Rastreando a Mão com o Filtro de Partículas com Hierarquia de Subespaços

Bruno Cedraz Brandão

Dissertação de Mestrado

Instituto de Computação
Universidade Estadual de Campinas

Rastreando a Mão com o Filtro de Partículas com Hierarquia de Subespaços

Bruno Cedraz Brandão

Janeiro de 2006

Banca Examinadora:

- Prof. Dr. Siome Goldenstein
  Instituto de Computação, Unicamp (Co-orientador)
- Prof. Dr. Paulo Cezar Pinto Carvalho
  Instituto Nacional de Matemática Pura e Aplicada
- Prof. Dr. Ricardo Machado Leite de Barros
  Faculdade de Educação Física, Unicamp
- Prof. Dr. Jorge Stolfi
  Instituto de Computação, Unicamp

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA CENTRAL DA UNICAMP
Bibliotecário: Helena Joana Flipsen – CRB-8ª / 5283

Brandão, Bruno Cedraz.
B733r Rastreando a mão com o filtro de partículas com hierarquia de subespaços / Bruno Cedraz Brandão. -- Campinas, SP : [s.n.], 2006.

Orientadores: Jacques Wainer, Siome Klein Goldenstein.
Dissertação (mestrado) - Universidade Estadual de Campinas, Instituto de Computação.

1. Visão por computador. 2. Rastreamento automático. 3. Interfaces (Computador). I. Wainer, Jacques. II. Goldenstein, Siome Klein. III. Universidade Estadual de Campinas. Instituto de Computação. IV. Título.

Tradução do título em inglês: Hand tracking with the subspace hierarchical particle filter.

Palavras-chave em inglês (Keywords): Computer vision, Automatic tracking, Computer interfaces.

Área de Concentração: Visão Computacional.

Titulação: Mestre em Ciência da Computação.

Banca examinadora: Siome Klein Goldenstein, Paulo Cezar Pinto Carvalho, Ricardo Machado Leite de Barros.

Data da defesa: 21-02-2006.

Rastreando a Mão com o Filtro de Partículas com Hierarquia de Subespaços

Este exemplar corresponde à redação final da Dissertação devidamente corrigida e defendida por Bruno Cedraz Brandão e aprovada pela Banca Examinadora.

Campinas, 21 de fevereiro de 2006.

Prof. Dr. Jacques Wainer
Instituto de Computação, Unicamp
(Co-orientador)

Prof. Dr. Siome Goldenstein
Instituto de Computação, Unicamp
(Co-orientador)

Dissertação apresentada ao Instituto de Computação, UNICAMP, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

TERMO DE APROVAÇÃO

Tese defendida e aprovada em 21 de fevereiro de 2006, pela Banca examinadora composta pelos Professores Doutores:

Prof. Dr. Paulo Cezar Pinto Carvalho
IMPA / RJ.

Prof. Dr. Ricardo Machado Leite Barros
FEEC / UNICAMP.

Prof. Dr. Siome Klein Goldenstein
IC / UNICAMP.

Resumo

Nesta dissertação, tratamos o problema de rastrear modelos hierárquicos através de visão computacional no contexto de interfaces gestuais. Gestos formam uma modalidade importante de comunicação humana, e ainda assim, existem poucas, e limitadas, aplicações computacionais com base em gestos. Nosso trabalho é mais uma iniciativa para reverter este quadro.

Começamos justificando e discutindo aplicações para a interface visual de gestos da mão. Descrevemos os componentes de um sistema de reconhecimento capaz de tornar esta interface possível. Revisamos a teoria Bayesiana da probabilidade, discutimos suas vantagens, os motivos que adiaram sua adoção em larga escala e derivamos, a partir dela, o filtro de partículas.

O filtro de partículas pode ser visto como uma solução aproximada para o problema de estimativa de parâmetros. É usado, por exemplo, para determinar os ângulos das juntas de um modelo tridimensional da mão que melhor caracterizem a pose e posição de uma mão real, gravada em uma seqüência de vídeo. Entretanto, a performance do filtro degrada a medida que aumentamos o número de dimensões do espaço de parâmetros. Nestes casos, um número exponencialmente maior de partículas é necessário para que o filtro convirja, inviabilizando aplicações interativas.

A nossa principal contribuição é o Filtro de Partículas com Hierarquia de Subespaços. A partir da estrutura presente em modelos hierárquicos, propomos uma forma de dividir o espaço de parâmetros em subespaços que são atribuídos a filtros de partículas diferentes, organizados na forma de um grafo acíclico orientado. Esta construção apresenta melhor convergência que o filtro de partículas original, possibilitando a recuperação dos parâmetros do modelo da mão com um número sensivelmente menor de partículas.

Usamos uma luva com seis marcadores idênticos para facilitar a extração de dados da imagem e então poder focar o trabalho na recuperação dos parâmetros do modelo. Dado o número restrito de marcadores, o modelo que usamos possui 15 graus de liberdade, nove a menos que um modelo da mão completo. Implementamos um sistema de reconhecimento, com base neste filtro, que roda a 30 quadros por segundo em um computador pessoal topo de linha e o validamos através de seqüências de vídeo reais e sintetizadas.

Abstract

This dissertation deals with the problem of tracking hierarchical models on the context of gesture interfaces based on computer vision. Gestures are an important medium for human communication, but there are few, and very limited, computational applications that rely on them. This work is an attempt to revert this.

We start by discussing and justifying several applications for a hand gesture interface. We describe the components of a recognition system that is capable of implementing such an interface. We review the Bayesian theory of probability, discuss its advantages, the reasons that holded back its large scale adoption, and we derive the particle filter on its terms.

The particle filter may be thought as an approximate solution for the parameter estimation problem. It may be used, for example, to recover every joint angle of a tridimensional hand model in order to match the pose and position of a real hand captured on video. Unfortunately, particle filtering does not scale gracefully to high dimensional parameter space applications. In these situations, a exponentially larger particle pool is required to ensure convergence, making it difficult to implement an interactive application.

Our major contribution is the Subspace Hierarchical Particle Filter. We claim that, with this method, we are able use the inherent structure present in hierarchical models to extract subspaces from the parameter space and assign them to different particle filters, organized into a directed acyclic graph structure. This construction improves the overall convergence, making it possible to recover hand model parameters with much less particles.

We employed a glove, with six identical markers painted on, to make it easier to extract data from the images. That way, we could focus on the model parameter recovery problem. Due to the small number of markers, we were able to work with 15 parameters only, nine less than a complete hand model. We built a recognition system based on the Subspace Hierarchical Filter that runs at 30 frames per second on a high-end personal computer. Finally, we validated our claims through real and synthesized video sequences.

Agradecimentos

Agradeço à minha mãe, Maria José Cedraz, sem seu suporte este trabalho não teria sido possível, à minha avó, Marcila, e aos meus irmãos, Pascífico, Rita, Miguel, Luciano, João Paulo e Esperança, pelo apoio.

Agradeço a Jacques Wainer pela iniciativa e espírito aventureiro, a Siome Goldenstein pelas boas sugestões, a Alexandre Falcão pela atenção e a Leizer Schnitman e a Wilton Oliver por acreditarem em mim. Agradeço à CAPES que, por intermédio do IC-UNICAMP, ajudou a financiar este projeto.

Aos meus amigos, sem nenhuma ordem em particular, Latino Neto, Luís Antônio Bispo, Christian Emmanuel, Alexandro Baldassin, Augusto Devegilli, Danilo Prates, Danilo Câmara, Ramaiana e Pollyana Arruda, Carlos Machado, Patrícia Buttini, Everton Constantino, André Rebouças, José Vitor Júnior, Adriano Guerreiro e a tantos outros, muito obrigado, onde quer que estejam.

Sumário

Lista de Tabelas

Lista de Figuras

Capítulo 1

Introdução

O objetivo deste trabalho, desde o início, foi o de construir um sistema capaz de reconhecer gestos da mão visualmente e que funcionasse interativamente. O caminho que escolhemos foi o de usar um modelo tridimensional para a mão, e rastreá-lo com o maior número de graus de liberdade que fosse possível. A correspondência obtida entre seqüência de vídeo e modelo, alimenta um reconhecedor que efetivamente relata que gestos foram feitos. Esta seqüência de gestos pode ser então usada para controlar um aplicativo.

Rastrear sistemas articulados e reconhecer movimentos humanos são duas tarefas difíceis. Da forma que modelamos o sistema, precisávamos construir um rastreador antes. O reflexo disto é que a maior parte deste trabalho é dedicado ao rastreamento da mão. Implementamos um reconhecedor mínimo para ter um sistema completo, mesmo que limitado. A maior contribuição deste trabalho, o Filtro de Partículas com Hierarquia de Subespaços, é um método Bayesiano para recuperação de parâmetros adequado para modelos articulados, não apenas a mão.

