colloquium exactarum

MANIPULAÇÃO DE GESTOS COM O KINECT EM AULAS EXPOSITIVAS

GESTURE MANIPULATION WITH KINECT ON LECTURES

Wilian Alves da Silva[1], Francisco Assis da Silva[1], Daniela Tereza Ascencio Russi[1], Mário Augusto Pazoti[1], Robson Augusto Siscoutto[1], Almir Olivette Artero[2], Marco Antonio Piteri[2]

[1] Universidade do Oeste Paulista – UNOESTE, Faculdade de Informática – FIPP, Presidente Prudente, SP. [2]Universidade Estadual Paulista – FCT/Unesp, Presidente Prudente, SP. e-mail: robson.siscoutto@unoeste.br

RESUMO - Este trabalho apresenta métodos para reconhecimento de gestos das mãos e braços, nos quais os resultados do reconhecimento são aplicados em objetos multimídia como vídeos, imagens e em apresentações em sala de aula. Para o reconhecimento dos gestos das mãos foi realizada a extração de defeitos de convexidade. Os defeitos foram extraídos através do processamento da imagem de profundidade do sensor Kinect com auxílio da biblioteca OpenCV para obter o fecho convexo e contorno das mãos. O reconhecimento de gestos considerando a posição dos braços de acordo com o movimento foi alcançado utilizando o algoritmo *Dynamic Time Warping*, através dos pontos do esqueleto capturados pelo sensor Kinect. Foram realizados experimentos para os gestos das mãos, que apesar das limitações impostas pelo sensor, mostrou-se o potencial do uso de reconhecimento de gestos não só para aulas expositivas, mas também para outras aplicações com necessidades semelhantes.
Palavras-chave: Kinect; Reconhecimento de Gestos; Aulas expositivas.

ABSTRACT - This paper presents methods for gesture recognition of hands and arms, where the results are applied on multimedia objects like videos, images and presentations in class. For hand gestures recognition it was performed the extraction of convexity defects. The defects were extracted by processing the depth frame from the Kinect sensor using OpenCV library to obtain the Convex Hull and hand contours. The recognition of gestures considering the position of the arms according with the movement was achieved using the Dynamic Time Warping algorithm, through the skeleton points captured by the Kinect sensor. Experiments were performed for hand gestures, which despite the limitations imposed by the sensor, showed potential for using gesture recognition not only for lectures, but also for other applications with similar needs.
Keywords: Kinect, Gesture Recognition, Lecture.

Recebido em: 03/10/2014
Revisado em: 07/11/2014
Aprovado em: 18/12/2014

1 INTRODUÇÃO

Interfaces naturais são um meio de proporcionar aos usuários uma comunicação com dispositivos de forma intuitiva e de fácil aprendizado. Seu uso ganhou maior espaço com a facilidade de se adquirir tecnologias com superfície multitoque encontrados em smartphones, tablets e dispositivos para reconhecimentos de gestos como o Kinect da Microsoft. O reconhecimento de gestos envolve a análise e modelagem do movimento, o reconhecimento de padrões e aprendizado de máquina, trata-se de uma tarefa complexa (TOGORES, 2011). Para Santos (2011), o uso de uma ferramenta que permita a interação entre o ser humano e o computador utilizando apenas as mãos é uma possibilidade que pode ser aproveitada em diversas áreas, como a da educação, medicina, entretenimento ou até mesmo aplicações militares.

O reconhecimento de gestos sem o apoio de artefatos acoplados às mãos tem sido buscado por diversos autores como Alvarenga (2011), Masutani et al. (2012), Prado Neto e Bruno (2012), Pulkit e Atsuo (2012) e Raheja, Chaudhary e Singal (2011) com o intuito de proporcionar métodos mais naturais de interação. Estes utilizaram componentes de entrada que variam desde uma simples webcam até sensores com melhores atributos, como é o caso do Kinect da Microsoft.

Na literatura foram encontrados autores que buscaram a realização do reconhecimento de gestos de diversas formas. No trabalho de Alvarenga (2011), por exemplo, foram utilizados os dados do esqueleto capturados pelo Kinect para treinamento de uma Rede Neural MLP (*Multi-Layer Perceptron*). Outros autores concentraram seu estudo no uso de Modelos Ocultos de Markov (TOGORES, 2011), no algoritmo *Dynamic Time Warping* (RYAN, 2012) e na comparação de ângulos das partes do corpo através do esqueleto capturado pelo Kinect (SILVEIRA, 2011).

O sensor Kinect é um dispositivo que tem grande potencial para facilitar a interação natural entre o homem e o computador. Até o momento, na área da educação, são poucas as aplicações utilizando os recursos oferecidos pelo dispositivo (HSU, 2011). As salas de aula e laboratórios já contam com inúmeros recursos para deixar as aulas mais ricas como equipamentos de projeção e quadros interativos (SAVI, 2009).

A proposta deste trabalho é realizar o reconhecimento e processamento de gestos das mãos e braços utilizando o dispositivo Kinect para aplicá-los na manipulação de objetos multimídia em salas de aula. A ideia principal da proposta visa ampliar a

capacidade de reconhecimento do corpo humano que já se pode fazer com o uso do Kinect, adicionando o reconhecimento de gestos com as mãos e braços.

