ADRIANO OLIVEIRA BARBOSA

Estimativa de postura a partir de imagens de profundidade

Maceió

2012

ADRIANO OLIVEIRA BARBOSA

Estimativa de postura a partir de imagens de profundidade

Dissertação de Mestrado na área de concentração de Computação Gráfica submetida em 7 de Maio de 2012 à Banca Examinadora, designada pelo Colegiado do Programa de Pós-Graduação em Matemática da Universidade Federal de Alagoas, como parte dos requisitos necessários à obtenção do grau de mestre em Matemática.

Orientador: Prof. Dr. Dimas Martínez Morera

Maceió

2012

Estimativa de postura a partir de imagens de profundidade

Dissertação de Mestrado na área de concentração de Computação Gráfica submetida em 7 de Maio de 2012 à Banca Examinadora, designada pelo Colegiado do Programa de Pós-Graduação em Matemática da Universidade Federal de Alagoas, como parte dos requisitos necessários à obtenção do grau de mestre em Matemática.

Banca Examinadora:

Prof. Dr. Dimas Martínez Morera (UFAL)
Orientador

Prof. Dr. Thales Vieira (UFAL)
Coorientador

Prof. Dr. Marcelo Siqueira (UFRN)

Catalogação na fonte
Universidade Federal de Alagoas
Biblioteca Central
Divisão de Tratamento Técnico
Bibliotecária Responsável: Fabiana Camargo dos Santos

B238e Barbosa, Adriano Oliveira.
Estimativa de postura a partir de imagens de profundidade / Adriano Oliveira Barbosa. – 2012.
26 f. : il.

Orientador: Dimas Martínez Morera.
Co-orientador: Thales Vieira.
Dissertação (Mestrado em Matemática) – Universidade Federal de Alagoas. Instituto de Matemática. Maceió, 2012.

Bibliografia: f. 25-26.

1. Imagem de profundidade. 2. Estimativa de postura. 3. Kinect. 4. Algoritmo Iteractive Closest Point. 5. Mínimos quadrados. I. Título.

CDU: 519.712:004.932

Agradecimentos

Gostaria de agradecer aos meus pais, Cicero e Cicera, que sempre investiram e acreditaram em minha educação e em meus sonhos. Agradeço também a todos os meus familiares: irmã, primos, tios e avós pelo apoio e por acreditar em mim. À minha esposa, Karla Lima, pelo extremo apoio, dedicação, paciência e companherismo em todos os momentos. Agradeço aos amigos da vida *online*, Alan, Felipe, Francine, Giovanni e ao meu cunhado José Carlos pelos momentos de muita diversão.

Agradeço a todos os meus amigos e professores do Instituto de Matemática da UFAL, da graduação e pós-graduação, por me ajudarem em todo esse caminho. Em especial aos meus ex-orientadores, Adán Corcho e Adelailson Peixoto, e aos professores, André Contiero, Dimas Morera, Elisa Sena, Francisco Vieira Barros, José Carlos, Krerley Oliveira, Luana Contiero, Márcio Batista, Marcos Petrúcio e Thales Vieira pelo exemplo e amizade.

Um grande agradecimento aos amigos de laboratório, Ailton Felix, Augusto Ícaro, Diego Chicuta, Fabrício Lira, Lucas Lins, Leandro Botelho, Nayane Freitas, Tainá Ribeiro, que acompanharam cada momento. Em especial a Douglas Cedrim e Michel Alves, pelas discussões e ajuda com programação. Agradeço também aos amigos matemáticos puros, Carlos Gonçalves, José Ivan, José Lucyan e Márcio Cavalcante.

Aos meus orientadores Dimas Morera e Thales Vieira, pela imensa ajuda e tempo dedicado ao desenvolvimento do trabalho; bem como ao Professor Marcelo Siqueira por ter aceitado participar da banca, corrigir o trabalho e dar sugestões.

As agências de fomento FAPEAL e Capes e à UFAL pelo apoio financeiro.

Resumo

Neste trabalho, apresentaremos um método para estimar a postura de uma pessoa a partir de uma imagem de profundidade. Para isso, utilizamos uma abordagem similar à feita por Grest et al. em *Nonlinear body pose estimation from depth images*, onde se faz uso de uma técnica para encontrar correspondências entre a imagem e o modelo que representará a postura juntamente com uma técnica de otimização não linear. Em nosso trabalho utilizamos a adaptação do algoritmo ICP (Iterative Closest Point) unido ao método dos mínimos quadrados.

A construção do modelo é feita automaticamente utilizando o algoritmo de Dijkstra para calcular distância na nuvem de pontos. A aquisição das imagens foi feita utilizando o Kinect.

Palavras-chave: imagem de profundidade, estimativa de postura, Kinect, ICP, iterative closest point, mínimos quadrados.

Abstract

We present a method to estimate the posture of a person from depth images. We use an approach similar to that made by Grest et al. in *Nonlinear body pose estimation from depth images*, which use a technique to find correspondences between image and the model that will represent the posture with a nonlinear optimization technique. We use the adaptation of ICP (Iterative Closest Point) algorithm attached to the least squares method.

Model construction is done automatically using Dijkstra's algorithm to calculate distance in the point cloud. Image acquisition was performed using Kinect.

Keywords: depth image, pose estimation, Kinect, ICP, iterative closest point, least squares.

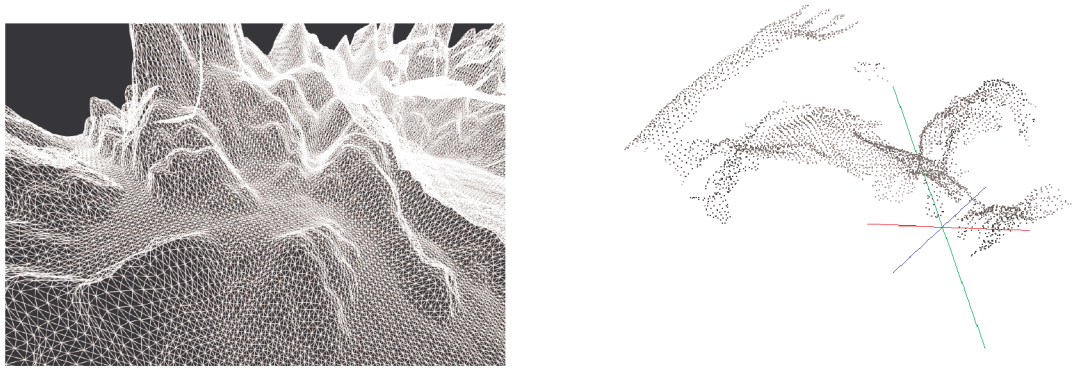
Sumário

1 Introdução

A comunicação está sempre presente em nosso cotidiano, seja pela fala ou por gestos, e é de fundamental importância. Pessoas que não podem falar, devido a alguma deficiência ou durante a prática de esportes como mergulho, têm sua própria linguagem e conseguem fazer com que outras pessoas que entendam essa linguagem consigam compreendê-las.

Apesar da evolução dos dispositivos que são responsáveis pela interação entre homem e máquina como, as telas de toque que estão presente nos celulares e tablets, a interface atual para interação com o computador ainda está atrelada ao uso de dispositivos como mouse e teclado.