A seguir, expomos a motivação para um sistema como este e damos uma visão geral dos componentes que normalmente o compõem.

1.1 Qual o Interesse em Gestos?

Por que há tanto interesse em interfaces gestuais? Ou melhor, qual a razão para tornar computadores capazes de reagir a gestos? Uma das razões é que boa parte da comunicação humana é feita através de gestos. Chamamos de gesto, os movimentos do corpo, particularmente da cabeça e mãos. A comunicação gestual é prática. Muitas vezes é mais rápido indicar um objeto com um gesto que tentar descrever detalhes suficientes da sua aparência ou localização a ponto de ser possível distingui-lo dos demais. Gestos podem ser usados para ilustrar a dinâmica de uma situação, por exemplo, a interação de duas peças em um sistema mecânico.

Gestos podem ter significado simbólico próprio. Usamos gestos simbólicos, por exemplo, quando queremos chamar a atenção do garçom em um restaurante ou quando queremos nos comunicar em um ambiente em que voz ou escrita não é possível. O gesto não é apenas um complemento de outras formas de comunicação. A comunicação exclusiva através de gestos é possível, a exemplo das línguas de sinais. O fato é que a comunicação gestual, tanto quanto a comunicação vocal, é natural para o ser humano.

Gestos entretanto, não são universais. Uma pessoa poderia indicar um objeto com a mão, com o beiço, com o olhar, com o pé e de diversas outras formas. O gesto é uma característica própria não só de uma cultura, como também do indivíduo. Um exemplo, que se caracteriza pela diferença cultural, é o gesto para o "sim" indiano que parece muito com o gesto ocidental para o "não". As línguas de sinais também refletem estas diferenças. Cada país adota uma convenção diferente dos demais, por exemplo, no Brasil usamos a LIBRAS (Língua Brasileira dos Sinais), nos Estados Unidos é usada a ASL (*American Sign Language*) e na Inglaterra a BSL (*British Sign Language*). Notamos que, ao contrário do que um leigo esperaria, a ASL e a BSL são ininteligíveis entre si.

Não usamos gestos apenas para comunicação. Manipulamos os objetos a nossa volta através de gestos. Mais ainda, comportamento humano pode ser inferido através de gestos. Nosso estado emocional tem reflexo em nossos gestos. É possível perceber quando uma pessoa está emocionalmente, ou fisicamente, incapacitada para realizar alguma tarefa. É possível determinar a reação física da pessoa a efeitos do ambiente como frio, calor e insetos. É possível estimar a intenção da pessoa através de gestos, fazemos isto o tempo todo.

Claramente, grandes avanços seriam possíveis no desenvolvimento da computação onipresente e realidade aumentada com um sistema robusto de reconhecimento e rastreamento de gestos. Colocando de outra forma, tornar sistemas computacionais capazes de interpretar gestos, não só tem o potencial de melhorar as formas de interação atuais, como permitirá novas modalidades de comunicação humano-computador atualmente inviáveis.

1.2 Gestos da Mão e Aplicações

Estamos interessados no rastreamento e reconhecimento de gestos da mão através de visão computacional. Gestos manuais são necessários em diversas aplicações, como será ilustrado adiante, e possuem peculiaridades que justificam seu estudo por si. Entre elas, a anatomia da mão, com seus muitos graus de liberdade, a velocidade e a diversidade de movimentos.

A escolha de visão computacional pode ser bem justificada. Câmeras de vídeo estão cada vez mais acessíveis e presentes. Muitos aparelhos de telefone móveis e PDA's vêm com câmeras embutidas e muitos computadores possuem *webcams* instaladas, potenciali-

zando a aplicação doméstica para algoritmos de visão. Além disto, visão computacional tende a liberar o usuário da interface, isto é, idealmente não precisaria vestir luvas ou quaisquer outros dispositivos, deixado-o livre para suas outras tarefas.

Como descrito na seção anterior, gestos são uma forma de comunicação e interação natural ao ser humano, o problema é determinar como usá-los para interagir com o computador. É importante que a tecnologia forneça possibilidades não presentes atualmente, ou que melhore sensivelmente as presentes, caso contrário as pessoas não vão investir na mudança [22].

Para maioria das aplicações implementadas nos computadores de hoje, a interface **WIMP** (*Window, Menu, Icon, Pointing Device*) é adequada. Entretanto, em algumas situações, suas limitações são aparentes. Aplicações com número elevado de funções forçam o usuário a gastar muito tempo gerenciando a interface ao navegar por menus, botões, ícones e mover janelas que eventualmente estejam umas sobre outras. Estes inconvenientes podem ser minimizados através de teclas de atalho e da ampliação da área de visualização, por exemplo. Em dispositivos portáveis, nenhuma das duas soluções é geralmente possível ou mesmo desejável. Nestes dispositivos a interface é o gargalo e há espaço para testar novas alternativas.

A interface WIMP é menos que adequada em sistemas que dependem de manipulação tridimensional de objetos. O mapeamento dos dispositivos de entrada tradicionais de duas dimensões para três dimensões não é imediato. Embora existam periféricos com seis graus de liberdade e resposta tátil [37] voltados para treinamento médico e projeto tridimensional, o custo não justifica seu uso para simples visualização. Há realmente um espaço nesta área, que pode ser complementado com uma interface gestual que use a posição e orientação das mãos como guia para a disposição dos objetos na cena.

A idéia de que a melhor interface é nenhuma interface [55] é bem recebida. O ideal da computação onipresente, embora pareça distante, é que os dispositivos computacionais percebam a intenção do usuário com pouca ou nenhuma influência de sua parte. Imagine esta situação:

> Uma pessoa senta no sofá enquanto olha para o aparelho de TV desligado. O computador doméstico (ou mesmo o próprio aparelho) reconhece o padrão e imediatamente sintoniza no canal preferido para o horário. Se, por acaso, a pessoa só estivesse a pensar na vida sem nenhum interesse na televisão, bastaria que erguesse o braço, talvez ao mesmo tempo que move a cabeça ligeiramente para o lado, para que o aparelho não o incomodasse mais. Caso realmente estivesse interessado na programação, entretanto não no canal sintonizado, com um simples movimento vertical da mão de cima para baixo a seleção de canais surge na tela, à la *The Lawnmower Man*[1]. Os movimen-

[1]Filme de ficção cientifica que aborda a realidade virtual, lançado em 1992, dirigido por Brett Leonard

> tos da sua mão são correspondidos por um cursor na tela que é dividida em regiões grandes o suficiente para evitar a necessidade de movimentos precisos por parte dele. Quando finalmente decide qual canal assistir, inclina rapidamente a mão para frente como se estivesse batendo na região selecionada e o canal é sintonizado. Este não lembra mais o que é um controle remoto.

Podemos pensar em muitas outras aplicações para gestos. Animadores poderiam usar gestos para controlar marionetes virtuais por uma fração do custo dos sistemas eletromecânicos de hoje. Algoritmos de compressão de vídeo [48] novos poderiam ser desenvolvidos a partir da predição potencializada pela interpretação dos gestos. A predição também tem o potencial de melhorar os sistemas de vigilância [32], cujo propósito é focar a atenção do operador para as situações mais suspeitas. Uma forma prática, robusta e barata de extrair, segmentar e classificar gestos auxiliaria na pesquisa de comportamento humano. O gesto é tão característico para cada pessoa, que existem trabalhos em que o reconhecimento do indivíduo é feito através de sua silhueta e movimentos corporais [8].

Interfaces gestuais podem ser usadas para gerar transcrições automáticas das línguas dos sinais. Há muito potencial para jogos de computador, inclusive há alguns produtos no mercado como o *EyeToy* para o *PlayStation* 2. Seguindo a tendência, o próximo lançamento da *Nintendo* para 2006, o *Revolution*, tem, no seu controlador, um sensor de movimento. O gesto manual, embora não adquirido através de visão, passa a ser interface principal da maioria dos jogos para este dispositivo.

Uma proposta de uso educacional de gestos da mão é o Enhanced Desk [31], que permite trabalho cooperativo e fusão de mídia eletrônica e impressa através de projetores e câmeras de vídeo. A informação é projetada na mesa, e os gestos do usuário sobre a mesa são capturados por meio de uma câmera infravermelha. Em uma das suas aplicações, o livro possui texturas que indicam ao computador que informação multimídia deve projetar na mesa. Esta informação pode ser uma página na internet, uma nota do autor, ou mesmo um programa interativo, como simuladores, que acompanhariam livros de física e respondem aos gestos do usuário.

Gesto tem o potencial de ser inserido em uma interface de uso comum, principalmente nas diversas aplicações nas quais outros tipos de interfaces deixam a desejar ou são impraticáveis.