O artigo está organizado da seguinte forma: a Seção 2 apresenta as características e componentes do sensor Kinect e o Kit de desenvolvimento; na Seção 3 são demonstradas as teorias, métodos utilizados; na Seção 4 é mostrada a metodologia para o desenvolvimento da proposta; na Seção 5 são discutidas as soluções aplicadas e resultados encontrados; e por último, na Seção 6 são mostradas as conclusões do trabalho.

2. O Sensor Kinect

O Kinect é um dispositivo para captura de movimentos, desenvolvido originalmente para o videogame Xbox 360. Ele se diferencia de outros dispositivos do gênero por não necessitar de um dispositivo em mãos, pois é capaz de detectar a posição do corpo, movimento, voz e gestos (JANA, 2012).

O sensor se trata de um dispositivo horizontal com seis componentes chave (Figura 1):

- Câmera RGB: suporta resoluções de 640x480 em 30 FPS a 1280x960 em 12 FPS;
- Emissor Infravermelho: emite luz infravermelha no ambiente;
- Sensor Infravermelho de profundidade: analisa a luz infravermelha refletida para gerar as informações de profundidade;
- Motor: utilizado para ajustar o ângulo de visão do sensor verticalmente;
- Conjunto de Microfones: permite a captura de sons com a direção da origem;
- LED: indica a situação do funcionamento do sensor.

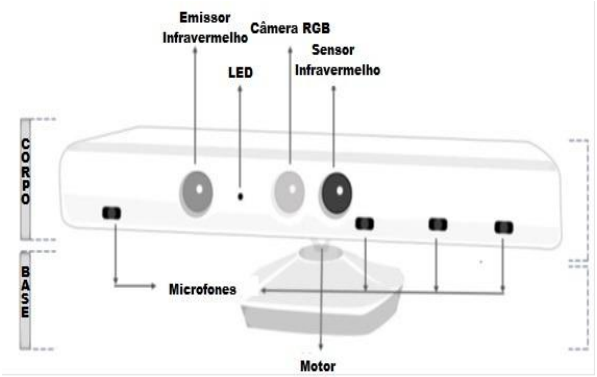


Figura 1. Componentes do Kinect.

Fonte: (JANA, 2012).

Os sensores de profundidade do Kinect consistem de um emissor infravermelho e um sensor de profundidade infravermelho. O emissor emite constantemente uma luz infravermelha em forma de pontos pseudorrandômicos (que apenas aparentam ser aleatórios) sobre tudo que está a sua frente. Esses pontos são lidos pelo sensor de profundidade infravermelho que os converte em informação de profundidade medindo a distância entre o sensor e o objeto de onde o ponto foi lido (JANA, 2012).

A versão do Kinect utilizada nesse trabalho foi a Kinect for Xbox, sendo que apenas os componentes que processam as informações de profundidade (emissor e sensor infravermelho) foram necessários para o desenvolvimento da proposta.

2.1 KINECT FOR WINDOWS SOFTWARE DEVELOPMENT KIT

O SDK (*Software Development Kit*) é um Kit de desenvolvimento de software, constituído por ferramentas e APIs (*Application Programming Interface*) para a criação de aplicações voltadas ao Microsoft Windows. Além disso, inclui os drivers importantes para a comunicação entre o computador e o sensor Kinect e códigos com exemplos de uso dos componentes.

A versão utilizada nesse trabalho do SDK para o Kinect foi a 1.7, lançada em março de 2013, e possui algumas facilidades novas em relação às anteriores. Uma delas é o *KinectInteractions* que corresponde a um conjunto de recursos que permitem que aplicações utilizando Kinect incorporem interatividade baseada em gestos. Alguns de seus recursos são a identificação e interação de uma das mãos onde o usuário pode pressionar botões e realizar o gesto de "agarrar" para mover objetos em uma lista, como se fosse uma barra de rolagem. Para o desenvolvimento de uma aplicação usando o Kinect, além desses recursos providos pelo SDK, faz-se necessário utilizar o *Kinect Developer Toolkit* 1.7 (MSDN LIBRARY, 2013b). Este torna mais fácil a obtenção e manipulação de informações mais específicas como é o caso do mapeamento das mãos na tela, que é feito de forma que o usuário consiga alcançar toda a área visível confortavelmente evitando movimentos exagerados.

3 CONCEITOS FUNDAMENTAIS

Este trabalho foi realizado com o objetivo de reconhecer os gestos das mãos e braços humanos de forma que pudesse ser aplicado na manipulação de objetos multimídia. As funcionalidades buscadas para o desenvolvimento da aplicação foram baseadas nas facilidades que poderiam ser utilizadas em aulas expositivas, onde atualmente, o professor ou palestrante faz um maior uso de apresentações, imagens e vídeos.

Toda a informação que diz respeito ao usuário é capturada pelo sensor Kinect junto ao Kinect for Windows SDK que propõe rotinas que facilitam a extração de informações básicas (ex: coordenadas do esqueleto).