Novas tecnologias vêm sendo desenvolvidas e podem melhorar essa interface de interação com o computador. Um exemplo de dispositivo capaz de atuar na melhoria dessa interação são as câmeras capazes de capturar imagens de profundidade (depth images). Essas câmeras funcionam como scanners 3D. Para cada pixel da imagem, elas conseguem estimar a que distância o objeto se encontra e assim criar um mapa de alturas correspondente àquela cena. Esse mapa de alturas pode ser visto como o gráfico de uma função $f : D \subset \mathbb{R}^2 \longrightarrow \mathbb{R}^+$, onde D representa o conjunto dos pixels da imagem e a função f associa a cada ponto de D um número real que corresponde a sua distância à câmera (figura 1.1). Esse mapa de alturas combinado com a imagem forma o que chamamos de imagem de profundidade, ou seja, $f : D \subset \mathbb{R}^2 \longrightarrow [0,1]^3 \times \mathbb{R}^+$, $f(i,j) = (r(i,j), g(i,j), b(i,j), d(i,j))$, onde r, g e b são as coordenadas RGB do pixel (i,j) e d é a distância do ponto do objeto que é representado pelo pixel (i,j) a câmera.



(a) Mapa de altura de um terreno

(b) Imagem de profundidade

Figura 1.1: Exemplos de mapas de altura

Videogames como o Xbox 360^{TM}, juntamente com o dispositivo Kinect^{TM} (Xbox 360 e Kinect são produtos da Microsoft), já exploram uma interação entre homem e máquina totalmente livre de contato com qualquer dispositivo e sem uso de qualquer acessório, utilizando somente gestos e comandos vocais (figura 1.2).

(a) (b)

Figura 1.2: (a) Kinect e Xbox 360 (b) Pessoas jogando o videogame

Um primeiro passo a ser dado para uso de dispositivos como as câmeras de profundidade é fazer com que o computador entenda o que a câmera está vendo. O objetivo é reconhecer partes do corpo e eventuais gestos e utilizá-los como instruções para operar o computador. Para isso, temos que extrair características que nos digam como o indivíduo em cena está se movimentando ou que pose está fazendo, tratar essa informação e ensinar o computador que operação deve ser executada.

Nosso trabalho tem como objetivo estimar a postura de uma pessoa diante do Kinect. O trabalho foi baseado no artigo devido a Grest et al., chamado *Nonlinear Body Pose Estimation from Depth Images* [5]. O texto está dividido em duas partes: a primeira parte trata dos trabalhos relacionados e um rápido apanhado sobre as pesquisas já desenvolvidas na área, além de falar um pouco sobre o funcionamento do Kinect. Na segunda parte, apresentamos o trabalho realizado; aqui, utilizamos a imagem adquirida a partir do Kinect para estimar a postura de uma pessoa em frente à câmera. Para isso, utilizamos uma técnica chamada ICP (Iterative Closest Point) [2] que é responsável por encontrar correspondências entre a imagem obtida e o modelo que utilizamos para representar a postura. A partir dessas correspondências, efetuamos uma otimização que tem como objetivo minimizar a distância entre os pontos do modelo e seus correspondentes. No último capítulo, apresentamos as conclusões, aplicações e trabalhos futuros.

2 Preliminares

Neste capítulo, apresentamos o Kinect e alguns trabalhos relacionados à estimativa de postura.

2.1 O Kinect

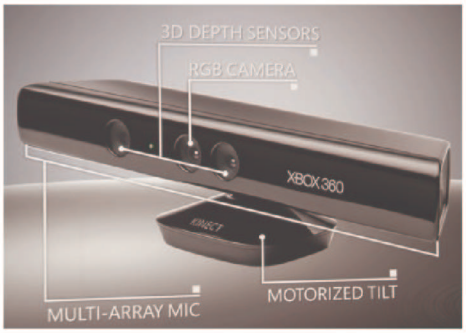
Em meados de 2010, durante a E3 (Eletronic Entertainment Expo), feira internacional dedicada a jogos eletrônicos, a Microsoft anunciou seu novo produto que até então era conhecido como Projeto Natal, o Kinect. Ele foi desenvolvido para funcionar como periférico para o console Xbox 360 e faz com que o usuário consiga jogar vários jogos sem o contato com nenhum tipo de dispositivo, utilizando apenas gestos e comandos vocais.

O Kinect é conhecido como "sensor de movimento", mas na verdade ele é um dispositivo de captura de imagem capaz de capturar informação de profundidade. Uma imagem de profundidade (depth image) é uma imagem onde cada pixel possui, além da sua definição de cor, uma informação de distância que indica o quão distante aquele pixel se encontra de um ponto fixo que determina a origem do sistema (a câmera, por exemplo). Dessa forma, o que o Kinect faz é gerar um filme com informação de profundidade e o console se encarrega de tratar essa informação, identificar e executar a tarefa correspondente a cada gesto ou comando de voz.

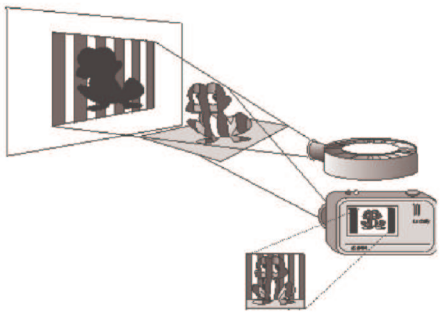
O Kinect é composto por uma câmera RGB, um sensor de profundidade e uma cadeia de microfones (figura 2.1(a)). O sensor de profundidade consiste de um projetor de luz infravermelha e um sensor CMOS. O projetor emite luz infravermelha em um padrão de luz estruturada que é reconhecido pelo sensor CMOS e a partir de um processo de triangulação o Kinect consegue determinar a que distância o objeto se encontra (figura 2.1(b)). Essa técnica é conhecida como estereoscopia e já é bem difundida [20].

Através de engenharia reversa [9, 10, 13, 14], descobriu-se que o Kinect é capaz de capturar vídeos a 30Hz usando uma resolução de 640x480 pixels com 8-bits de cor, um filtro de cor de Bayer e 11-bits para profundidade, gerando 2.048 níveis de sensibilidade. Seu sensor de profundidade consegue trabalhar com objetos situados a distâncias no intervalo de 0,7-6m. por sua vez, a cadeia de microfones composta por 4 microfones opera capturando áudio a 16kHz e 16-bits em cada canal.

Diante dessas características e seu baixo custo, o equipamento foi rapidamente adotado por pesquisadores em todo o mundo como alternativa aos scanners 3D e é amplamente explorado como ferramenta para pesquisa em visão computacional, registro e reconstrução de objetos 3D [15, 10, 13, 14, 9].



(a) Slide da Microsoft na E3



(b) Luz estruturada

(c) Imagem infravermelho do Kinect

Figura 2.1: Partes do Kinect e luz estruturada

2.2 Trabalhos relacionados

Estimar a postura do corpo humano a partir de uma imagem não é uma tarefa fácil para o computador. A combinação de várias técnicas de Computação Gráfica e Visão Computacional é necessária para se alcançar o objetivo. Daí a dificuldade inerente a esse tipo de problema.

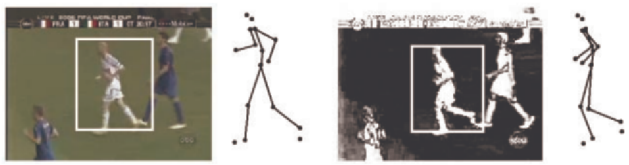
Duas abordagens são utilizadas em Computação Gráfica: uma a partir de imagens adquiridas de qualquer câmera convencional e outra que utiliza informação de profundidade (geometria da cena) adquirida a partir de câmeras especiais como scanners 3D ou o próprio Kinect.

Além da classificação acima podemos fazer mais uma classificação que se aplica a ambos os grupos descritos acima: trabalhos que utilizam conjuntos de treino, ou seja, o computador é previamente ensinado a reconhecer alguns padrões preestabelecidos e os que não utilizam conjuntos de treino, onde todo processamento é feito em tempo de execução, sem nenhum conhecimento prévio.