1.3 Problemas com Interfaces Gestuais Visuais

Infelizmente, há razões para que a interface gestual, através de visão computacional, não esteja tão disseminada. Entre elas o custo computacional dos algoritmos de visão, o

e produzido pela New Line Cinema.

número de graus de liberdade, a maleabilidade da mão e a velocidade do movimento. Há propostas de hardware específico, como câmeras de alta velocidade, que simplificariam os algoritmos de correspondência e minimizariam o problema da velocidade do movimento [45]. Entretanto, estes dispositivos são voltados para aplicações industriais, e, até o momento, apresentam qualidade de imagem inferior a das câmeras mais difundidas. Outro problema é o ambiente no qual os gestos são extraídos. No caso geral, o fundo da cena, formado por todos os objetos que não são os de interesse, é complexo contendo padrões que podem confundir os algoritmos. Comunicação por gestos tem a vantagem de ser imune a ruído sonoro, mas, em geral, é dependente de um bom ângulo de visão. Pode-se apenas estimar de forma limitada se a mão estiver ocultada por outro objeto ou auto-ocultada.

O gesto, na grande maioria das vezes, tem seu sentido dependente do contexto. Sistemas, com exceção dos mais triviais, teriam que ser capazes de inferir este contexto. Existem propostas para acoplar bases de conhecimento ao sistema de reconhecimento para dotá-los de conhecimento específico de domínio e, com isto, reduzir a ambiguidade, a exemplo do trabalho de Miners, Basin e Kamel [39].

Outro problema é a **segmentação temporal**. Gestos, em geral, são contínuos. É necessário separar o gesto de interesse de movimentos involuntários e dos demais gestos. A distinção das três fases do gesto: preparação, execução e retorno, muitas vezes não é observada, isto é, gestos podem se sobrepor e, freqüentemente, o fazem. Entretanto, entendemos que alguns tipos de gestos são usados em situações diferentes. Algumas aplicações podem depender apenas de uma classe específica, o que possivelmente facilitaria o processo de segmentação. O primeiro passo para estudar as características de cada classe é classificar os gestos. Existe uma taxonomia proposta por Pavlovic *et al.* [43], voltada especificamente para reconhecimento de gestos da mão.

Nesta taxonomia os movimentos são separados em **gestos** e **movimentos não intencionais**. Os **movimentos não intencionais** não são necessariamente inúteis, podem ser usados, por exemplo, para inferir comportamento. **Gestos**, por sua vez, são separados em duas grandes classes: **manipulativos** e **comunicativos**. Gestos **manipulativos** são usados para interagir com o mundo físico. Entretanto, o controle de objetos virtuais através de gestos no qual há *feedback* contínuo para o usuário, também faz parte desta classe. Os gestos **comunicativos** estão divididos em **atos** e gestos **simbólicos**. Os **atos** podem ser **mímicos**, que consistem principalmente da imitação de alguma outra entidade, ou **dêiticos**, que é o gesto de apontar. Os **simbólicos** podem ser **referenciais**, aqueles que têm significado próprio e completo, como o gesto para chamar o garçom, e **modificadores**, que são usados em conjunto com outras modalidades de comunicação para complementar a informação, como o pescador indicando o tamanho do peixe com os braços estendidos.

Temos algumas observações a fazer a respeito de gestos manipulativos. Assumindo que consigamos um sistema capaz de capturar confiavelmente os gestos da mão. Qual o efeito de usar gestos manipulativos na ausência do sentido do tato, ou, no caso de algumas interfaces planas, na presença de sensação tátil contraditória ao que se esperaria numa situação real? Experimentos com privação sensorial, a exemplo dos tanques de privação sensorial de John Lilly, podem induzir ansiedade e alucinações [19]. O quanto isto se aplica à ausência apenas do tato em termos de desconforto é algo que precisa ser determinado se desejamos usar interfaces manipulativas, sem apoio de objetos, por longos períodos.

É fácil constatar que, em algumas aplicações, a ausência do tato afeta a produtividade. Em um sistema de modelagem tridimensional por exemplo, independente da orientação do objeto na tela, se há resposta tátil, o usuário sabe imediatamente quando tocou a superfície do objeto, e possivelmente, até a textura. A indústria de jogos eletrônicos e simuladores há muito tempo dotam dispositivos de controle com *Force-feedback*, particularmente manches. É interessante constatar que, no caso de uma interface não voltada para treinamento, a resposta do dispositivo não precisa ser fisicamente realista. Respostas que apenas parecem corretas intuitivamente, a exemplo do jogo *Freespace 2* (1999), são suficientes para incrementar o efeito de imersão.

Uma interface manipulativa realizada puramente através de visão está, de certa forma, contra a corrente da teoria da manipulação direta de Shneiderman [50], que tende a tornar o tato, cada vez mais, parte da interface. Outro ponto a considerar é a ergonomia. Falta de apoio para os braços garantidamente inviabiliza qualquer aplicação de longa duração.

Entretanto, há formas de contornar alguns problemas em domínios específicos e não há dados que determinem os limites reais que a falta de tato e os fatores ergonômicos imporiam. Como exposto anteriormente, gestos são particulares não só de cada cultura, mas para cada indivíduo. Como as pessoas vão responder às limitações de gestos destes sistemas? Vai ser possível montar um conjunto comum de gestos? Caso contrário, quanto os sistemas terão que se adaptar ao usuário? Estes são mais temas para pesquisa.

1.4 Sistemas de Reconhecimento

Temos bastante confiança a respeito da possibilidade de programar um computador de forma a torná-lo capaz de reconhecer gestos, tendo em vista que o cérebro consegue recuperar a posição e a pose da mão com um olho apenas. Nesta seção, descrevemos uma proposta para o sistema de reconhecimento.

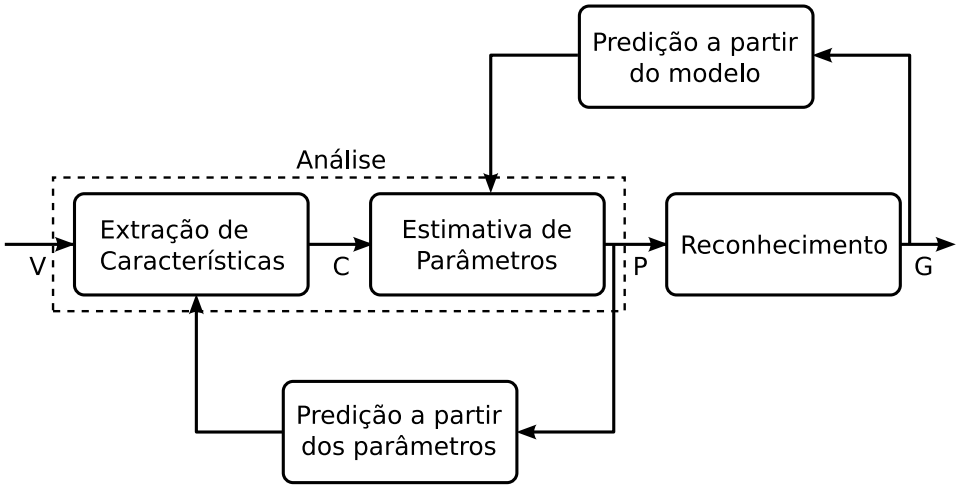


Figura 1.1: Análise e reconhecimento dos gestos. V representa as imagens, C as características, P os valores dos parâmetros e G os gestos reconhecidos.

1.4.1 Um Sistema de Reconhecimento

Sistemas de reconhecimento podem ser divididos em três etapas principais: **Extração de características**, **estimativa de parâmetros** e **reconhecimento**. A figura 1.1, originalmente encontrada no artigo de Pavlovic [43], sumariza as três etapas.

A entrada do sistema é uma seqüência de vídeo V. A primeira etapa extrai características C da imagem, úteis para o rastreamento (seção 1.4.3). Exemplos de características relevantes para gestos manuais são: posição das pontas dos dedos, contorno da mão, posição das juntas e posição da palma. A etapa de estimativa de parâmetros tenta inferir qual conjunto de valores P os parâmetros do modelo assumiriam para melhor explicar as características observadas. Como será apresentado no capítulo 4, usamos um modelo esquelético tridimensional da mão, no qual os ângulos das juntas e da palma, junto com a posição da palma, formam os parâmetros. As duas primeiras etapas formam o que se chama de fase de análise. Esta fase é responsável pelo rastreamento do objeto de interesse. A etapa de reconhecimento deve decidir, a partir do conjunto de valores de parâmetros, se houve ou não algum gesto que nos interessa. O reconhecedor, então, fornece a relação de gestos, G, junto com seus parâmetros para o aplicativo final.

É útil usar informação de mais alto nível para reduzir a região de busca em uma etapa anterior. Portanto, fazemos com que a etapa de reconhecimento realimente a etapa de estimativa de parâmetros e a etapa de estimativa realimente a etapa de extração. A seguir descrevemos cada uma das etapas com mais detalhes.