A identificação dos dedos foi feita com auxílio do EmguCV (EMGUCV, 2013), que é uma plataforma que permite o uso de funções do OpenCV em linguagens suportadas pelo .Net *framework*. Foram utilizadas funções para extração de

contornos e o algoritmo Fecho Convexo (*Convex Hull*), que a biblioteca também oferece.

Para o reconhecimento dos gestos considerando o movimento dos braços, foi utilizado o algoritmo *Dynamic Time Warping* aplicado nas informações das coordenadas do esqueleto obtidas através do sensor Kinect.

3.1 ALGORITMO FECHO CONVEXO

De acordo com Gonzalez e Woods (2007), o fecho convexo *H* de um conjunto arbitrário *S* é o menor conjunto convexo contendo *S*. O conjunto *D* obtido a partir da diferença *H* – *S* contém os defeitos de convexidade do conjunto *S*.

Para Esperança e Cavalcanti (2002), o fecho convexo é a formação de um polígono simples (aproximação) com base em um conjunto de pontos. Este é utilizado como um método que precede outros algoritmos sobre estes conjuntos. O polígono formado não deve ocupar mais espaço do que o próprio conjunto de pontos.

O Algoritmo 1, retirado de Forster (2007), mostra a construção do fecho convexo para pontos em 2D seguindo os conceitos da Varredura de Graham (ESPERANÇA; CAVALCANTI, 2002).

Algoritmo 1. Construção do fecho convexo em 2D

1: Dados vértices ordenados em y
Insira o primeiro vértice e o segundo vértice nas listas esquerda e direita.
2: Para cada outro vértice *V*
Percorrer lista da esquerda inversamente removendo vértices até garantir que o ângulo das arestas em relação à horizontal é sempre decrescente.
Percorrer a lista da direita inversamente removendo vértices até garantir que o ângulo das arestas em relação à horizontal é sempre crescente.
Insira *V* nas duas listas.
3: Fim.

A Figura 2 mostra a formação do fecho convexo para um conjunto de sete pontos, sendo que cada quadro representa as etapas definidas no Algoritmo 1.

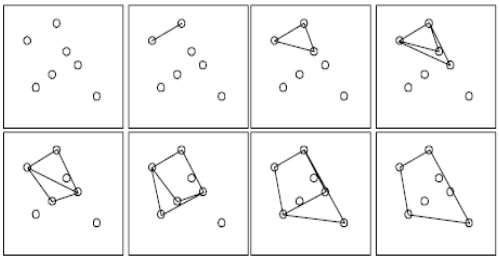


Figura 2. Formação do fecho convexo. Fonte: (FORSTER, 2007).

3.2 ALGORITMO DYNAMIC TIME WARPING

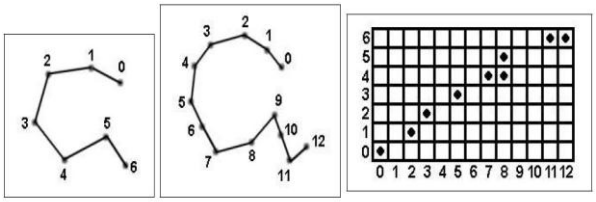
Segundo Senin (2008), o algoritmo *Dynamic Time Warping* (DTW) é conhecido em muitas áreas. Foi introduzido nos anos 60, explorado nos anos 70 na aplicação de reconhecimento de fala e mais tarde tornou-se muito utilizado no reconhecimento de gestos, caligrafia, assinaturas digitais e outros.

O DTW é um algoritmo extremamente eficiente na medição de similaridades que minimiza os efeitos de distorção permitindo uma transformação "elástica" das séries cronológicas para detectar formas similares com diferentes fases, ou seja, ele pode ser

usado para determinar se duas curvas ou sequências são similares, sendo que cada uma pode variar de acordo com o tempo ou velocidade.

Para que o DTW possa comparar duas curvas deve ser criado um caminho correspondente através da matriz de custo acumulado, que seria representado por uma lista de combinações de pontos das curvas. Para cada combinação de pontos é calculada a distância entre os dois, geralmente é usada a distância Euclidiana.

As distâncias calculadas são somadas e o resultado é normalizado dividindo ele pelo número de combinações no caminho correspondente. O valor resultante é a distância entre as curvas. A Figura 3 ilustra a formação da matriz utilizando as duas curvas a serem comparadas e também uma representa como seria formado o caminho correspondente.



(a) (b) (c)

Figura 3. Curvas (a e b) e caminho correspondente (c).

Fonte: (NIELS, 2004).

No Algoritmo 2, adaptado do trabalho de Senin (2008), é mostrado o método de criação da matriz de custo acumulado. Essa é a parte mais importante do algoritmo *Dynamic Time Warping* já que é possível obter o caminho correspondente e a distância (diferença) entre as curvas.