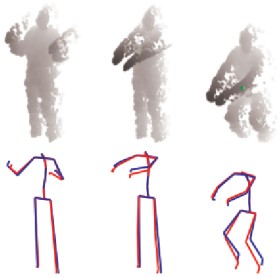
Jaeggli et al. [7] (figura 2.2(a)) extraem silhuetas de imagens comuns e as usam como entrada para um método de otimização estatística que estima a postura e utiliza movimento de corrida e caminhada como conjunto de treino. Schwartz et al. [16] fazem uso de dados de treino, mas utilizam câmera *time-of-flight* (ToF) [1] na aquisição de imagens (figura 2.2(b)). Nesse trabalho, os autores fazem a estimativa da postura além de utilizar um conjunto de ações previamente treinadas para classificar qual ação o usuário está executando. A abordagem de Sun et. al [19] utiliza várias câmeras comuns apontado para o mesmo objeto e, utilizando-se

[1] Uma câmera time-of-flight é um sistema de câmera capaz de calcular distância baseado na velocidade da luz que ele emite, medindo o "tempo de vôo" do sinal de luz entre a câmera e o objeto.

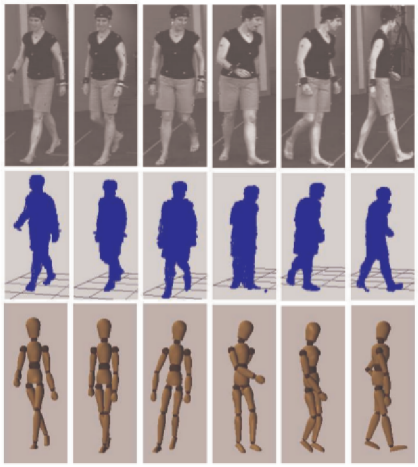
da silhueta dessas imagens, geram um conjunto de voxels que é utilizado como informação tridimensional (figura 2.2(c)). A partir daí eles propõem um método de regressão chamado *Multi-Variate Relevance Vector Machine* que é responsável por aprender e mapear a informação dos voxels num modelo mais simples que determina a postura.



(a) Jaeggli et al. [7]



(b) Schwarz et al. [16]



(c) Sun et al. [19]

Figura 2.2: Trabalhos relacionados

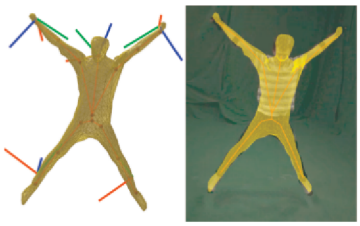
Trabalhos que não utilizam um conhecimento prévio são mais dependentes da extração de características. Como a aparência do corpo humano é altamente afetada por iluminação e mudanças de postura, as técnicas tentam ser independentes desses fatores. Para lidar com esse problema, Kehl e van Gool [8] (figura 2.3(a)) usam múltiplos pontos de vista vindos de várias câmeras para gerar uma reconstrução volumétrica em 3D da postura estimada.

Uma abordagem muito comum para tratar do problema de detecção de movimento e estimativa de postura é a utilização de marcadores para facilitar e agilizar a etapa de extração de características. Pons-Moll et al. [12] (figura 2.3(b)) utilizam marcadores no corpo e um conjunto de orientação local, vindo dos sensores associados a estes marcadores, assim como as imagens vindas das câmeras para criar um rastreador (tracker) híbrido que é capaz de estimar a postura do corpo humano.

Recentemente, vários trabalhos vêm explorando as câmeras ToF na análise de movimento e postura do corpo humano. Esse tipo de câmera tende a facilitar o trabalho porque, além da imagem comum gerada por qualquer câmera ordinária, ela é capaz de estimar a distância a que o objeto se encontra. Essa informação geométrica da cena é muito relevante e vem se revelando

uma grande ferramenta no estudo de detecção de movimento e de postura. Soutscheck et al. [18] descrevem uma metodologia para reconhecimento de gestos da mão que é aplicada em imagens médicas. Os autores ainda efetuaram uma pesquisa de satisfação. A aplicação foi avaliada como muito boa em se tratando de intuição, adaptatividade e usabilidade da ferramenta. Já Holte et al. [6] (figura 2.3(c)) tratam a detecção de gestos: 11 gestos distintos, previamente treinados, são reconhecidos pelo computador com uma precisão de cerca de 85% quando o usuário não indica quando o gesto foi iniciado e terminado e em torno de 92% de precisão se o gesto é feito com o indicador de início e término. Zhu et al. [21] (figura 2.3(d)) utilizam uma câmera ToF para estimar a postura da parte superior do corpo humano com ajuda de probabilidades e coerência temporal.

(a) Kehl et al. [8]



(b) Pons-Moll et al. [12]

(c) Holte et al. [6]

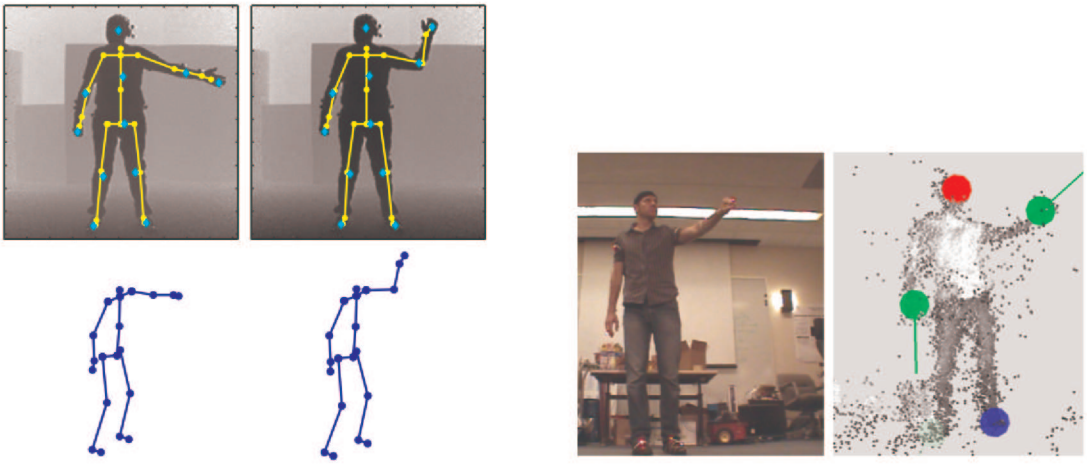
(d) Zhu et al. [21]

Figura 2.3: Trabalhos relacionados

Com pouco mais de um ano, o Kinect já despertou um gigantesco interesse na comunidade acadêmica por conta de sua capacidade. Schwarz et al. [15] (figura 2.4(a)) fazem uma abordagem bem parecida com a proposta por Plagemann et al. [11] (figura 2.4(b)) para estimativa da postura. Eles usam tanto uma câmera ToF quanto o Kinect, e fazem uso do mapa de alturas fornecido pela câmera para gerar um grafo que permite calcular distâncias na nuvem de pontos. A partir do cálculo de distâncias, é possível identificar alguns pontos com as juntas do corpo humano (cotovelo, joelho, ombro e etc) e assim construir o esqueleto.

Recentemente foi publicado um trabalho, desenvolvido pelo Microsoft Research Cambridge & Xbox Incubation, que apresenta como é feita a estimativa de postura no Xbox [17]. Nele, Shotton et al. utilizam uma árvore de decisão que julga muito precisamente quais pontos da nuvem representam as juntas, mas para isso é necessário um conjunto de treino gerado com muito esforço computacional.