1.4.2 Extração de Características

Encontrar características adequadas ao problema é uma tarefa importante. Sem boas características, o sistema tem aplicabilidade limitada ou nenhuma. Por exemplo, se as características são demasiadamente ambíguas, a etapa seguinte poderá ter problemas em determinar quais partes do modelo correspondem a quais características e irá se confundir facilmente. Isto é particularmente verdade, se as características forem pontuais. Se a extração não for robusta, isto é, se há muita informação contradizente, como características que indicam a presença de um objeto em lugares onde não há, ou se é intermitente, a próxima etapa certamente terá problemas em encontrar os valores de parâmetros que representem o vídeo.

Podemos usar luvas coloridas para facilitar a extração, mas isto impõe restrições ao sistema, tanto devido ao fato do usuário ter que usar o equipamento, quanto à limitação de cores do fundo da cena. Entretanto, usando uma luva com marcadores únicos suficientes [14], um fundo de cena escuro e uma câmera com resolução adequada, é possível recuperar os valores de todos os parâmetros da mão através de cinemática reversa e otimização não linear, quadro a quadro. Se o fundo de cena não pode ser feito tão neutro assim e a resolução da câmera for baixa, a luva ainda pode ser usada mas, como a quantidade de cores para marcadores será reduzida e menos detalhes estarão disponíveis, limitará os gestos que poderão ser rastreados.

1.4.3 Estimativa de Parâmetros

A etapa de estimativa de parâmetros é implementada, comumente, através de um filtro Bayesiano. É a última etapa da fase de análise que, normalmente, é responsável pelo rastreamento. **Rastreamento** pode ser entendido como a atividade de inferir o estado, ou seja, os valores dos parâmetros de um modelo, a partir de uma seqüência de observações que, no nosso caso, são as características. Estes parâmetros podem ser, por exemplo, **pose**[2], movimento e posição. Rastreamento não é limitado a objetos físicos. Sigal [51], por exemplo, usa para estimar a distribuição dos valores da cor da pele em um espaço de cores após mudanças nas condições de iluminação, assumindo que rotação, translação e escala são as operações possíveis sobre a distribuição. O capítulo 2 trata especificamente de filtros Bayesianos, ou preditivos, como também são conhecidos.

[2]Aqui, consideramos que a pose de um modelo é representada por todos os parâmetros que não fazem parte das transformações rígidas. No caso do modelo da mão, as transformações rígidas são representadas pela posição tridimensional e do ângulo da base da palma e a pose pelos ângulos das juntas dos dedos.

1.4.4 Reconhecimento

Reconhecimento, a não ser que restrito a um escopo muito limitado, é o problema mais difícil dos anteriores. Wexelblat [61] defende que os únicos sistemas gestuais interessantes são os capazes de reconhecimento contínuo de gestos, uma vez que gestos discretos tiram a naturalidade e reduzem-se ao nível de teclas de função. Isto é, para que forçar um gesto não natural, se já podemos usar gestos não naturais para apertar um botão em um controle remoto? Entendemos por gestos discretos aqueles nos quais o usuário é forçado a mover a mão para uma posição determinada, executar o gesto, e retirá-la após o gesto ser executado, um gesto de cada vez. Sistemas de reconhecimento contínuos são obrigados a lidar com a segmentação temporal dos gestos, um problema complexo como já discutimos. Além disto, como visto anteriormente, existe o problema da ambiguidade que não pode ser tratado localmente, a não ser que o universo de possibilidades de gestos seja muito limitado.

Modelos Ocultos de Markov [47, 58, 5], ou HMM, vêm sendo aplicados com relativo sucesso na área de reconhecimento de gestos. Possui as vantagens de realizar a segmentação automática e suportar algum grau divergência entre o modelo do gesto e a seqüência de parâmetros observada. Entretanto, é dependente de treinamento, e não escala bem. Para cada gesto, um modelo novo tem que ser introduzido. Um outro problema é tratar movimentos não intencionais. Dificilmente a pessoa consegue evitar estes movimentos e portanto devem ser tratados juntamente com os tipos de gestos esperados pelo sistema. Em reconhecedores que usam HMM, um modelo separado é usado para tratar estes gestos.

1.5 Organização Deste Trabalho

Organizamos o resto da dissertação como se segue. No capítulo 2, expomos o método Bayesiano que utilizamos no rastreador. No capítulo 3, introduzimos o filtro de partículas com hierarquia de subespaços, como uma forma para tornar o rastreador mais eficiente e, portanto, mais próximo de ser executado interativamente. No mesmo capítulo, comparamos a proposta com trabalhos na literatura que envolvem reconhecimento visual de gestos da mão ou que envolvem rastreamento visual de objetos articulados. No capítulo 4, aprofundamos os detalhes da implementação do rastreador e do reconhecedor e apresentamos dados experimentais que validam a melhora obtida com o nosso filtro. No capítulo 5 concluímos o trabalho e apresentamos propostas para trabalhos futuros. Nos apêndices, relacionamos algoritmos e resultados experimentais.

Capítulo 2

Método Bayesiano para Rastreamento de Parâmetros

Neste capítulo, exploramos uma forma de extrair os parâmetros do modelo da mão a partir do vídeo através de um método Bayesiano. Justificamos o uso da teoria da probabilidade e, em seguida, revisamos brevemente a interpretação Bayesiana. Um método de estimação de parâmetros baseado nesta teoria é apresentado, seguido do filtro de partículas.

2.1 Justificativa

É consensual que, quanto melhor é a extração de características, mais fácil é extração de parâmetros, e vice-versa. A exemplo do rastreamento da mão, se, a cada quadro, dispomos das coordenadas na imagem de cerca de dezesseis regiões conhecidas da mão, podemos obter uma solução fechada para a determinação dos parâmetros a partir do algoritmo dos três pontos [20]. Entretanto, em geral, não dispomos de equipamento com precisão suficiente, não podemos contar sempre com restrições no fundo da cena e/ou a complexidade do algoritmo de extração torna impraticável o seu uso. No caso da mão, há ainda outro agravante: a auto-oclusão pode impedir a aquisição de informação suficiente para a solução fechada.

Uma forma de tentar diminuir a complexidade da etapa de extração de características é usar uma solução probabilística na etapa seguinte. Uma técnica relativamente simples pode ser usada na extração de características que deixa alguma ambiguidade para ser resolvida na etapa de estimação de parâmetros. Métodos Bayesianos são populares na área de visão computacional. Dentre eles o filtro de partículas, por sua generalidade e implementação simples, é comumente usado para implementar esta última etapa. A seguir revisamos a teoria da probabilidade necessária para o entendimento do filtro e dos capítulos posteriores.

2.2 Vertentes da Probabilidade

A teoria da probabilidade é o meio matemático mais usado para tratar a incerteza. Alternativas seriam aritmética afim [11] e lógica nebulosa [2] por exemplo. Existem duas vertentes, ou interpretações, populares para a teoria. A vertente freqüencista limita-se a interpretação de probabilidades originadas como razões de ocorrência em seqüências de experimentos repetidos. Esta vertente foi a que credibilizou a probabilidade no século XX como uma teoria sólida, definida axiomaticamente[1], e com um meio mecânico e racional de determinar e trabalhar com probabilidades. Esta é a interpretação da probabilidade mais difundida atualmente. Muitas referências importantes existem em termos exclusivamente freqüencistas, *e.g.* Feller [18] e Papoulis [42].

A outra vertente é a Bayesiana. Nesta, a probabilidade é interpretada como o grau de plausabilidade que se tem a cerca de uma hipótese ou proposição. Duas das maiores diferenças entre as vertentes, é que a teoria Bayesiana possui regras para inferência e não considera freqüências como a única fonte de probabilidades. Do ponto de vista freqüencista, inferência não é parte da probabilidade, mas de outra área correlacionada, a inferência estatística. Embora consiga bons resultados em casos específicos, e seja aplicada extensivamente, usar inferência estatística geralmente envolve aplicar uma solução *ad hoc* para cada aplicação, ao passo que, a inferência Bayesiana é uma solução geral. Uma crítica comum aos métodos da inferência estatística [26] é que, se a aquisição dos dados sofrer distúrbios, como uma interrupção prematura, toda a análise deve ser refeita, pois a interrupção pode descaracterizar a **variável aleatória**[2] decidida inicialmente.

Podemos citar três fatores que restringiram a aceitação em larga escala da teoria Bayesiana. O primeiro é uma justificativa para a escolha das fórmulas, aparentemente arbitrárias, para lidar com plausabilidades. Isto é, uma garantia de unicidade e consistência, de forma que, a partir de um mesmo conjunto de dados, ou seja, de um mesmo estado de conhecimento, não seja possível inferir dois resultados diferentes. Para probabilidades definidas a partir, exclusivamente, de freqüências, as fórmulas definidas a partir dos axiomas de Kolmogorov são trivialmente adequadas, mas para graus de plausabilidade esta conclusão não existia. Cox [10] solucionou o problema provando que, se o grau de conhecimento de uma hipótese é definido como um número real, então a forma consistente e única de tratá-lo é através das regras definidas por dois funcionais, que são equivalentes às equações dos axiomas da probabilidade Bayesiana[3].