Algoritmo 2. MatrizCustoAcumulado(X,Y)

```
1: Inicio  n ← |X|,  m ← |Y|, dtw[] ← new [n x m]
2: dtw(0, 0) ← 0
3: Para i = 1 até n faça
        dtw( i , 1 ) ← dtw( i  - 1 , 1 ) + custo( i , 1 )
4: Para  j = 1 até m faça
        dtw( 1 , j ) ←  dtw(1; j - 1) + custo( 1 ,  j )
5: Para i = 1 até  n faça
   Para j = 1 até m faça
        dtw( i , j ) ← custo( i , j )+ min{ dtw( i - 1 , j );
dtw( i ,  j- 1 ); dtw( i - 1 ,  j – 1 ) }
6: retorna dtw
```

Inicialmente o Algoritmo 2 define o tamanho da matriz através da quantidade de pontos na curva (linha 1). Nas linhas 2, 3 e 4 são feitas as inicializações da matriz calculando as distâncias (Euclidiana) iniciais, sendo que o procedimento mostrado elimina a necessidade de se preencher a tabela com valores que representem o infinito positivo e aumenta o desempenho do algoritmo. A linha 5 representa o cálculo das distâncias restantes na matriz e, por fim, o resultado é retornado (linha 6).

Com a matriz de custos definida, para encontrar o caminho correspondente é necessário percorrer a matriz de forma inversa a qual foi mostrada no Algoritmo 2 (linha 5), sempre buscando o menor valor.

A distância ou diferença entre as curvas pode ser obtida através da divisão do maior valor presente no caminho correspondente pelo tamanho da curva

comparada (quantidade de pontos). A partir deste valor é possível determinar um limite para comparação e validar ou não a similaridade entre as curvas.

4 METODOLOGIA

Nessa seção são mostrados os procedimentos para a realização do reconhecimento de gestos considerando a contagem de dedos das mãos e gestos que dependem da análise a trajetória dos braços durante a execução de movimentos. Também são mostrados os gestos definidos e problemas encontrados.

4.1 RECONHECIMENTO DE GESTOS CONSIDERANDO O DESLOCAMENTO DOS BRAÇOS

Para o reconhecimento dos gestos envolvendo o braço por completo, é necessário que toda a sua trajetória seja armazenada e comparada. A realização dessa comparação foi feita com o uso do algoritmo *Dynamic Time Warping*. Este permite obter bons resultados com um conjunto de treinamento menor quando comparado a outros que são usados no reconhecimento de padrões. Assim como visto no trabalho de Andreasen e Hummersgaard (2011), em que experimentos foram realizados com o uso do Modelo Oculto de Markov e o *Dynamic Time Warping* para uma mesma aplicação.

A extração das coordenadas do esqueleto importantes para essa tarefa é realizada na captura das informações do esqueleto, que é através de um evento que pode ser acessado por meio do SDK (com o nome de *SkeletonFrameReady*) e permite obter as informações sobre o posicionamento e partes do corpo do usuário. A cada ciclo realizado pelo evento, os dados do esqueleto são filtrados de forma que as coordenadas importantes para os gestos com braços como mãos, pulsos e cotovelos (Figura 4) sejam armazenadas em um vetor.

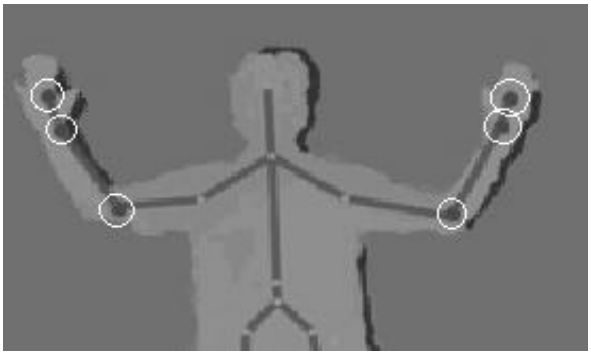


Figura 4. Pontos importantes para trajetória dos gestos.

Além disso, as coordenadas dos ombros também são armazenadas para que possa ser feita a normalização dos pontos. A normalização é necessária para que os gestos a serem reconhecidos se tornem independentes da posição do usuário diante do sensor.

O tratamento das coordenadas envolve dois passos, sendo o primeiro centralizar todas as coordenadas capturadas de acordo com os ombros e o segundo

realizar a normalização de acordo com a distância entre eles.

Para cada gesto são consideradas as coordenadas capturadas e processadas em quadros (*frames*) de forma intercalada. Todas as sequências ficam armazenadas em uma lista para que o algoritmo DTW possa analisá-la e realizar o reconhecimento dos gestos. A lista contém as séries de coordenadas normalizadas *X* e *Y* dos três pontos importantes de cada braço que é capturado.

O reconhecimento dos gestos ocorre durante a execução do evento de captura das informações do esqueleto, em que os movimentos do usuário são mantidos em um *buffer* constantemente. Ao atingir o número mínimo de quadros capturados, o gesto executado tem a sua última posição comparada com a última posição dos gestos armazenados, da mesma forma como sugerido pelo projeto "KinectDTW" em CODEPLEX (2011). No caso dessa comparação resultar em uma diferença aceitável, é então aplicado o algoritmo *Dynamic Time Warping* para a comparação do gesto como um todo.

4.2 RECONHECIMENTO DE GESTOS CONSIDERANDO OS DEDOS DAS MÃOS

O reconhecimento dos gestos das mãos foi baseado na contagem de dedos e posicionamento entre elas. A contagem de dedos considerou dedos ativos aqueles que não estivessem fechados ou grudados um no outro. Para a realização da contagem foi necessário o uso do algoritmo de fecho convexo utilizando a biblioteca OpenCV.

