacional e reconhecimento de comandos por voz como alternativas para a interação entre o jogador e o jogo.

Neste artigo é apresentada uma proposta de um sistema de interação com jogos de computador baseado no reconhecimento de movimentos da face e comandos de voz do jogador. Estes são captados com a utilização de um microfone e de uma webcam, em seguida processados através de técnicas de visão computacional e reconhecimento de padrões para serem interpretados e as suas respectivas ações executadas no jogo. O artigo esta organizado da seguinte maneira: na sessão 2 são apresentados trabalhos relacionados ao uso de visão computacional e comandos de voz em jogos. Na sessão 3 são apresentados detalhes sobre a implementação dos módulos do sistema. A sessão 4 apresenta os resultados obtidos e a sessão 5 finaliza este artigo apresentando as conclusões e perspectivas de trabalhos futuros.

2. Trabalhos Relacionados

Diversas pesquisas sobre novas formas de interação dos usuários com jogos eletrônicos têm sido realizadas nos últimos anos. Dentre as pesquisas que utilizam visão computacional, pode-se citar Paula *et al.* [2006], que propõem o "Câmera Kombat", um jogo de luta multijogador que utiliza uma webcam para detectar quando sequências pré-determinada de movimentos são realizadas. Em uma abordagem um pouco diferenciada, Teixeira *et al.* [2006], apresenta um sistema, chamado de "GeFighters", para a interação com jogos baseada em marcadores, onde o usuário deve segurar em cada uma das mãos um marcador e através de uma webcam este marcador é mapeado para ações no jogo.

Em pesquisas referentes ao uso de comandos de voz para a interação em jogos, pode-se citar Prihatmanto [2007], que apresenta um sistema para jogos de estratégia no qual se alia o desenvolvimento de um dispositivo de toque com comandos de voz. A interação através de comandos de voz compreende as ações que serão executadas no jogo. De acordo com os experimentos realizados pelo autor, concluiu-se que os usuários conseguem utilizar livremente os dois modos de interação, existindo uma tendência de utilização de comandos de voz para escolher a ação a ser feita e do "Stick Of Fire" para enfatizar o local em que a ação deve ser feita. Seguindo uma linha de pesquisa semelhante, Kuo e Chirravuri [2009] apresentam a criação de um jogo de xadrez com movimentação das peças através de comandos por voz.

Existem também aplicações comerciais que buscam introduzir no mercado novas formas de interação. Um exemplo é o Kinect (conhecido também como Projeto Natal) [ProjectNatal 2010] desenvolvido pela Microsoft para o console XBox 360. O sistema faz uso de uma câmera RGB para o reconhecimento do rosto do jogador e outra infravermelha para o reconhecimento dos movimentos e a profundidade do ambiente. No lançamento do projeto foi demonstrada a sua usabilidade na interação com jogo utilizando movimentos corporais. Entretanto, poucas publicações sobre o projeto foram feitas e os resultados de desempenho e precisão não foram analisados em nenhum teste acadêmico. Testes informais revelaram dificuldades no reconhecimento de movimentos de jogadores sentados, além de atrasos na execução de ações em relação aos movimentos do jogador [Esperon 2010].

3. Desenvolvimento

O sistema para a interação em jogos proposto neste trabalho é dividido em dois módulos distintos. Primeiro, o módulo responsável pelo reconhecimento dos movimentos da face do usuário. Este módulo utiliza uma webcam para capturar em tempo real imagens da face do usuário, que, em seguida, são processadas através de técnicas de visão computacional para que seja realizado o reconhecimento dos movimentos do jogador. Os movimentos são então interpretados e as ações correspondentes a eles são executadas no jogo. O segundo módulo do sistema é o reconhecedor de comandos de voz. Este módulo faz o uso de um microfone para captar a fala do usuário e em seguida processá-la. Quando uma palavra correspondente a um comando é detectada, a ação associada a ela é executada no jogo. A arquitetura completa do sistema pode ser vista na figura 1.

Para tornar o sistema de interação com jogos flexível, ele foi desenvolvido na forma de um módulo extra para a *game engine* 3D Game Builder [Lima 2008].