[1] A interpretação axiomática da teoria das probabilidades deve-se ao russo Andrey Nikolaevich Kolmogorov.

[2] A literatura freqüencista batiza o mapeamento do espaço dos possíveis resultados de um experimento para os reais no intervalo $[0, 1]$ de variável aleatória. O conceito de variável aleatória é fundamental para a teoria freqüencista, mas é pouco citado na literatura Bayesiana, a menos quando a título de comparação. Em obras Bayesianas inteiras [6], o conceito não é referenciado uma única vez.

[3] Os axiomas Bayesianos são deriváveis dos de Kolmogorov e vice-versa.

O segundo fator diz respeito à atribuição das probabilidade iniciais, também conhecidas como **distribuições *a priori***. Em geral, temos um conjunto de hipóteses do qual precisamos escolher uma, por exemplo, possíveis valores para a posição da mão no espaço, onde cada valor é uma hipótese. Este conjunto pode ser discreto ou, para o caso generalizado até o limite, contínuo, *i.e.* temos um conjunto denso de hipóteses, uma para cada valor de posição nos reais. A primeira etapa da aplicação do método Bayesiano é definir a distribuição inicial das probabilidades sobre as hipóteses. Os Bayesianos consideram esta etapa como uma parte substancial da teoria e a solução de alguns problemas depende apenas da sua realização.

Se nos limitar-mos a freqüências, que são propriedades mensuráveis ou estimáveis do mundo real, temos uma forma racional de determinar as probabilidades para cada hipótese, embasada pela lei dos grandes números. Entretanto, esta pode ser uma simplificação demasiada. Segundo a interpretação Bayesiana, as freqüências correspondem a probabilidades apenas para alguns **estados de conhecimento**, *i.e.* quando tudo o que sabemos do sistema são as freqüências.

O método Bayesiano exige que qualquer **informação testável** relevante para o problema possa ser usada para a definição da distribuição inicial. Geralmente, este tipo de informação não define unicamente uma distribuição de probabilidade, mas, para manter a teoria consistente, deve haver um método para determinar uma distribuição única dado um conjunto inicial de dados. Uma opção lógica é, de todas as possíveis distribuições, escolher a mais imparcial, ou seja, a que representa apenas a informação disponível. A solução é derivada do trabalho de Shannon [49], e usa a noção da **entropia** da distribuição. A idéia é maximizar a incerteza da distribuição, representada pela entropia, respeitando as restrições impostas pelos dados. Esta técnica, **MaxEnt**, idealizada por Jaynes [26], torna plausível o uso de informação não freqüencial em distribuições de probabilidade.

Para ilustrar o MaxEnt, consideramos o recorrente exemplo do dado de seis lados [7]. Queremos determinar a probabilidade de cada lado ocorrer. A única restrição que temos é que a soma das probabilidades é um, que é a restrição mínima para uma distribuição de probabilidade. A expressão a maximizar é

$$H = \underbrace{-\sum_{x=1}^{6} P(L_x|B)\log P(L_x|B)}_{\text{entropia}} + \lambda \underbrace{\left[1 - \sum_{x=1}^{6} P(L_x|B)\right]}_{\text{restrição}}. \tag{2.1}$$

Onde λ é um multiplicador de Lagrange, L_x é a hipótese $\{Ocorrência\ do\ lado\ x\}$ e B representando nosso atual estado de conhecimento. A distribuição $P(L|B)$ deve armazenar as probabilidades das nossas seis hipóteses, e lemos cada um de seus elementos como a probabilidade de L_x dado B, ou seja, tudo que sabemos até o momento. As restrições são construídas de forma que seu valor seja zero, então H continua sendo o valor da entropia.

Para cada informação, uma restrição é acrescentada, com seu multiplicador de Lagrange associado. Derivando em relação a cada $P(L_x|B)$ e de λ, chegamos a um sistema de sete equações, cuja solução é

$$P(L_x|B) = 1/6 \text{ e } \lambda = 1 - \log 6. \tag{2.2}$$

Então o MaxEnt confirma o **princípio da indiferença**. Quando não há informação que privilegie uma hipótese sobre as demais, a distribuição[4] que maximiza a entropia é a uniforme. Quando a média e variância são usadas como restrições, a distribuição resultante é uma Gaussiana. Determinar distribuições iniciais a partir tipos de informação diferentes é uma área ativa de pesquisa da teoria Bayesiana. Existem outros princípios gerais além do MaxEnt, cada um capaz de associar probabilidades a diferentes classes de informação. Espera-se que a descoberta de princípios novos abra novas áreas de aplicação para a teoria.

Os resultados principais usados na resolução dos dois problemas anteriores são conhecidos desde a década de 40, entretanto a teoria Bayesiana permaneceu restrita a poucos grupos até meados dos anos 80. O terceiro fator é a complexidade dos cálculos integrais necessários para a aplicação da teoria Bayesiana. Neste aspecto, a teoria freqüencista é vantajosa. Apesar de restrita a freqüências, possui soluções de aplicação simples, como a inferência por distribuição χ^2, por exemplo. Felizmente, técnicas de integração numérica, como a Monte Carlo, unidas ao poder computacional crescente, aliviaram o problema dos cálculos, e hoje aplicar a teoria Bayesiana é, muitas vezes, a opção mais simples e, ao mesmo tempo, a mais abrangente.

2.3 Teoria Bayesiana da Probabilidade

A teoria da probabilidade, ou plausabilidade, pode ser desenvolvida de forma a obedecer três propriedades [33]:

(I) O grau de plausabilidade é representado por números reais.

(II) Deve haver uma correspondência qualitativa com o senso comum.

(III) A teoria deve ser consistente.

A primeira propriedade relaciona a plausabilidade com uma quantidade física fácil de trabalhar e implementar em computadores. Por convenção, quando mais plausível for a hipótese, maior este valor. A segunda propriedade torna a teoria consistente com a lógica

[4]Certas distribuições com características interessantes são muito recorrentes, como a Gaussiana e a uniforme, e seu estudo é de grande utilidade. Os trabalhos de Feller [18] e Papoulis [42] são referências importantes neste aspecto, contendo muitos resultados não triviais.

dedutiva quando uma hipótese é absolutamente verdadeira ou absolutamente falsa. Para ilustrar a propriedade, considere que A, B e C são três hipóteses e que $A|B$ representa a plausabilidade de uma hipótese A ser verdadeira dado que a hipótese B já é tida como verdade. Se, ao atualizar nosso conhecimento do problema de forma que a hipótese C seja reformulada como C' e desta forma $A|C' > A|C$ e $B|C' = B|C$, então é de se esperar que a plausabilidade de A e B sejam tais que $AB|C' > AB|C$.

A terceira propriedade reflete, na verdade, três noções de consistência. Conclusões que podem ser obtidas por duas ou mais deduções diferentes devem ter o mesmo resultado. A teoria deve ser capaz de usar toda a evidência relevante para o problema, ou seja, não deve haver a necessidade de, arbitrariamente, ignorar alguma informação disponível e basear a conclusão apenas nas restantes. Estados de conhecimento equivalentes devem ser associados a graus de plausabilidade equivalentes. Esta última afirmação é uma generalização do princípio da indiferença.

A partir destas três propriedades, que passaremos a chamar de *desiderata*, e com a opção de trabalhar com valores apenas no intervalo $[0, 1]$, é possível derivar as seguintes equações [29]:

$$P(AB|C) = P(A|BC)P(B|C), \tag{2.3}$$

$$P(A + B|C) = P(A|C) + P(B|C) - P(AB|C). \tag{2.4}$$

São a regra do produto (2.3) e a regra da soma (2.4), e formam os axiomas[5] da teoria Bayesiana das probabilidades. Em ambas as equações, C representa todo o conhecimento anterior, ou *a priori*. Com esta notação, podemos representar um conjunto de hipóteses mutuamente exclusivas[6] e exaustivas[7] por

$$\sum_{i=1}^{N} P(A_i|B) = 1. \tag{2.5}$$

Se o princípio da indiferença se aplica, ou seja

$$P(A_i|B) = \frac{1}{N},\ i = 1..N, \tag{2.6}$$

podemos derivar a **regra da urna de Bernoulli**. Se H é um subconjunto com M das N hipóteses A_i e a evidência B implica que estas são verdadeiras e as demais $N - M$ são falsas, então

$$P(H|B) = \frac{M}{N}. \tag{2.7}$$

[5] A regra da soma é geralmente axiomatizada como $P(A|B) + P(\overline{A}|B) = 1$. A equação (2.4) é uma generalização da regra, e é mais usada na prática.