Apesar de existir vários trabalhos que conseguem estimar postura de maneira satisfatória, o problema de estimar postura ainda não está totalmente resolvido. Todos os trabalhos possuem restrições, como o clássico problema de oclusão do usuário em cena. Sempre que outro objeto



(a) Schwarz et al. [15]

(b) Plagemann et al. [11]

Figura 2.4: Trabalhos relacionados

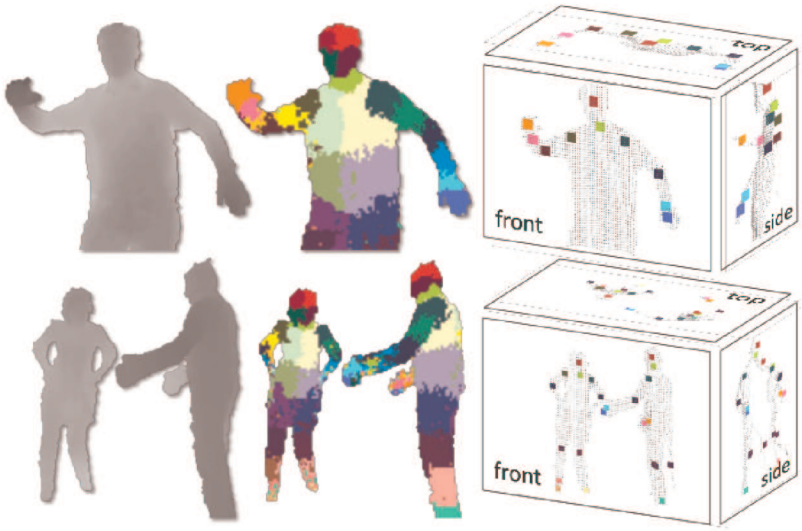


Figura 2.5: Shotton et al. [17]

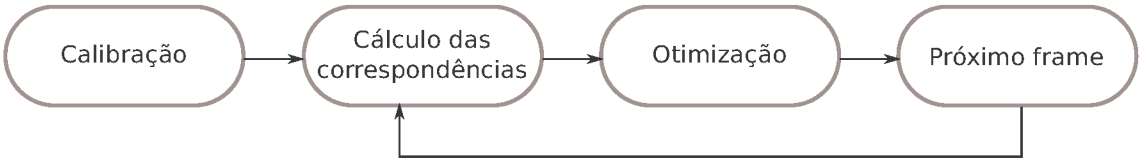
em cena ou o próprio corpo do usuário oculta parte ou totalmente o alvo que deve ter a postura estimada, os algoritmos atuais perdem a precisão [17, 11] e até fazem representações ruins da postura [5].

3 Estimando a postura

Nosso objetivo é estimar a postura de uma determinada pessoa numa cena a partir de dados de uma imagem de profundidade. Para a aquisição dos dados, utilizamos o Kinect junto com a biblioteca OpenNI [10].

3.1 Visão geral do método

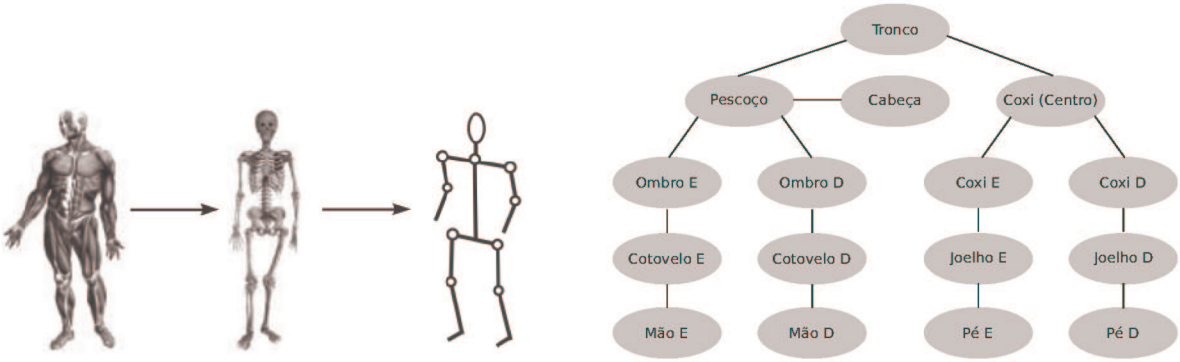
O método que apresentaremos segue o seguinte roteiro: inicialmente, efetuamos uma calibração automática utilizando somente a imagem de profundidade gerada pela câmera, que é responsável pela construção do modelo que representará o usuário em cena; após a calibração, fazemos a busca dos pontos da imagem correspondentes aos pontos do modelo e de posse dessas correspondências efetuamos um passo de otimização que tem como objetivo calcular a transformação que melhor ajusta o modelo ao formato da imagem de profundidade.



3.2 Modelo de esqueleto utilizado

As articulações que conectam os ossos possuem vários pontos de contato e inúmeros graus de liberdade tornando uma representação completa extremamente difícil. Entretanto, várias aplicações necessitam apenas de uma representação simplificada. Em nosso trabalho utilizamos um modelo, derivado da hierarquia de Joe-stick [4], que possui um conjunto de articulações do corpo humano suficiente para uma boa representação dos movimentos comumente realizados pelas pessoas. Esta hierarquia é mostrada na figura 3.1(b).

A partir do modelo hierárquico podemos construir diversos modelos. É possível utilizar modelos bem simples ou modelos com objetos mais complexos (figura 3.2(a)). Como nosso objetivo é puramente estimar a postura, utilizaremos um modelo simples em nossas representações, mas que possui todas as articulações necessárias para a aplicação (característica devida à hierarquia de Joe-stick). Nosso modelo é composto por segmentos de reta conectados de modo que essas ligações possuam os graus de liberdade necessários para representar o movimento do corpo humano (figuras 3.2(b), 3.4(b)).



(a) (b)

Figura 3.1: Simplificação do corpo humano em (a) e Modelo hierárquico em (b)

(a) Possíveis modelos do joe-stick (b) Nosso modelo

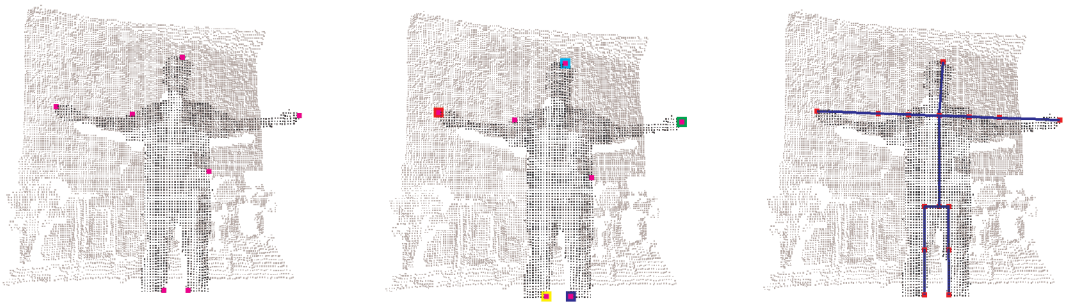
Figura 3.2: Modelos derivados de joe-stick

3.3 Calibração do modelo

As dimensões do esqueleto mudam de um usuário para outro. Por este motivo, precisamos defiir (calibrar) nosso modelo antes de todos os cálculos de correspondências e otimização serem realizados. Como o modelo é sensível ao usuário e à sua posição diante da câmera, é necessário uma calibração automática que define o modelo a partir da nuvem do usuário.