O primeiro passo foi realizar o isolamento das mãos, desse modo não é necessário o processamento do conteúdo total das imagens. Essa etapa foi executada utilizando os meios que o Kinect disponibiliza para captura das coordenadas das mãos e informações de profundidade (*SkeletonStreamReady* e *DepthFrameReady,* respectivamente*,* acessados por meio do SDK). Para cada mão foi definida uma área retangular de tamanho suficiente para envolvê-la de acordo com o intervalo de distância entre o usuário e o sensor que foi determinado entre 1.25 e 1.6 metros. Este tamanho foi definido em 150 x 100 pixels. Através de testes considerando as dimensões da imagem obtida que é de 640 x 480 pixels, o tamanho definido permite a movimentação das mãos sem a perda de dados por posicionamento, desde que o usuário se mantenha no centro da imagem capturada pelo Kinect.

Durante a captura dos quadros de profundidade, todos os pixels dentro do intervalo de 8.5 centímetros à frente e atrás da coordenada *Z* (profundidade) das mãos foram definidos como branco e os demais como preto. Isso possibilitou obter imagens

segmentadas das mãos (Figura 5), tornando desnecessário o reconhecimento por pigmentação da pele. A distância definida para comparação da coordenada *Z* permitiu eliminar pixels do corpo do usuário que poderiam aparecer dependendo do posicionamento das mãos.

Figura 5. Isolamento da mão.

O próximo passo foi extrair os defeitos de convexidade da mão. Para isso foi utilizado o OpenCV, em que é extraído o contorno aproximado da mão e aplicado o algoritmo de fecho convexo nesse contorno. Os defeitos de convexidade podem ser observados como os espaços na cor preta entre os dois contornos da Figura 6.

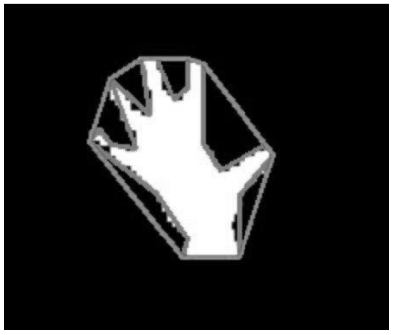


Figura 6. Contorno aproximado da mão e fecho convexo.

Para a classificação dos defeitos de convexidade que representem dedos, é usada a heurística apresentada nas Equações 1, 2 e 3 (PULKIT; ATSUO, 2012):

$$dedo = \begin{cases} 1 \quad \text{se}\,(s_y < b_y \quad \text{ou} \quad d_y < b_y) \ \text{e} \\ (s_y < d_y) \ \text{e} \ l_d > \dfrac{altura\_caixa}{n} \\ 0 \quad \text{caso} \quad \text{contrário} \end{cases} \qquad (1)$$

$$l_d = \sqrt{(s_x - d_x)^2 + (s_y - d_y)^2} \qquad (2)$$

$$dedos\_qtd = \sum dedo \qquad (3)$$

onde *s* é o ponto inicial do defeito de convexidade, b_y é a coordenada *y* do ponto central da caixa (definida como retângulo de menor área que envolve o contorno das mãos), *d* é o ponto de profundidade do defeito de convexidade. O valor do parâmetro "altura_caixa" utilizado na Equação 1 é definido com a altura de *b*.

Após a aplicação da heurística, os defeitos considerados como dedos são identificados assim como é mostrado na Figura 7. As retas foram desenhadas apenas para demonstração usando os pontos *s* e *d* de cada defeito de convexidade com suas coordenadas *x* e *y*.

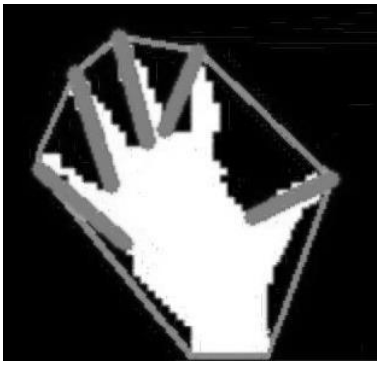


Figura 7. Defeitos de convexidade reconhecidos como dedos.

4.3 GESTOS DEFINIDOS

Neste trabalho foram definidos três gestos que dizem respeito à movimentação dos braços. Isso foi feito baseando-se nos tipos de objetos abordados (Seção 4.4) e nos gestos utilizados atualmente em smartphones que possuem telas sensíveis ao toque, assim como pode ser observado na Figura 8.

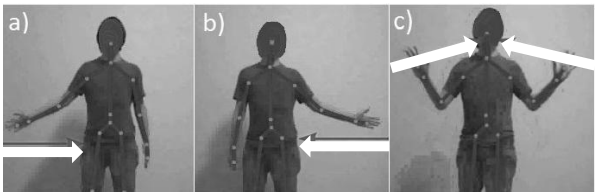


Figura 8. Gestos com os braços: (a) Anterior (b) Próximo (c) Fechar.