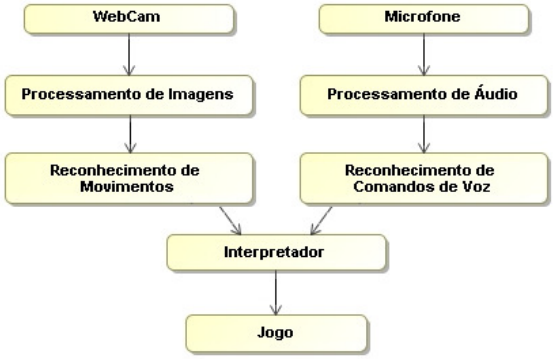

Figura 1: Arquitetura do sistema de interação.

3.1 Reconhecimento de Movimentos

Para o desenvolvimento do módulo de detecção de movimentos foi utilizada a biblioteca OpenCV, que consiste em uma coleção de funções para o processamento de imagens, reconhecimento de objetos e movimentos, além disso, proporciona um código eficiente e otimizado. A linguagem de programação C++ foi utilizada para a codificação do sistema. Para tornar o módulo o mais genérico possível, ele foi desenvolvido na forma de uma DLL (*Dynamic Link Library*). Isto permite que outros programas, além do 3D Game Builder, possam utilizar as suas funções independentemente da linguagem em que eles sejam desenvolvidos.

Para a detecção da face do usuário através da câmera foi utilizado o classificador Haar Classifier e a base de treinamento "haarcascade_frontalface_alt2.xml", ambos do próprio OpenCV. O Haar Classifier consiste em um método de classificação proposto por Viola e Jones [2001] e aprimorado por Lienhart e Maydt [2002]. O método foi criado inicialmente para a detecção de faces, mas pode ser aplicado para a detecção de qualquer tipo de objeto rígido. O método baseia-se em somas e subtrações de regiões retangulares das imagens que posteriormente são submetidas a um *threshold* para a extração das características [Bradski e Kaehler 2008]. A grande vantagem deste classificador é que ele rejeita rapidamente regiões onde é pequena a possibilidade de se encontrar o objeto buscado. Cerca de 70% a 80% das imagens que não contém o objeto são rejeitadas nas duas primeiras comparações com a base de treinamento. Além disso, segundo Bradski e Kaehler [2008] o método possui altas taxas de reconhecimento, aproximadamente 99,9% com poucos falsos negativos ou objetos não reconhecidos, além de uma taxa de rejeição de menos de 50%.

A análise da imagem capturada pela câmera é feita em tempo real, cada frame é processado e a face é detectada. O número de frames capturados por segundo é ajustado automaticamente com base no tempo gasto para o processamento e reconhecimento da face. Caso o tempo gasto em cada frame seja alto, o número de frames capturados será reduzido. Caso o tempo seja baixo, o número de frames é aumentado. Após a imagem ser capturada através da câmera, ela é aplicada no classificador Haar. Como resultado tem-se uma sequência de retângulos com a posição dos objetos detectados. Dentre este conjunto de objetos detectados, somente a primeira face encontrada na imagem é utilizada, mesmo que mais pessoas estejam na frente da câmera – apenas uma delas terá o controle da aplicação.

A detecção dos comandos a serem executados baseia-se no posicionamento da face do usuário. Um retângulo central, como ilustrado na figura 2, define a área de movimentação. Com base na posição do ponto central da face do jogador, é possível diferenciar 5 possíveis estados, ou seja, 5 possíveis locais onde o ponto central da face do jogador pode estar (centro do retângulo ou em uma das 4 laterais). Para cada um destes 5 estados permite-se a associação de uma ou mais ações para serem executadas no jogo. Estas ações podem ser programadas através da linguagem script do 3D Game Builder.