No processo de calibração, nós utilizamos uma abordagem baseada no trabalho de Grest et al. [11] que usa o algoritmo de Dijkstra [3] para calcular distância na nuvem de pontos e encontrar pontos de interesse. Lembre que nossa nuvem de pontos possui estrutura de imagem, e portanto a estrutura de vizinhança de um grafo. O que pretendemos com isso é aproveitar a estrutura isométrica que nosso corpo possui, apesar de ser maleável, a distância (medida sob nosso corpo) entre a ponta de nossos dedos e nosso ombro, por exemplo, é invariante diante dos movimentos possíveis ao corpo humano. Os pontos de interesse são calculados da seguinte forma: seja s um ponto da nuvem que pertence ao conjunto de pontos $\mathscr{P}$ da pessoa (para encontrar esse primeiro ponto em $\mathscr{P}$, ajustamos a cena de modo que a pessoa seja o objeto mais próximo da câmera e tomamos s como o ponto mais próximo da câmera), utilizamos s como semente para o algoritmo. A partir de s o algoritmo calcula a distância a todos os outros pontos de $\mathscr{P}$ e toma o ponto mais distante como o primeiro ponto de interesse. Como estamos interessados apenas nos pontos que pertencem a $\mathscr{P}$, foi usado um critério que julga se um vizinho (nossa imagem de profundidade é uma imagem comum com informação de distância,

então podemos usar a relação de vizinhança da própria imagem para encontrar os vizinhos de um ponto qualquer) v_p de um ponto p de $\mathscr{P}$ ainda é um ponto de $\mathscr{P}$, que simplesmente analisa a distância entre esses dois pontos, ou seja, $v_p \in \mathscr{P}$ se $d(v_p, p) < \varepsilon$, onde ε é um *threshold* que foi obtido empiricamente. Utilizamos então esse ponto de interesse e recalculamos as distâncias para encontrar um segundo ponto de interesse da mesma forma que o primeiro. Repetimos o procedimento e encontramos os outros pontos de interesse, sempre utilizando o conjunto de pontos de interesse acumulado na iteração anterior como semente para o algoritmo de Dijkstra. Efetuar o procedimento sete vezes é suficiente para encontrar os cinco pontos que realmente nos interessam que são as duas mãos, os dois pés e a cabeça [11] (ver figuras 3.3(a) e 3.3(b)). Agora basta identificar quais são esses pontos. Para isso exigimos que na etapa de calibração o usuário fique na posição de T (com as pernas levemente separadas e com os braços abertos), e com isso tomamos os pontos com maior e menor coordenada x como sendo as mãos, o ponto com maior coordenada y como sendo a cabeça e os dois pontos com menor coordenada y como sendo os pés. O ponto do pescoço é o ponto médio entre as duas mãos e o ponto central da cintura como (x_{pescoo}, y_c), onde y_c é $\frac{3}{4}$ da altura de $\mathscr{P}$ ($|y_{cabea} - y_p|$). Cada cotovelo é o ponto médio entre a mão respectiva e o pescoço, já o ombro é o ponto médio entre o pescoço e o cotovelo. Os pontos laterais da cintura possuem a mesma coordenada x do pé que está no mesmo lado e a coordenada y do ponto central da cintura. Por fim, os joelhos são definidos como o ponto médio entre os laterais da cintura e os pés respectivos (figura 3.3(c)).



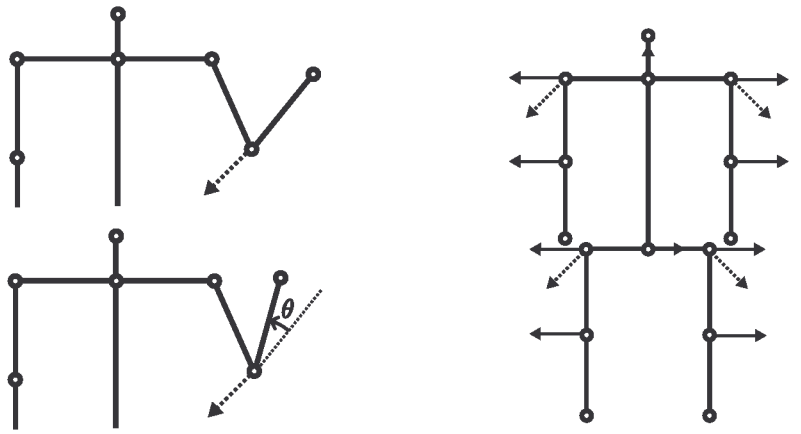
(a) Pontos de interesse (b) Pontos identificados (c) Modelo construído

Figura 3.3: Etapas da construção do modelo

3.4 Os movimentos do corpo humano

Nosso corpo pode ser visto como um conjunto de partes rígidas ligadas por juntas articuladas. Por exemplo, podemos dividir nosso braço em duas partes, braço e ante-braço, e essas partes são ligadas por nossas articulações, ombro e cotovelo. Por simplicidade estamos contando a mão como parte do antebraço. A partir daí podemos modelar o movimento de um aceno como uma rotação do antebraço em torno de um eixo fixado no cotovelo (figura 3.4(a)). Em nossos desenhos, os eixos tracejados apontam pra fora do papel.

Nosso modelo possui dois tipos de juntas: articulações e extremidades. A maioria das articulações têm apenas um grau de liberdade e, portanto, possuem um eixo de rotação. As articulações correspondentes aos ombros e à parte inicial das pernas têm dois graus de liberdade e como tal possuem dois eixos de rotação cada. Na figura 3.4(b) estão representados todas as juntas e seus respectivos eixos de rotação. Dessa forma, tudo que precisamos fazer para



(a) Movimento de aceno (b) Eixos de rotação do nosso modelo

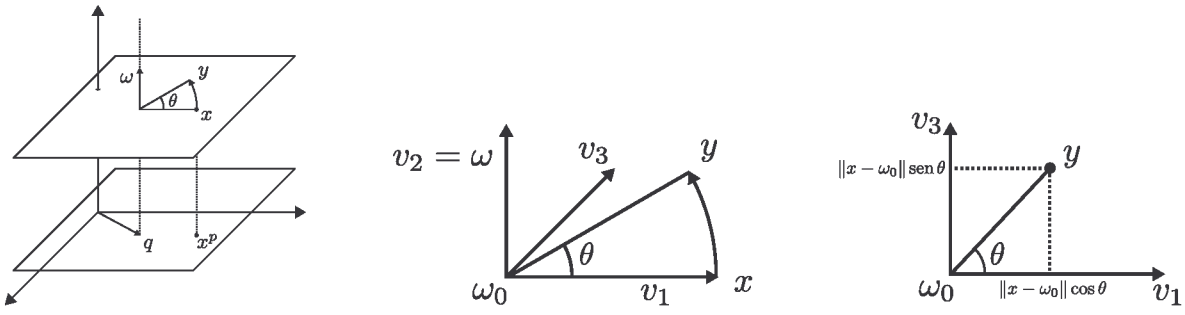
Figura 3.4: O movimento é definido pelos eixos de rotação

nosso modelo se movimentar é aplicar rotações de ângulos adequados em cada um desses eixos fazendo com que as partes do corpo dependentes daquela junta também realizem a rotação indicada. Veja, por exemplo, o movimento de aceno mostrado na figura 3.4(a).

3.5 Movimento e rotação em torno de um eixo arbitrário

Movimentos do corpo humano, como acenar ou abrir e fechar os braços, podem ser expressos através de rotações em torno de certos eixos [5]. Para movimentar nosso modelo, basta saber como efetuar rotações em torno de um eixo arbitrário.

Seja x uma junta do nosso modelo. Queremos que essa junta seja rotacionada de θ radianos em torno de um eixo arbritário, representado por seu vetor diretor unitário ω, veja figura 3.5(a). Chamaremos de y o resultado da rotação aplicada a x.