Em relação às mãos, foram definidos cinco gestos (Figura 9). Foi necessário levar em consideração a limitação da qualidade da imagem de profundidade, a qual não permitiu a contagem de dedos em todos os estados possíveis da mão devido às exceções que podem ocorrer na contagem de defeitos de convexidade.

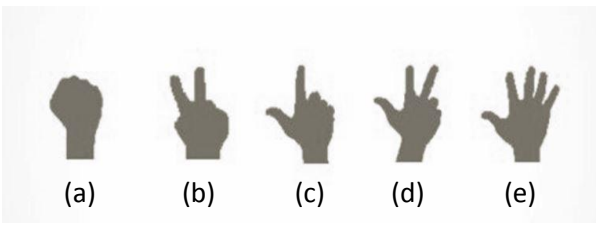


Figura 9. Gestos com as mãos: (a) utilizado para arrastar, (b) selecionar conteúdo, (c) usado como *zoom*, (d) mostrar menu de itens, (e) sem ações.

A contagem de apenas um dedo, por exemplo, pode ser confundida com o defeito de convexidade extraído entre o pulso e a parte externa da mão onde dependendo do formato da mão, gera um defeito com dimensões suficientes em relação ao que acontece quando se tem apenas um dedo levantado (Figura 10). O mesmo ocorre na diferenciação de quatro e cinco dedos (mão aberta).

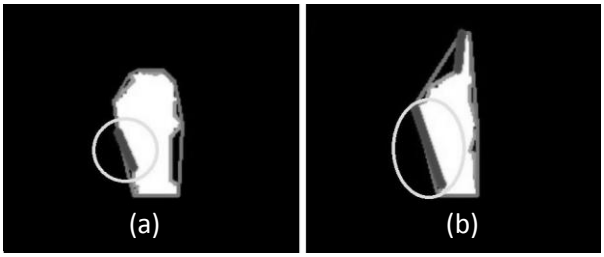


Figura 10. Falha na classificação dos defeitos de convexidade. (a) Mão fechada (b) Apenas um dedo levantado.

Para diferenciar os gestos (b) e (c) da Figura 9 foi necessário calcular o espaço entre as pontas dos dedos. Esse espaço pode alterar de acordo com a distância entre o usuário e o sensor, já que a imagem das mãos fica maior ao se aproximar e menor ao se afastar. Através de testes buscando a menor taxa de erros possível, foi definido um espaço entre os dedos para diferenciar estes gestos de forma que a distância entre o

usuário e o sensor possa variar de 1.2 a 1.6 metros.

4.4 OBJETOS MANIPULADOS E FORMAS DE INTERAÇÃO

Foram definidos três tipos de objetos a serem manipulados pelos gestos já discutidos (Seção 4.3), sendo estes, imagens, vídeos e apresentações multimídia. Para isso, os formatos de conteúdo comumente utilizados em aulas expositivas e apresentações foram escolhidos.

Para as imagens, toda a manipulação se baseou nos gestos levando em consideração os dedos. No gesto utilizado para *zoom*, por exemplo, foram armazenadas as coordenadas iniciais das mãos ao realizar o gesto e então foi calculada a porcentagem de aumento ou redução quando ambas as mãos mudam para um estado diferente, efetivando as alterações no objeto. Outro gesto utilizado para este tipo de objeto foi o de arrastar, definido para quando se tem todos os dedos da mão fechados.

No caso dos vídeos e apresentações foram incorporadas ações buscando gestos simples com os braços, como demonstrado nos gestos denominados “Anterior”, “Próximo” e “Fechar“ (Figura 8).

Os tipos de objetos definidos permitiram o desenvolvimento de uma aplicação capaz de indexar os materiais (imagens, vídeos e apresentações) de acordo com a necessidade do usuário em um menu de itens (Figura 11) que é facilmente acessado com o gesto definido como (d) na Figura 9 pela mão esquerda. Ao acessar o menu é possível escolher um dos materiais com a mão direita através do gesto (b) da Figura 9. Também foi definida a navegação pela lista do menu (quando se tem muitas imagens, vídeos ou apresentações) ao posicionar a mão nas extremidades direita ou esquerda da tela da aplicação.

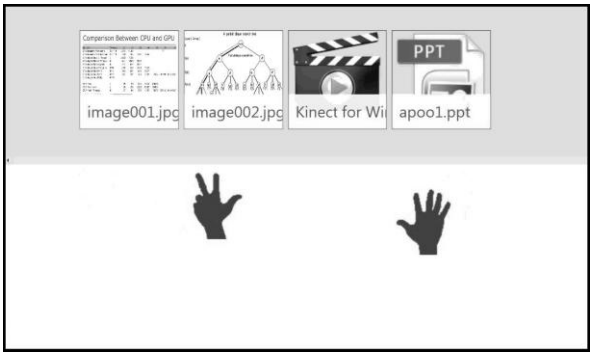


Figura 11. Navegação pelos materiais do usuário.