[6] Se a hipótese A_i é verdadeira dado B, então $A_j|B$ é automaticamente falsa para $i \neq j$.

[7] Para um B qualquer, pelo menos algum A_i é verdadeiro.

Boa parte dos resultados da teoria freqüencista, bem como outros que não estão presentes nesta, podem ser derivados a partir da regra da urna e dos axiomas. Entretanto, vamos nos restringir àqueles que usados neste trabalho.

Muitas vezes precisamos extrair, ou sumarizar, informações simples[8], mas expressivas, de uma distribuição de probabilidades. Por exemplo, quando queremos reportar um estado da mão em um particular momento ou ter uma estimativa da precisão de uma distribuição. Descritores para este fim podem ser calculados através da técnica dos momentos. Considere uma distribuição $f(x) = P(A|B)$ onde x é um inteiro, A_x é uma hipótese afirmando $\{o\ valor\ do\ parâmetro\ X\ é\ x\}$ e o conjunto de hipóteses A_i, $i = 0..N$, exaustivo e mutuamente exclusivo. Desta forma $f(x)$, também conhecido como **função de densidade de probabilidades**, ou **PDF**, pode ser lido como "probabilidade do valor ser x dado o atual estado de conhecimento". O **valor esperado** ou **média** (2.8) e a **variância** (2.9), respectivamente, o primeiro momento e o segundo momento central, são dados por:

$$\mathbf{E}(X) = \mu = \sum_{x=0}^{N} x f(x), \tag{2.8}$$

$$\mathbf{Var}(X) = \mathbf{E}(X^2) - (\mathbf{E}(X))^2 = \sum_{x=0}^{N} (x-\mu)^2 f(x). \tag{2.9}$$

Outro descritor muito usado é a **mediana**. A mediana é definida como o valor m tal que as hipóteses $\{x > m\}$ e $\{x < m\}$ tenham igual probabilidade. A mediana também pode ser definida a partir da **distribuição acumulada de probabilidade** ou **CDF**. A distribuição acumulada $F(y)$ é dada por

$$F(y) = \sum_{x=0}^{y} f(x). \tag{2.10}$$

A mediana m é tal que $F(m) = 1/2$. Na realidade, para distribuições discretas, pode haver mais de uma mediana, ou nenhuma estritamente falando. É, respectivamente, o caso quando há um número par de hipóteses ou $F(y)$ não assume o valor $1/2$ para nenhum y válido. Geralmente é satisfatório assumir a mediana como sendo o valor

$$m = \text{argmin}_y (F(y) - 1/2)^2, \tag{2.11}$$

ou seja, m tal que $F(m)$ é mais próximo de $0,5$. A mediana tende a representar distribuições assimétricas melhor que a média e tende a ser menos sensível a valores extremos, também chamados de *outliers*.

[8] Geralmente estas sumarizações não são capazes de descrever unicamente a distribuição, mas assumem formas convenientes como um escalar, um vetor ou uma matriz de poucas dimensões.

2.4 O Teorema de Bayes

A partir da regra do produto (2.3), com os rótulos das hipóteses trocados para facilitar a interpretação,

$$P(\theta D|I) = P(D\theta|I) = P(\theta|DI)P(D|I) = P(D|\theta I)P(\theta|I), \tag{2.12}$$

podemos deduzir o Teorema de Bayes

$$P(\theta|DI) = P(\theta|I)\frac{P(D|\theta I)}{P(D|I)}. \tag{2.13}$$

A importância do teorema de Bayes está em determinar a probabilidade da nossa hipótese após atualizar nossas suposições, ou seja, o estado de conhecimento atual, com novas informações sem ter que reformular o problema. Segundo a interpretação usual do teorema, θ é um conjunto de hipóteses $\{\theta_1, \theta_2, \ldots\}$ a respeito do valor de um parâmetro de interesse, D é uma proposição que representa um novo dado e I representa o estado de conhecimento antes de avaliar o dado, *i.e.* a informação que sabemos *a priori*. $P(D|I)$ é um fator de normalização para manter $\sum_i P(\theta_i|DI) = 1$. $P(\theta|I)$ é a distribuição anterior, ou nosso conhecimento *a priori* a respeito das hipóteses. $P(\theta|DI)$ é a distribuição posterior, ou seja, a distribuição *a posteriori* de θ quando nosso estado de conhecimento é atualizado com o novo dado. A relação entre as distribuições *a priori* e *a posteriori* é lógica e não temporal. Diz respeito ao estado de conhecimento antes e depois de ser atualizado, que não necessariamente está relacionado a instantes de tempo.

$P(D|\theta I)$, quando fixamos D e θ é um conjunto de hipóteses, é chamada de **função de verossimilhança**. Representa uma distribuição de probabilidades, cada uma interpretada como a probabilidade do dado D ser adquirido quando assumimos que θ_i é verdade. O conjunto não é necessariamente uma distribuição de probabilidades válida, tendo em vista que, no geral, $\sum_i P(D|\theta_i I) \neq 1$. Apenas após multiplicada pelos outros fatores é que o resultado garantidamente será uma distribuição de probabilidade. Em outras palavras, a verossimilhança é um conjunto de probabilidades condicionais da observação em relação a cada hipótese θ_i.

Para determinar a distribuição *a posteriori* precisamos determinar os três termos à direita. A verossimilhança e a distribuição *a priori* são obtidas diretamente, isto é, a forma de determiná-las faz parte da modelagem do problema. O fator de normalização pode ser convenientemente reescrito nos termos destes dois. Para isto usamos o conceito de **marginalização**[9]. Estamos assumindo que alguma das hipóteses em θ é verdadeira[10],

[9] Segundo Loredo [33], o termo vem de uma prática antiga: listar as probabilidades conjuntas de duas variáveis discretas, $P(X, Y|I)$, em uma tabela e preencher as margens com o somatório das linhas e colunas, $P(X_i|I) = \sum_{j=1}^{n_Y} P(X_i, Y_j|I)$.

[10] Assumimos também que as hipóteses são mutuamente exclusivas e exaustivas.

logo, a hipótese $\{\theta_1$ ou θ_2 ou $\ldots\}$, ou seja, $\{\theta_1 + \theta_2 + \ldots\}$, é sempre verdade. O fator de normalização pode ser expandido então da seguinte forma:

$$\begin{aligned} P(D|I) &= P(D[\theta_1 + \theta_2 + \ldots]|I) && (2.14)\\ &= P(D\theta_1|I) + P(D\theta_2|I) + \ldots && (2.15)\\ &= \sum_i P(D\theta_i|I) && (2.16)\\ &= \sum_i P(D|\theta_i I)P(\theta_i|I). && (2.17) \end{aligned}$$

Desta forma, o Teorema de Bayes pode ser reescrito como:

$$\underbrace{P(\theta|DI)}_{\text{Posterior}} = \underbrace{\frac{\overbrace{P(\theta|I)}^{\text{Anterior}}\ \overbrace{P(D|\theta I)}^{\text{Verossimilhança}}}{\sum_i P(\theta_i|I)P(D|\theta_i I)}}_{\text{Normalização}}. \tag{2.18}$$

2.5 O Filtro Bayesiano

Enunciando de forma probabilística o problema de estimativa de parâmetros, precisamos, a cada quadro, determinar a distribuição de probabilidade dos parâmetros considerando todos os dados adquiridos até o momento e que esteja de acordo com o modelo da dinâmica do sistema. Os dados adquiridos em um determinado instante são também chamados de **observações**. O teorema de Bayes, da forma como anunciamos anteriormente, não considera diretamente esta dinâmica. Quando estamos tentando determinar parâmetros que variam com o tempo, a distribuição anterior pode não ser mais válida se a observação demora a chegar. Para completar o arcabouço do método Bayesiano precisamos acrescentar o modelo dinâmico do sistema.

Nossos parâmetros serão codificados na nossa hipótese $\theta_i \in \mathbb{R}^d$ que consiste de um vetor com d valores de parâmetros. Desta forma, podemos falar de espaço de hipóteses e espaço de parâmetros intercambialmente. Por razões históricas [1], este tipo de problema é dito um problema de filtragem. No instante $(k-1)$, temos a distribuição $P(\theta_{k-1}|D_{k-1})$ descrevendo o estado do sistema. No instante k, queremos obter a distribuição $P(\theta_k|D_k)$ que inclui o novo dado d_k e leva em conta a dinâmica do sistema entre os instantes $(k-1)$ e k.