(a) (b) (c)

Figura 3.5: (a) Visão geral. (b) Zoom nas proximidades de x. (c) y na base $\{v_1, v_2, v_3\}$

Deduziremos agora uma forma para efetuar uma rotação em torno de eixos arbitrários. Sejam q o ponto do eixo ω que está mais próximo da origem, ω_0 o ponto de interseção entre o eixo ω e o plano que contém x e é ortogonal a ω e v_1, v_2 e v_3 definidos da seguinte forma (figura 3.5):

$$\begin{aligned} v_1 &= \frac{x-\omega_0}{\|x-\omega_0\|} \\ v_2 &= \omega \\ v_3 &= \omega \times v_1 \end{aligned} \tag{3.1}$$

observe que v_1, v_2 e v_3 formam uma base ortonormal, assim podemos escrever y como

$$\begin{aligned} y-\omega_0 &= \|x-\omega_0\|\cos\theta\ v_1 + \|x-\omega_0\|\sin\theta\ v_3 \\ &= \|x-\omega_0\|\cos\theta\left(\frac{x-\omega_0}{\|x-\omega_0\|}\right) + \|x-\omega_0\|\sin\theta\left(\omega\times\left(\frac{x-\omega_0}{\|x-\omega_0\|}\right)\right) \\ &= \cos\theta\,(x-\omega_0)+\sin\theta\,(\omega\times(x-\omega_0)) \\ y &= \omega_0+\cos\theta\,(x-\omega_0)+\sin\theta\,(\omega\times(x-\omega_0)) \end{aligned} \tag{3.2}$$

Note que $\langle x,\omega\rangle\,\omega$ é a projeção de x sobre ω, dessa forma, $\omega_0 = q+\langle x,\omega\rangle\,\omega$. Logo

$$\begin{aligned} y &= q+\langle x,\omega\rangle\,\omega+\cos\theta\,(x-\omega_0)+\sin\theta\,(\omega\times(x-\omega_0)) \\ &= x+(q-x)+\langle x,\omega\rangle\,\omega+\cos\theta\,(x-\omega_0)+\sin\theta\,(\omega\times(x-\omega_0)) \end{aligned}$$

onde $q-x+\langle x,\omega\rangle\,\omega = \omega_0 - x$, assim

$$\begin{aligned} y &= x+(1-\cos\theta)(\omega_0-x)+\sin\theta\,(\omega\times(x-\omega_0)) \\ &= x+(1-\cos\theta)(q-x^p)+\sin\theta\,(\omega\times(x-q)) \end{aligned}$$

onde $x^p = x-\langle x,\omega\rangle\,\omega$ e $\omega\times(x-q-\langle x,\omega\rangle\,\omega) = \omega\times(x-q)$. Portanto

$$\begin{aligned} y &= x+\sin\theta(\omega\times(x-q))+(q-x^p)-\cos\theta(q-x^p) \\ y &= R_{\omega,q}(\theta)\cdot x \end{aligned} \tag{3.3}$$

onde $x^p = x-\langle x,\omega\rangle\,\omega$ é a projeção de x no subespaço ortogonal a ω (plano que corta a origem cujo vetor normal é ω) e $\langle\cdot,\cdot\rangle$ denota o produto interno de $\mathbb{R}^3$.

De posse dessas rotações podemos definir a função f que aplica as rotações a cada ponto do nosso modelo. Se o modelo possuir j eixos de rotação nas articulações, precisaremos de $p = j+6$ parâmetros para aplicar todas as transformações necessárias, sendo três $(\theta_1,\theta_2,\theta_3)$ para translação que será aplicada a todas as juntas, três $(\theta_4,\theta_5,\theta_6)$ para as rotações globais (em torno de cada eixo coordenado) que também serão aplicadas a todas as juntas e j $(\theta_7,...,\theta_p)$ devidos aos eixos das articulações, aplicados aos respectivos dependentes de cada junta. Dessa forma, temos $f:\mathbb{R}^p\times\mathbb{R}^3\to\mathbb{R}^3$ definida por

$$f(\theta,x) = (\theta_1,\theta_2,\theta_3)+R_x(\theta_4)\cdot R_y(\theta_5)\cdot R_z(\theta_6)\cdot R_{\omega_7,q_7}(\theta_7)\cdot \ ... \ \cdot R_{\omega_r,q_r}(\theta_r)\cdot x \quad (3.4)$$

onde $r < p$, $\theta = (\theta_1,...,\theta_p)$, $(\theta_1,\theta_2,\theta_3)$ é a translação, R_x, R_y e R_z são rotações em torno dos eixos coordenados e $R_{\omega_i,q_i}(\theta_i)$, $i \in \{7,...,p\}$ são rotações de θ_i radianos em torno dos eixos associados a cada junta. As rotações $R_{\omega_i,q_i}(\theta_i)$ citadas acima são apenas as rotações cujo eixo está associado às juntas que estão ligadas ao ponto x. Por exemplo, se x é o ponto que representa a mão, a rotação em torno do eixo que está ligado ao joelho não deve afetar x. Essas rotações são as que estão acima do ponto x na árvore de hierarquia (ver 3.1(b)).

Além disso,

$$\begin{aligned}
\frac{\partial f}{\partial \theta_i}(\theta,x) &= e_i, \text{ se } i \in \{1,2,3\} \quad (3.5)\\
\frac{\partial f}{\partial \theta_i}(\theta,x) &= R_x(\theta_4)\cdot ... \cdot R_{\omega_{i-1},q_{i-1}}(\theta_{i-1})\cdot R'_{\omega_i,q_i}(\theta_i)\cdot R_{\omega_{i+1},q_{i+1}}(\theta_{i+1})\cdot ... \cdot R_{\omega_p,q_p}(\theta_p)\cdot x,\\
&\quad \text{se } i \in \{4,...,p\}
\end{aligned}$$

onde

$$R'_{\omega,q}(\theta) = \frac{\partial R_{\omega,q}(\theta)}{\partial \theta} = \cos\theta(\omega \times (x-q)) + \sin\theta(q - x^p)$$

$\frac{\partial f}{\partial \theta_i}$ é o vetor que indica para onde x se move quando efetuarmos o conjunto de rotações θ.

Em particular, se $\theta = 0$, a derivada se reduz a:

$$\frac{\partial f}{\partial \theta}(\theta,x) = \left.\frac{\partial R_{\omega,q}(\theta)}{\partial \theta}\right|_{\theta=0} = \omega \times (x-q) \quad (3.6)$$

3.6 ICP e Otimização

Para encontrar os parâmetros que melhor se ajustam ao modelo em relação à nuvem de pontos, usamos a técnica conhecida como ICP [2] seguido de uma otimização não linear. Nesta seção, descrevemos ambas as técnicas.

Dados os conjuntos $X = (x_1,x_2,...,x_n)$ de pontos (juntas) do modelo e $Y = (y_1,y_2,...,y_n)$ de pontos da nuvem (nossos pontos de referência), queremos encontrar os parâmetros θ_i, $i \in \{1,...,p\}$ que melhor ajustam o conjunto X ao conjunto Y, onde cada ponto será levado de X para Y através da função $f(\theta,x)$ descrita acima.

Primeiramente, temos que encontrar as correspondências entre os pontos do nosso modelo e os pontos da nuvem. Para isso utilizaremos a técnica ICP. Esta técnica consiste em, para cada ponto do modelo, encontrar o ponto da nuvem que está mais próximo. Para isso exigimos que o modelo esteja numa postura próxima da postura da nuvem, o que pode ser conseguido utilizando uma calibração inicial. De posse dessas correspondências efetuamos o cálculo dos θ's que

melhor ajustam o modelo à nuvem e aplicamos os respectivos θ's ao modelo resultando num modelo modificado e mais próximo da nuvem. Repetimos esse processo (por isso, iterativo) até que a diferença entre o θ encontrado na próxima iteração e o θ atual seja aceitavelmente pequenos ou até atingirmos uma quantidade máxima de iterações.