Um exemplo de utilização é o uso dos 5 estados para o controle da movimentação de um personagem em cenário 3D. Enquanto o centro da face do usuário estiver dentro do retângulo central nenhum movimento precisa ser executado. Quando a face mover-se para um dos lados de fora do retângulo (*left* ou *right*), o personagem no jogo pode virar para o lado correspondente. Ao mover a face para a parte superior (*front*) do retângulo, o personagem pode andar para frente. Ao mover o centro da face para a área inferior (*back*) do retângulo, o personagem pode andar para trás. Além disso, o sistema também permite a execução de movimentos em conjunto, por exemplo, ao mover a face para o canto superior direito (*right, front*) o personagem no jogo iria virar para a direita enquanto anda.

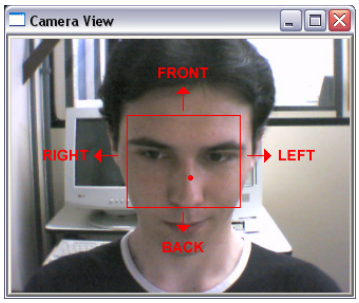

Figura 2: Visualização do sistema de detecção de movimentos.

3.2 Reconhecimento de Comandos de Voz

O módulo para o reconhecimento dos comandos de voz também foi desenvolvido em linguagem C++ e, assim como o módulo de reconhecimento de movimentos, foi implementado em uma DLL para facilitar a sua utilização por outros aplicativos.

O primeiro passo para o reconhecimento dos comandos de voz é o pré-processamento do áudio. Um sinal pode conter grandes variações nos valores das amplitudes. Para eliminar estas variações é necessário normalizá-las. A forma de normalização adotada neste trabalho consiste em dividir cada amostra do sinal pelo valor da amostra de maior amplitude. Desta forma, todos os valores das amplitudes permanecem entre -1 e 1, não apresentando alterações no gráfico da onda. Da mesma forma que ocorre com a amplitude, a amostragem do sinal pode ser diferente para cada gravação. Para amenizar este problema aplicou-se a discretização do sinal, ou seja, coleta-se apenas um número de amostras, sempre igual, ao longo do sinal de todas as palavras. Neste trabalho foram utilizadas 2000 amostras para cada sinal.

Após o pré-processamento é realizada a extração das características. Para isso as 2000 amostras de amplitude da palavra são divididas em 49 frames ou janelas; para cada janela somam-se todos os valores das amplitudes e então se calcula a sua média e seu desvio-padrão (equação 1). Também se calcula a média e o desvio-padrão de todo o sinal, totalizando 100 características para cada palavra.

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{i=N}\left(X_i - \overline{X}\right)} \qquad (1)$$

Utilizando-se das características das palavras gera-se uma base de treinamento para o classificador com diversas amostras de palavras que irão gerar o vocabulário do jogo. Para o reconhecimento de padrões foram utilizados os algoritmos KNN (K-Nearest Neighbors) [Cover e Hart 1967] e SVM (Support Vector Machine) [Vapnik 1995]. É permitido que o usuário selecione qual dos algoritmos vai ser utilizado para o reconhecimento. O KNN é mais rápido que o SVM, porém possui uma menor taxa de acertos no reconhecimento das palavras.

A captura do áudio é realizada em tempo real através de um microfone. Para evitar que ruídos do ambiente sejam processados, o sistema analisa continuamente o áudio e somente realiza a classificação de sinais cujas amplitudes sejam maiores do que as do som do ambiente. Quando uma variação de amplitude é captada, o sistema inicia a gravação do sinal de entrada. No momento que a amplitude do sinal retorna para estado próximo ao do ambiente, o sistema realiza a classificação do sinal que foi gravado anteriormente. Sobre o sinal captado são executados o pré-processamento e a extração de suas características. Em seguida essas características são enviadas para o classificador, previamente treinado, que utilizará um dos algoritmos de reconhecimento de padrões para a determinação da classe a qual o sinal (palavra) de entrada pertence.

Para evitar que sons ou palavras que não fazem parte do vocabulário do sistema sejam classificados dentre as possíveis classes, o algoritmo de classificação incorpora uma classe de rejeição. Desta maneira, quando as características de um sinal de entrada se diferem muito das utilizadas no vocabulário do sistema, o sinal de entrada é simplesmente descartado.