Se considerar-mos D_k a proposição $\{d_1 d_2 \ldots d_k I\}$, ou seja, toda a informação adquirida até o instante k, podemos reescrever o Teorema de Bayes da seguinte forma:

$$P(\theta_k|D_k) = \frac{P(\theta_k|D_{k-1})P(d_k|\theta_k D_{k-1})}{\sum_i P(\theta_k^i|D_{k-1})P(d_k|\theta_k^i D_{k-1})}. \tag{2.19}$$

De agora em diante, chamaremos $P(\theta_k|D_{k-1})$ de **predição** da distribuição no instante k, a partir de informações em $(k-1)$. É neste ponto que iremos inserir a dinâmica do sistema. Se assumir-mos que o sistema é **Markoviano**, isto é, o estado atual pode ser determinado apenas pelo estado anterior, podemos usar a técnica da marginalização para obter $P(\theta_k|D_{k-1})$ a partir de $P(\theta_{k-1}|D_{k-1})$ e uma função de transição de estados simples.

$$P(\theta_k|D_{k-1}) = \sum_i P(\theta_k\theta^i_{k-1}|D_{k-1}) \tag{2.20}$$

$$= \sum_i P(\theta_k|\theta^i_{k-1}D_{k-1})P(\theta^i_{k-1}|D_{k-1}) \tag{2.21}$$

$$= \sum_i P(\theta_k|\theta^i_{k-1})P(\theta^i_{k-1}|D_{k-1}). \tag{2.22}$$

A última passagem deve-se à suposição Markoviana. $P(\theta_k|\theta_{k-1})$ é conhecida como **função de transição de estado** e é definida pela dinâmica do sistema.

O filtro Bayesiano é uma solução teoricamente simples, mas infelizmente sua aplicação direta é trabalhosa para conjuntos densos de hipóteses. O filtro Kalman [38] é uma solução exata para o problema, mas é limitado a distribuições parametrizadas por Gaussianas e funções de transição de estado lineares. Para distribuições paramétricas em geral, ou funções não linearizáveis, a aplicação do filtro Bayesiano tende a gerar integrais intratáveis algebricamente. Uma solução numérica simples, como adotar uma grade no espaço de hipóteses e calcular valores da distribuição apenas nestes pontos, é ineficiente. Uma solução melhor para o problema é apresentada a seguir.

2.6 O Filtro de Partículas

O filtro de partículas é uma solução aproximada para o problema de estimação Bayesiana. Tem a propriedade de concentrar o processamento em regiões de alta probabilidade e pode ser aplicado em problemas modelados com funções não lineares e distribuições arbitrárias.

Neste filtro, as distribuições de probabilidade são representadas (figura 2.1) por um conjunto de partículas $X_k = \{x^{(1)}_k, \ldots, x^{(N)}_k\}$. Cada partícula $x^{(i)}_k \in \mathbb{R}^d$, representa, na iteração k, uma hipótese e possui um peso associado $w^{(i)}_k$, de forma que $\sum_i w^{(i)}_k = 1$. Intuitivamente, quando todos $w^{(i)}_k$ são iguais entre si, a probabilidade de uma região do espaço de hipóteses é dada pela concentração de partículas. A probabilidade de qualquer hipótese pode ser recuperada através de uma técnica de estimativa de densidade como a **janela de Parzen** [17]. De acordo com a técnica,

$$P\left(y|X_k\right) = \sum_{i=1}^{N} w^{(i)}_k \ \frac{1}{\ell^d} \ \varphi\left(\frac{y - x^{(i)}_k}{\ell}\right), \tag{2.23}$$

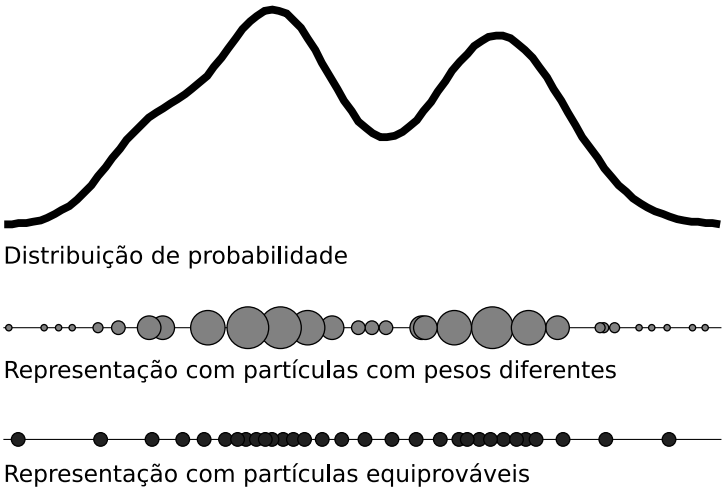


Figura 2.1: Representação de uma distribuição de probabilidades através de partículas. A distribuição acima pode ser representada por partículas com peso (centro) ou sem peso (abaixo), neste último caso a concentração de partículas é o único indicador da probabilidade. A distribuição original pode ser recuperada aplicando a técnica da janela de Parzen sobre as partículas.

onde ℓ é a largura da janela (aresta do hipercubo d-dimensional), φ é uma função *kernel* e y é uma hipótese qualquer do espaço de hipóteses. Para que a distribuição resultante seja válida, isto é,

$$P\left(y|X_k\right) \geq 0 \text{ e } \sum_{i=1}^{N} P\left(y_i|X_k\right) = 1, \tag{2.24}$$

a função *kernel* usada deve ser uma função de distribuição de probabilidade também. A Gaussiana é uma escolha comum, em geral,

$$\varphi(u) = \frac{1}{(2\pi)^{d/2}} e^{-u/2}. \tag{2.25}$$

A escolha do valor da largura ℓ da janela depende da aplicação. O algoritmo trivial para cálculo da densidade (probabilidades das partículas apenas) pela janela de Parzen é $O(N^2)$, mas pode ser aproximado usando algoritmos sobre árvores-kd duais para $O(N \log N)$ [25].

O filtro atualiza a distribuição de partículas em três etapas: **predição**, **atualização** e **reconfiguração**. Iniciando o sistema com uma distribuição $P(x_{k-1}|D_{k-1})$ formada de partículas com mesmo peso, convenientemente $w_{k-1}^i = 1/N$, as etapas são como se segue:

(1) Na etapa de predição, cada partícula $x_{k-1}^{(i)}$ de $P(x_{k-1}|D_{k-1})$ é movida, de forma independente, para uma nova posição no espaço de estado $x_k^{(i)}$ com probabilidade

$P(x_k^{(i)}, x_{k-1}^{(i)})$ dada pela função de transição do sistema. Em outras palavras, cada partícula passa pela função do sistema $x_k^{(i)} = f_{k-1}(x_{k-1}^{(i)}, \mu_{k-1})$, onde $f_k : \mathbb{R}^d \times \mathbb{R}^m \to \mathbb{R}^d$ e μ_{k-1} representa um ruído branco[11] de média zero, usado para gerar pequenas perturbações e evitar o colapso de partículas[12]. Desta forma, determinamos a distribuição *a priori* aproximada $P(x_k|D_{k-1})$, equivalente a necessária para a aplicação do teorema de Bayes.

(2) Na etapa de atualização, cada partícula da distribuição *a priori* tem seu peso atualizado com o dado adquirido no momento k, de forma que

$$w_k^{(i)} = w_{k-1}^{(i)} \frac{P(d_k|x_k^{(i)} D_{k-1})}{\sum_j P(d_k|x_k^{(j)} D_{k-1})}. \tag{2.26}$$

O efeito dos pesos é tornar mais prováveis as partículas em regiões que correspondem com as observações e, como conseqüência, formar a distribuição *a posteriori* $P(x_k|D_k)$ [52]. A observação, geralmente, é uma entidade completamente diferente das hipóteses. Portanto, precisamos de uma função que gere observações simuladas a partir de hipóteses. Para isto usamos a função $d_k^{(i)} = h_k(x_k^{(i)}, \nu_k)$, onde $h_k : \mathbb{R}^d \times \mathbb{R}^r \to \mathbb{R}^p$ e ν_k simula um ruído branco de média zero que modela o erro de observação. Desta forma é possível comparar a observação d_k e a observação provável $d_k^{(i)}$ gerada através de $x_k^{(i)}$ na função de verossimilhança $P(d_k|x_k^{(i)} D_{k-1})$.

(3) A etapa de reconfiguração visa aumentar a eficiência do filtro convertendo a distribuição *a posteriori*, encontrada na etapa anterior, para uma equivalente, mas com pesos iguais. Para isto, N partículas de $P(x_k|D_k)$ são copiadas com probabilidade proporcional a $w_k^{(i)}$ para uma nova distribuição. As partículas são copiadas com peso associado $w_k^{(i)} = 1$. Partículas originalmente com maior peso tenderão a ser copiadas mais vezes. Em contrapartida, partículas com menor peso tenderão a ser descartadas. Com a etapa de reconfiguração, podemos simplificar a equação (2.26) para

$$w_k^{(i)} = P(d_k|x_k^{(i)} D_{k-1}) \tag{2.27}$$

e a distribuição final ainda será válida. Outra vantagem é que as partículas ficam mais concentradas nas regiões de maior probabilidade, enquanto regiões de menor probabilidade têm suas partículas descartadas.