Agora que já temos todo o pipeline descrito, basta saber como calcular os parâmetros θ. Com efeito, nosso problema é encontrar os valores de $\theta_1, ..., \theta_p$ que minimizam a diferença $\|Y - F(\theta)\|$, onde a função $F : \mathbb{R}^p \to \mathbb{R}^{3n}$ é dada por $F(\theta) = (f(\theta, x_1), ..., f(\theta, x_n))$. Como já sabemos que nosso modelo está numa postura próxima da postura real, podemos assumir na iteração t que:

$$F(\theta^t + \Delta\theta) \approx F(\theta^t) + \left.\frac{\partial F}{\partial \theta}\right|_{\theta^t} \cdot \Delta\theta \tag{3.7}$$

onde a derivada $\left.\frac{\partial F}{\partial \theta}\right|_{\theta^t}$ é a Jacobiana da função F.

Seja $\varepsilon^t = Y - F(\theta^t)$ o erro do nosso problema de minimização na iteração t, então nosso problema pode ser reformulado da seguinte maneira:

$$min_{\Delta\theta} \ \left\|\varepsilon^t - J\Delta\theta\right\|, \tag{3.8}$$

ou seja, $\varepsilon^t - J\Delta\theta \approx 0 \Rightarrow \varepsilon^t \approx J\Delta\theta$. Estamos tratando aqui de um sistema sobredeterminado, ou seja, possui mais equações que incógnitas, pois como J tem tamanho $3n \times p$ e nosso modelo tem 15 pontos e 14 eixos de rotação, temos $n = 15$ e $p = 14 + 6 = 20$. Para resolver o sistema, utilizamos a técnica de mínimos quadrados, que equivale a multiplicar ambos os lados da equação por J^T (transposta de J) transformando nosso sistema em um sistema de equações normais $p \times p$:

$$J^T J\Delta\theta = J^T \varepsilon^t \tag{3.9}$$

para isso, utilizamos a biblioteca Lapack++ [1]. Observe que não temos como garantir que $J^T J$ será não singular. Quando a matriz tem determinante nulo, fazemos uma pequena perturbação na diagonal da matriz para que deixe de ser singular.

Note que em cada passo da iteração podemos assumir que os θ's iniciais são todos iguais a zero, pois o novo sistema não depende da iteração anterior. Dessa forma, simplificamos bastante os cálculos (veja equação 3.6). Além disso, vale salientar que, uma vez aplicados os parâmetros encontrados, precisamos recalcular todos os eixos de rotação (ω's) e os pontos de referência (q's) das juntas que mudaram de posição.

3.7 Resultados

Em nossos testes, conseguimos que a postura fosse estimada em tempo real rodando em um computador com processador Intel Dual-Core E2180 @ 2 Ghz e 2 Gb de ram.

Experimentos

Nossos experimentos foram feitos com uma sequência de imagens real, onde o usuário inicia na posição de calibração e executa movimentos de abrir, fechar e levantar os braços. Os testes mostraram que o algoritmo consegue estimar a postura do usuário com 20 graus de liberdade em tempo real.

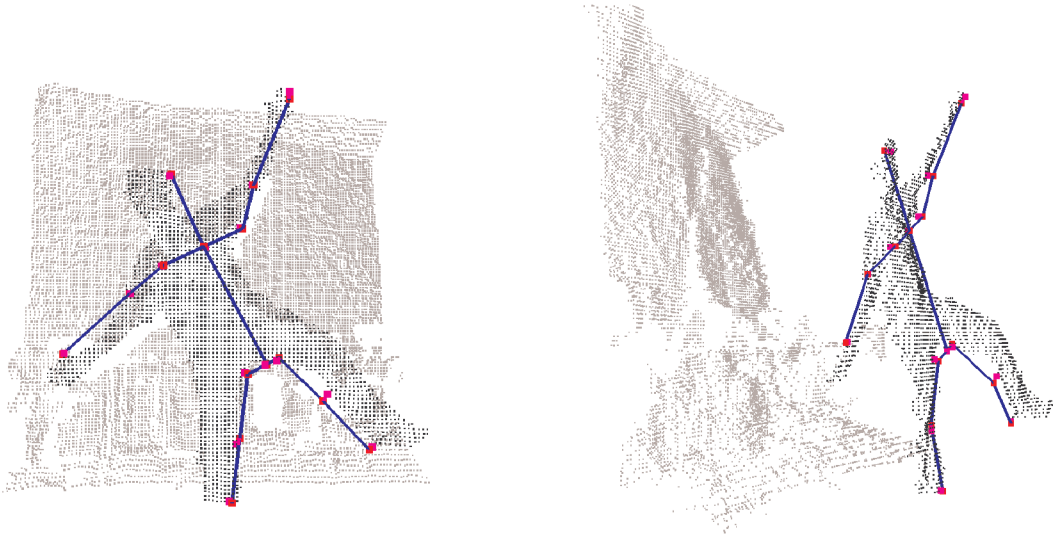


Figura 3.6: Postura estimada

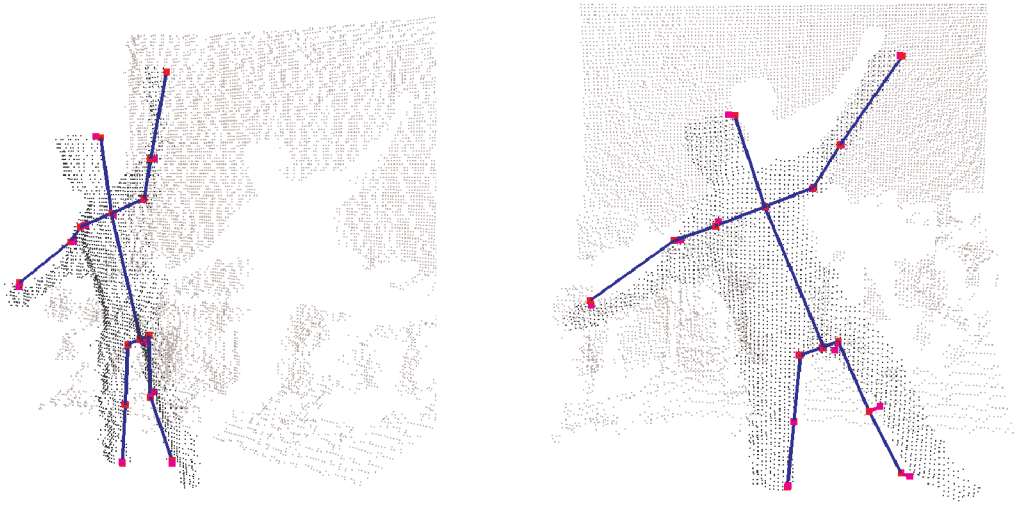
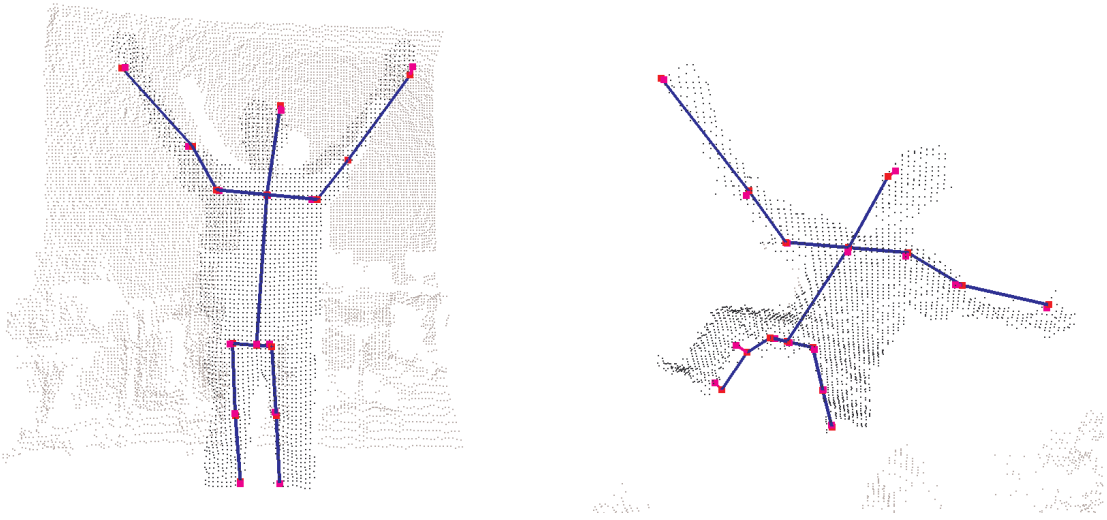


Figura 3.7: Mesma postura vista de ângulos diferentes

Posturas mais complexas também foram testadas, Figura 3.7 e Figura 3.8(b). A próxima seção mostra o que pode acontecer com o modelo quando tentamos posturas muito complexas.