Para a integração do módulo de reconhecimento de comandos de voz com 3D Game Builder, foi adicionado a linguagem script da *game engine* o evento "OnVoiceCommand". Este evento é executado no jogo sempre que um comando de voz é reconhecido. Isto permite que o desenvolvedor possa programar qualquer tipo de ação para o jogo com base em um comando de voz.

Um dos quesitos importantes para um bom resultado no reconhecimento dos comandos de voz é à base de treinamento. Por este motivo, permiti-se que o jogador, na primeira vez que for iniciar um jogo com este sistema, grave a sua própria base de treinamento, melhorando consideravelmente o percentual de reconhecimento das palavras.

4. Resultados

Para validar o sistema de interação proposto foram realizados testes de desempenho, precisão e usabilidade. Todos os experimentos foram realizados em um Intel Core 2 Quad 2.40 GHz com 4 GB de RAM. A seguir serão apresentados os resultados dos testes.

Para avaliar o módulo de reconhecimento de comandos de voz, primeiramente foi realizado um teste de precisão utilizando uma base de treinamento composta por 8 palavras distintas e 160 exemplos de treinamento adquiridos de diferentes indivíduos. Utilizou-se esta base para classificar os exemplos de uma base de testes composta por 100 exemplos. Obteve-se uma taxa de reconhecimento de 84% utilizando o algoritmo KNN e de aproximadamente 93% utilizando o SVM. Como a taxa de acertos depende diretamente do treinamento do classificador, estes valores podem variar conforme o tamanho da base de treinamento. Com uma base maior pode ter-se uma maior porcentagem de acertos. Para avaliar o desempenho do módulo foi realizada a classificação de uma sequência de 30 palavras e, para cada uma delas, calculou-se o tempo necessário para o pré-processamento do áudio e a classificação da palavra. Como resultado, obteve-se a média de 0.35 segundos para classificação de uma palavra desconhecida utilizando o algoritmo KNN e de 0.68 segundos utilizando o algoritmo SVM.

Para avaliar o módulo de reconhecimento de movimentos da face do usuário, primeiramente foi realizado o teste de precisão. Como mencionado anteriormente utiliza-se no sistema a base de treinamento "haarcascade_frontalface_alt2.xml" fornecida na biblioteca OpenCV. Testou-se a base de treinamento para processar um conjunto de 100 imagens obtidas através da webcam em resolução de 640x480. No resultado do teste obteve-se 68 faces encontradas, 3 faces perdidas e 7 falsos positivos (faces encontradas onde não existiam faces). Para avaliar o desempenho do módulo, calculou-se o tempo necessário para classificar o mesmo conjunto de

100 faces do teste anterior. Como resultado, obteve-se a média de 72.4 milissegundos para detectar a face em cada imagem.

Para avaliar a usabilidade do sistema de interação foram desenvolvidos alguns exemplos de jogos utilizando o reconhecimento de movimentos e o reconhecimento de comandos de voz. Entre os jogos criados, pode-se cita o "Spider Robot" (figura 5). O objetivo do jogo é fazer a aranha robô mover algumas caixas que estão no cenário para os seus devidos lugares. Neste exemplo, é demonstrada a precisão do sistema de reconhecimento de movimentos da face do jogador na movimentação da aranha para mover as caixas. Neste caso, a precisão é de extrema importância para permitir que o jogador se aproxime corretamente da caixa e consiga empurrá-la exatamente para a sua devida posição.

Figura 5: Jogo Spider Robot utilizando sistema de interação.

Para demonstrar que o sistema de interação se tornou genérico, podendo ser utilizado em qualquer estilo de jogo, foi desenvolvido um protótipo de um RPG onde o jogador utiliza a movimentação de sua face para andar com o personagem principal pelo cenário e comandos de voz para interagir com outros personagens. Por exemplo, o comando de voz "oi" é utilizado para iniciar um dialogo com outro personagem, o comando "atacar" faz o personagem executar um ataque. Outros comandos, como "pular" e "correr", também podem ser usados. Durante os diálogos, também é possível interagir com comandos de voz. Por exemplo, quando o jogador é questionado sobre algo cujas possíveis respostas sejam sim ou não, basta responder com o comando de voz correspondente, no caso "sim" ou "não".