5 RESULTADOS

No desenvolvimento dos gestos das mãos, encontrou-se dificuldade na contagem dos dedos. Devido à qualidade das imagens com informações de profundidade capturadas, a contagem retorna resultados muito imprecisos. Para resolver este problema, foi necessário aplicar o cálculo de maior ocorrência de gestos. Este cálculo registrou as poses das mãos dentro de sete quadros capturados e processados intercaladamente e então foi feita a busca da pose que ocorreu o maior número de vezes.

Isso permitiu diminuir consideravelmente as taxas de erro durante a execução da aplicação. No entanto, se de um lado o cálculo de maior ocorrência ajudou a diminuir a taxa de erros, do outro ele afetou de forma negativa no tempo de resposta. Por precisar esperar por no mínimo catorze quadros para sua realização, foi adicionado um atraso de aproximadamente 467 milissegundos no tempo de resposta do reconhecimento do gesto, levando em consideração a captura de trinta quadros por segundo da imagem de profundidade. Este número pode ser maior já que o sensor Kinect retorna quadros nulos aleatoriamente durante sua execução (JANA, 2012).

O reconhecimento das mãos e dos braços necessitou que fossem definidas situações de uso para cada um separadamente. Executar os dois métodos ao mesmo tempo com objetos ativos de tamanho considerável (Ex: imagens com resoluções maiores) causou aumento na taxa de erros de reconhecimento e aplicação das ações.

Para a realização dos experimentos relacionados ao reconhecimento das mãos foram capturadas 100 amostras de gestos após o cálculo de maior ocorrência e aproximação de contornos com o mesmo valor. Foram levadas em consideração as distâncias aproximadas de 1.4 e 1.9 metros entre as mãos e o sensor Kinect. O teste foi repetido para cada gesto separadamente. As mãos ficaram em movimento e manteve-se o mesmo gesto até que as amostras fossem capturadas.

Nas Figuras 12 e 13 é possível observar as limitações impostas pela qualidade da imagem de profundidade na realização do reconhecimento dos gestos das mãos.

Figura 12. Imagens de profundidade capturadas na distância aproximada de 1.4 metros.

Figura 13. Imagens de profundidade capturadas na distância aproximada de 1.9 metros.

A Figura 13, por exemplo, demonstra que a qualidade da imagem faz com que os dedos das mãos fiquem deformados ou até mesmo desapareçam, impossibilitando a extração dos defeitos de convexidade corretamente.

Observando as Tabelas 1 e 2 é possível notar o peso que a imagem de profundidade capturada pelo sensor Kinect exerce na ocorrência de erros do

reconhecimento de gestos em distâncias superiores a 1.6 metros.

Tabela 1. Relação de acerto para aproximadamente 1.4 metros de distância do sensor para 100 amostras.

Gestos	Mão Direita (erros)	Acerto (%)	Mão Esquerda (erros)	Acerto (%)
	0	100%	0	100%
	0	100%	0	100%
	2	98%	6	94%
	0	100%	0	100%
	2	98%	5	95,00%

Na Tabela 2, por exemplo, o gesto de mão fechada só conseguiu uma taxa de acerto maior porque a mão fechada é facilmente representada, o que não acontece com o gesto de mão aberta, já que nessa distância é difícil o sensor conseguir diferenciar as lacunas entre dedos abertos. Nota-se também que existe uma pequena diferença entre a taxa de acerto das mãos, onde na maioria dos casos a mão direita consegue melhor resultado.

Tabela 2. Relação de acerto para aproximadamente 1.9 metros de distância do sensor para 100 amostras.

Gestos	Mão Direita (erros)	Acerto (%)	Mão Esquerda (erros)	Acerto (%)
	1	99%	1	99%
	45	55%	77	23%
	49	51%	70	30%
	28	72%	42	58%
	70	30%	90	10%

Apesar dessa diferença ser relativamente pequena, é conhecido o problema presente na versão 1.7 do SDK, o qual apresenta falhas de precisão quanto ao posicionamento da mão esquerda (MSDN LIBRARY, 2013a). Esse problema pode ser amenizado alterando a aproximação do contorno extraído da imagem da mão esquerda.

Em relação aos gestos com os braços, não foram realizados experimentos. Assim como pode ser visto no trabalho de Andreasen e Hummersgaard (2011), as falhas podem ocorrer por motivos que variam desde a dificuldade até a precisão da execução de algum gesto. Usuários treinados em menor tempo tendem a errar o posicionamento correto do gesto. Além disso, deve ser considerado o parâmetro de diferença aceitável entre os gestos sendo executados e os gestos armazenados, o qual

pode ser mudado de acordo com a necessidade.

6. CONSIDERAÇÕES FINAIS

Este trabalho apresenta uma maneira mais intuitiva e interativa de comunicação com dispositivos computacionais para aulas expositivas como sendo a ideia inicial, que favoreceria um professor a ter um controle maior e mais amplo de um conjunto de objetos (vídeos, imagens, apresentações etc.) que poderiam ser utilizados. Apesar dos problemas encontrados quanto à limitação imposta pela imagem de profundidade do sensor Kinect e eventuais falhas dos métodos utilizados, mostrou-se o potencial desse tipo de aplicação. Através dela foi possível manipular objetos comumente usados em apresentações sem o uso de dispositivos acoplados às mãos ou ao restante do corpo. Além disso, o uso de reconhecimento de gestos em aulas expositivas abrange conteúdos que tem utilidade em outras áreas como é o caso da medicina, que pode aproveitar a manipulação de imagens para visualização de radiografias em salas de cirurgia, evitando o contato direto com o material. Também podem ser criados painéis interativos, o que é muito interessante para atividades relacionadas à propaganda, já que o uso de tecnologias atrai atenção e pode até proporcionar uma visualização de produtos em lojas ou eventos.