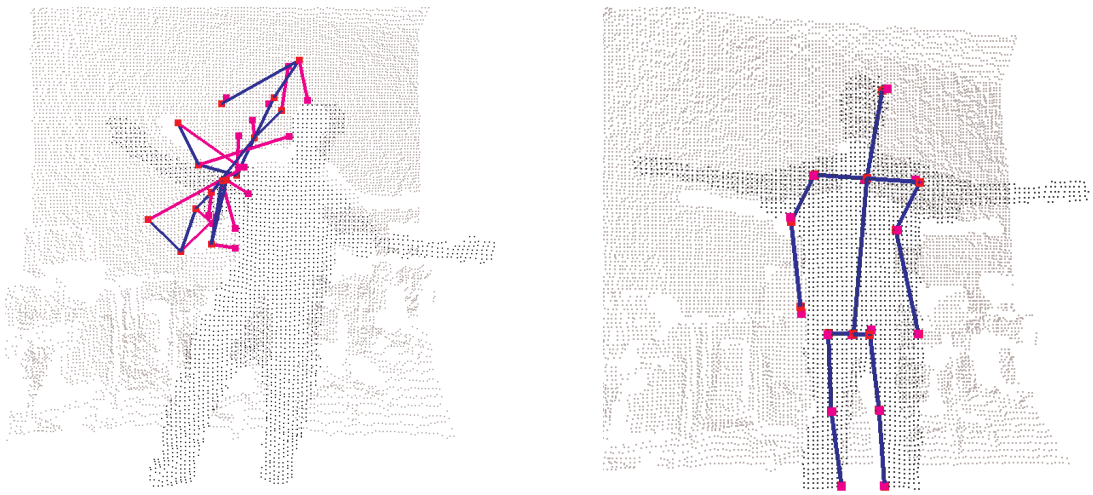
(a) Postura sguinte a pose de calibração

(b) Ponto de vista por trás do usuário

Figura 3.8: Postura estimada

Limitações

O algoritmo é capaz de calcular a postura de uma única pessoa por vez e o usuário deve ser o objeto da cena mais próximo da câmera para que a calibração automática funcione. O método



(a) Modelo totalmente perdido devido a resultado ruim na otimização

(b) Após oclusão parcial, os braços foram perdidos

Figura 3.9: Limitações do método

não prevê uma recalibração, caso seja necessário. Por exemplo, quando acontece a oclusão total ou parcial do usuário, ou auto-oclusão, seria necessário fazer uma recalibração, o que não está completamente coberto pelo algoritmo.

A Figura 3.9(b) mostra um esqueleto que foi calculado após o usuário ter posto os braços por trás do corpo, enquanto que a Figura 3.9(a) mostra o que pode acontecer ao modelo quando

o usuário é totalmente ocluso.

Tais limitações são devidas ao cálculo das correspondências pelo ICP. Uma vez encontrada uma correspondência ruim, a otmização retornará parametros ruim para as rotações.

4 Conclusões

Neste trabalho apresentamos uma técnica para estimativa de postura a partir de imagens de profundidade. Isso é feito encontrando correspondências entre a imagem e o modelo que representará a postura e a partir daí aplicando uma otimização que visa minimizar a distância entre os pontos do modelo e seus correspondentes na imagem. Por possuir um pipeline simples, a técnica pode ser adaptada para utilizar outros métodos nas etapas de cálculo de correspondências e otimização além dos que foram apresentados neste trabalho.

Também apresentamos um método para calibração automática que usa distância na nuvem para identificar pontos de interesse que são usados na definição do modelo.

Ainda há pesquisa a ser feita com relação a estimativa de postura. Apesar de existir métodos bastante robustos para estimativa de postura [17], casos de oclusão parcial ou total, ou quando o usuário gira em torno do próprio eixo alterando a orientação ainda causam confusão nos métodos existentes.

Referências Bibliográficas

[1] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK Users' Guide*. Society for Industrial and Applied Mathematics, Philadelphia, PA, third edition, 1999.

[2] P.J. Besl and N.D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:239–256, 1992.

[3] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. The MIT Press, 2 edition, 2001.

[4] J. Gomes and L. Velho. *Fundamentos da Computacao Grafica*. IMPA, 2003.

[5] D. Grest, J. Woetzel, and R. Koch. Nonlinear body pose estimation from depth images. *Pattern Recognition*, pages 285–292, 2005.

[6] M.B. Holte, T.B. Moeslund, and P. Fihl. Fusion of range and intensity iformation for view invariant gesture recognition. *Computer Vision and Pattern Recognition Workshops*, 2008.

[7] T. Jaeggli, E. Koller-Meier, and L. V. Gool. Learning generative models for multi-activity body pose estimation. *Int. J. Comput. Vis.*, 83(2):121–134, 2009.

[8] R. Kehl and L. Gool. Markerless tracking of complex human motions from multiple views. *Comput. Vis. Image Underst.*, 104(2):190–209, 2006.

[9] OpenKinect. Openkinect, 2012. [Online; acessado 31-Fevereiro-2012].

[10] OpenNI organization. *OpenNI User Guide*, November 2010. Last viewed 19-01-2011 11:32.

[11] C. Plagemann, V. Ganapathi, D. Koller, and S. Thrun. Real-time identification and localization of body parts from depth images. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 3108 –3113, may 2010.

[12] G. Pons-Moll, A. Baak, T. Helten, M. Müller, H.P. Seidel, and B. Rosenhahn. Multisensor-fusion for 3d full-body human motion capture. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[13] PrimeSense Inc. *Prime Sensor™ NITE 1.3 Algorithms notes*, 2010. Last viewed 19-01-2011 15:34.

[14] Radu Bogdan Rusu and Steve Cousins. 3d is here: Point cloud library (pcl). In *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 9-13 2011.

[15] L. A. Schwarz, A. Mkhitaryan, D. Mateus, and N. Navab. Human skeleton tracking from depth data using geodesic distances and optical flow. *Image and Vision Computing*, to apear.

[16] L. A. Scwarz, D. Mateus, V. Castanneda, and N. Navab. Manifold learning for tof-based human body tracking and activity recognition. *British Machine Vision Conference (BMVC)*, 2010.

[17] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1297 –1304, june 2011.

[18] S. Soutschek, J. Penne, J. Hornegger, and J. Kornhuber. 3-d gesture-based scene navigation in medical imaging applications using time-of-flight cameras. *Computer Vision and Pattern Recognition Workshops*, 2008.

[19] Y. Sun, M. Bray, A. Thayananthan, B. Yuan, and P. Torr. Regression-based human motion capture from voxel data. *British Machine Vision Conference (BMVC)*, 2006.

[20] L. Velho, P. C. Carvalho, E. Medeiros, A. Sá, , A. Montenegro, A. Peixoto, and L. Rivera. *Fotografia 3D*. 25º Coloquio Brasileiro de Matematica - IMPA, Rio de Janeiro, 2005.

[21] Y. Zhu, B. Dariush, and K. Fujimura. Controlled human pose estimation from depth image streams. *Computer Vision and Pattern Recognition Workshops*, 2008.