5. Conclusões e Trabalhos Futuros

Neste artigo foi apresentada a proposta de um sistema que alia técnicas de visão computacional e de reconhecimento de comandos de voz para a interação com jogos de computador. O sistema apresentou bons resultados, tanto em questão de performance como em usabilidade. A união da interação através de movimentos da face do usuário, juntamente com comandos de voz, se mostrou eficaz e simples de ser utilizada pelos jogadores.

Os experimentos de desempenho foram realizados em um Intel Core 2 Quad 2.40 GHz, onde cada módulo do sistema utilizava um núcleo do processador para a sua execução. O sistema se demonstrou bastante eficaz em questões de desempenho. Como trabalho futuro, pretende-se analisar o desempenho do sistema em outras configurações, assim como analisar a possibilidade de migração de parte do processamento do reconhecimento de movimentos e de voz para GPU. Pretende-se também explorar outras formas de interação através de movimentos corpóreos, utilizar outras partes do corpo do jogador, assim como incorporar ao sistema a possibilidade de utilizar sequência de movimentos para a interação. Desta forma, pretende-se reduzir a quantidade limitada de ações que podem ser executadas com base nos movimentos do jogador.

O desenvolvimento do sistema como um módulo extra para a *game engine* 3D Game Builder vai permitir que novos jogos, utilizando estas formas de interação, possam ser criados. Os módulos de interação apresentados neste artigo, assim como as DLLs e as informações sobre a sua utilização, serão disponibilizados nas próximas versões do 3D Game Builder através do site oficial da *game engine*: www.eternix.com.br

Referências

BRADSKI, G., KAEHLER, A., 2008. Learning OpenCV: Computer Vision with the OpenCV Library, O'Reilly.

COVER, T.M., HART, P.E., 1967. Nearest neighbor pattern classification. IEEE Transactions on Information Theory 13 (1): 21-27.

ESPERON, M., 2010. Preocupações e 1 Mentira sobre o Kinect. Available from: http://istoerablog.blogspot.com/2010/06/5-preocupacoes-e-1-mentira-sobre-o.html. [Accessed 17 June 2010].

KUO, M., CHIRRAVURI, V., 2009. Voice-Controlled Chess Game on FGPA Using Dynamic Time Warping. Available from: https://courses.csail.mit.edu/6.111/f2008/projects/varunc_Project_Proposal.pdf. [Accessed 17 June 2010].

LIENHART, R., MAYDT, J., 2002. An Extended Set of Haar-like Features for Rapid Object Detection, EEE ICIP, vol. 1, pp. 900-903.

LIMA, E.S., 2008. 3D Game Builder: Uma Game Engine para a Criação de Jogos e Ambientes 3D, Trabalho de Graduação - Universidade do Contestado, Porto União, SC.

PAULA, L.P.R., BONINI, N.R., MIRANDA, R.F., 2006. Camera Kombat - Interação Livre para Jogos. *In: SBGames 2006*, Recife, Brasil.

PRIHATMANTO, A.S., 2007. Multimodal Interaction utilizing Voice Command in Real-Time Strategy Game *In: ICEEI2007,* Bandung, Indonesia.

PROJECTNATAL, 2010. Microsoft Wave - Project Natal. Available from: http://www.microsoft.com/uk/wave/hardware-projectnatal.aspx. [Accessed 17 June 2010].

TEIXEIRA, J.M., FARIAS, T., MOURA, G., LIMA, J.P., PESSOA, S., TEICHRIEB, V., 2006. GeFighters: an Experiment for Gesture-based Interaction Analysis in a Fighting Game. *In: SBGames 2006*, Recife, Brasil.

VAPNIK, V., 1995. The Nature of Statistical Learning Theory, Springer, New York.

VIOLA, P., JONES, P., 2001. Rapid object detection using a boosted cascade of simple features. *In: CVPR01*, pp. 511–518.