REFERÊNCIAS

ALVARENGA, M. L. T. **Reconhecimento de gestos 3D utilizando o dispositivo Kinect**. [São Paulo?]: Universidade de São Paulo, Instituto de Ciências Matemáticas e de Computação, 2011.

ANDREASEN, L.; HUMMERSGAARD, R. **Natural user interface for web browsing on a PC**. Aalborg: Aalborg University, 2011.

CODEPLEX. **Kinect SDK dynamic time warping (DTW) gesture recognition**. 2011. Disponível em: < http://kinectdtw.codeplex.com/>. Acesso em: 29 mar. 2013.

EMGUCV. **EmguCV OpenCV in .NET**. Disponível em: <http://www.emgu.com/>. Acesso em: 26 ago. 2013.

ESPERANÇA, C.; CAVALCANTI , P. R. **Geometria computacional**: fecho convexo. Rio de Janeiro: UFRJ, 2002. 33 slides. Disponível em: <http:// orion.lcg.ufrj.br/gc/download/**Fecho**%20**Convexo**.ppt>. Acesso em: 12 dez. 2013.

FORSTER, Carlos H.Q. **Visão computacional**: descrição e feições. São José dos Campos: Instituto Tecnológico de Aeronáutica, 2007. 35 slides. Disponível em: <http://www.comp.ita.br/~forster/CC-222/lecture/08-Descricao-Feicoes.pdf>. Acesso em: 12 dez. 2013.

GONZALEZ, R. C.; WOODS, R. E. **Digital image processing**. 3rd. [S.l.]: Prentice Hall,. 2007.

HSU, H. J. **The pontential of Kinect as interactive educational technology**. Taiwan: Fo Guang University, 2011. 5 p.

JANA, A. **Kinect for Windows SDK programming guide.** Birmingham,UK: Packt Publishing ,2012.

MASUTANI, V. H. et al. Reconhecimento de mãos usando cores e formas aplicado no projeto de uma interface gestual. In: WORKSHOP DE VISÃO COMPUTACIONAL, 8., 2012, Goiânia. **Anais…** Goiânia - GO, 2012.

MSDN LIBRARY. **1.7 SDK and Developer Toolkit Known Issues**. 2013a. Disponível em:<http://msdn.microsoft.com/en-us/library/dn188692.aspx>. Acesso em: 9 set. 2013.

MSDN LIBRARY. **Kinect for windows SDK**. 2013b. Disponível em: <http://msdn.microsoft.com/en-us/library/hh855347.aspx>. Acesso em: 22 mar. 2013.

NIELS, R. **Dynamic time warping**: an intuitive way of handwriting recognition? Nijmegen, Netherlands: Radboud University Nijmegen, 2004.

PRADO NETO, E. X.; BRUNO, O. M. **Reconhecimento de gestos em Imagens de profundidade usando Kinect**. São Carlos: Universidade de São Paulo, 2012.

PULKIT, K.; ATSUO, Y. **Hand gesture recognition by using logical heuristics.** Technical Report 25, Japan Advanced Institute of Science and Technology, School of Information Science, 2012.

RAHEJA, J. L.; CHAUDHARY, A.; SINGAL, K. **Tracking of Fingertips and Centre of Palm using KINECT**. In proceedings of the 3 rd IEEE International Conference on Computational Intelligence, Modelling and Simulation, Malaysia, 20-22 Sep, 2011, pp. 248-252.

RYAN, D. J. **Finger and gesture recognition with Microsoft KINECT**. [S.l.]: University of Stavanger, Department of Eletctrical and Computer Engineering, 2011.

SANTOS, E. S. **Manipulação de objetos 3D em aplicações de realidade aumentada por meio da movimentação da mão.** Uberlândia: Universidade Federal de Uberlândia, 2011.

SAVI, R. **Utilização de Projeção multimídia em salas de aula: observação do uso em três escolas públicas.** Florianópolis: Universidade Federal de Santa Catarina, Pós-Graduação em Engenharia e Gestão do Conhecimento, 2009.

SENIN, P. **Dynamic time warping algorithm review**. Honolulu, Usa: University Of Hawaii At Manoa, 2008.

SILVEIRA, M. A. **Técnica de navegação em documentos utilizando Microsoft Kinect.** 2011. 36 f. (Trabalho de Conclusão de Curso) - Curso de Ciência da Computação, Universidade Federal do Rio Grande do Sul, Porto Alegre.

TOGORES, T. **Vitruvius**: um reconhecedor de gestos para o **Kinect.** 2011. 43 f. (Trabalho de Conclusão de Curso) - Curso de Ciência da Computação, Universidade de São Paulo, São Paulo.