Esta é a versão do filtro conhecida como **Bootstrap** [24] e foi a que usamos neste trabalho. No entanto, uma extensão comum é usar uma **distribuição proposta** que

[11] Ruído branco é, em geral, representado por uma distribuição uniforme. Quando Gaussianas são usadas, chamamos de ruído branco normal.

[12] O colapso de partículas ocorre quando todo o conjunto se concentra (colapsa) em uma única hipótese. Levando em conta a etapa (3), o colapso ocorre relativamente rápido se as partículas não são perturbadas.

leve em conta o dado mais recente no lugar de $P(x_k, x_{k-1})$, que é apenas uma previsão. A natureza desta extensão varia muito de aplicação em aplicação e apresentamos alguns exemplos na seção 3.4.

O algoritmo é fácil de ser implementado (seção A.1) mas possui alguns inconvenientes. Entre eles está a chamada **maldição da dimensionalidade**. Aplicações que precisam ser modeladas com espaços de hipóteses de alta dimensão, sofrem com o aumento exponencial no número de partículas necessário para aproximar adequadamente a distribuição *a posteriori*. No próximo capítulo apresentamos uma forma de atenuar o problema aproveitando a estrutura da função de observação.

Capítulo 3

Filtro de Partículas com Hierarquia de Subespaços

Neste capítulo, apresentamos o Filtro de Partículas com Hierarquia de Subespaços, uma nova variedade de filtro de partículas. O novo filtro é composto por um conjunto de filtros de partículas especializados em determinadas regiões do espaço de parâmetros e organizados na forma de um grafo acíclico. Desenvolvemos uma estratégia para dividir o espaço de parâmetros em subespaços, de acordo com a estrutura implícita na função de observação de modelos hierárquicos, e para associá-los aos filtros de partículas especializados. Com o rastreamento segmentado proposto por este método, melhoramos a convergência geral e permitimos uma redução significativa no número de partículas.

3.1 Aspectos Gerais e Possíveis Topologias

O **Filtro de Partículas com Hierarquia de Subespaços**, ou SHPF, é estruturado na forma de um grafo acíclico. Os filtros especializados, que o formam, possuem número de partículas e amplitude de busca próprios. A **amplitude de busca** é uma forma de definir uma região de busca limitada no espaço de parâmetros a partir da restrição do movimento das partículas. Todos os filtros compartilham as mesmas funções de observação e verossimilhança. A função de transição de estado varia apenas nos parâmetros que pode estimar. Esta informação também é dada pelas amplitudes de busca. Amplitude zero para um parâmetro indica que este parâmetro não é rastreado, e conseqüentemente, não será modificado pela função de transição de um determinado filtro especializado. Em contrapartida, podemos dizer que a um filtro é atribuído um parâmetro se a função de transição de estado do filtro modifica este parâmetro, ou seja, o valor da amplitude é positivo. Idealmente, a maioria dos filtros deve estimar apenas alguns poucos parâmetros para que o número de partículas seja o menor possível. A tabela 3.1 resume os parâmetros

globais e locais da configuração de cada filtro especializado.

Configuração dos Filtros Especializados	
Individual	Coletiva
Número de partículas	Função de observação
Amplitudes de busca	Função de verossimilhança
Função de transição de estado	

Tabela 3.1: Configurações globais e locais dos filtros especializados.

As vezes é necessário introduzir alguns filtros especiais, chamados de agregadores. Filtros agregadores são responsáveis por uma busca geral, mas limitada, e são usados principalmente para suavizar distribuições que foram geradas a partir da combinação de duas ou mais distribuições anteriores. Esta situação ocorre quando um conjunto de filtros converge para um único ponto no grafo, como veremos adiante. Os filtros agregadores têm atribuídos mais parâmetros que os demais filtros especializados, mas com amplitude de busca limitada. Isto os torna capazes de explorar apenas uma pequena região do espaço de parâmetros, normalmente não o suficiente para realizar a estimação completa. Por este motivo, exigem consideravelmente menos partículas que um filtro projetado para realizar o rastreamento completo.

Existem duas formas básicas de organizar os filtros especializados. Eles podem ser colocados de forma a realizar a estimação em série ou em paralelo. Estas duas formas de relacionamento podem ser usadas para construção de topologias mais complexas. O estimador serial é similar a trabalhos anteriores [35]. Entretanto, na nossa proposta, podemos acrescentar o filtro agregador como o último elemento do grafo para tratar casos em que a **suposição de decomposição** (equação 3.2) não é totalmente aplicável (seção 3.4), *i.e.* quando os parâmetros são levemente correlacionados. Esta topologia básica é ilustrada na figura 3.1. Nesta figura, a cada filtro de **A** a **E** é atribuído um subespaço do espaço de parâmetros, respectivamente, $\mathbf{x}^A$ a $\mathbf{x}^E$. O filtro agregador **F** cobre todo o espaço, mas sua amplitude de busca é limitada. O filtro **A** é alimentado com a distribuição anterior e a distribuição posterior é obtida, no final do processamento, através do filtro **F**. A transmissão de partículas de um filtro **A** para o outro **B** é feita simplesmente copiando o número de partículas necessário da distribuição posterior do filtro **A**, durante a etapa de reconfiguração, para formar a distribuição anterior do filtro **B**.

Como dissemos, nosso grafo pode ter ramos que são estimados em paralelo (Figura 3.2). Ramos surgem na função de observação, h_k, de modelos hierárquicos, quando alguns elementos do vetor de observação, $d = h(\mathbf{x})$, não são influenciados pelo mesmo conjunto de parâmetros, $\mathbf{x}$, *i.e.*, $\{\exists \mathbf{x}_\xi \neq \mathbf{x}_\psi, i \neq j | h_i(\mathbf{x}_\xi) = h_i(\mathbf{x}_\psi) \wedge h_i(\mathbf{x}_\xi) \neq h_j(\mathbf{x}_\psi)\}$. Cada ramo resultará no valor de um elemento do vetor de observação. Estas distribuições estimadas

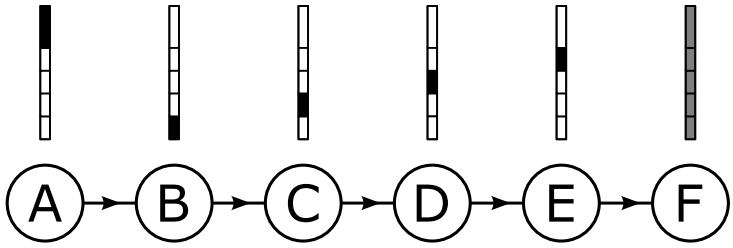


Figura 3.1: Exemplo de estimação serial. O filtro agregador foi acrescentado para fazer um ajuste final nos casos em que a suposição de MacCormick (equação 3.2) não é válida. As regiões escuras nas colunas acima dos filtros, correspondem àqueles parâmetros que são estimados por cada filtro, da esquerda para a direita: $\mathbf{x}^A$, $\mathbf{x}^B$, $\mathbf{x}^C$, $\mathbf{x}^D$ e $\mathbf{x}^E$. A região acinzentada representa a amplitude de estimação limitada.

em paralelo precisam ser unidas em uma única distribuição para descrever o estado global do sistema. Esta operação não é tão imediata. Se unirmos igualmente o subespaço completo de cada distribuição paralela, a distribuição resultante provavelmente será viesada em favor da distribuição inicial. Isto se deve ao fato de que a tendência é atribuir determinados parâmetros a um filtro apenas. Os demais, que não rastreiam estes parâmetros, terão estes subespaços herdados da distribuição inicial. O efeito é um comprometimento severo na amplitude de busca do filtro.

Um problema, relativo à aproximação por partículas, é que o rastreamento de um subespaço pode influenciar na distribuição de subespaços ortogonais. As partículas, em um dado filtro especializado, podem ser movidas apenas nas direções correspondentes aos parâmetros que o filtro estima. Na etapa de reconfiguração, serão priorizadas aquelas partículas com maiores probabilidades, dadas pela função de verossimilhança. Como conseqüência, as distribuições herdadas dos filtros antecessores provavelmente serão modificadas. No pior caso, cada filtro paralelo introduzirá uma hipótese incompatível, que não necessariamente descreverá o estado do sistema, e nem mesmo a média simples entre elas.

O comportamento descrito acima é ilustrado na Figura 3.3, na qual mostramos uma distribuição de partículas altamente correlacionada nas proximidades de uma observação de distribuição elipsoidal. Os parâmetros correspondentes ao eixo Z foram estimados anteriormente, resultando na figura à esquerda. Note que o único movimento possível[1] das partículas neste momento é paralelo ao eixo Z. Estimamos então independentemente os parâmetros que correspondem aos eixos X e Y. O grafo à direita mostra a projeção

[1]Supondo que os parâmetros representados pelos eixos X e Y não são estimados pelo filtro que tem o eixo Z atribuído.

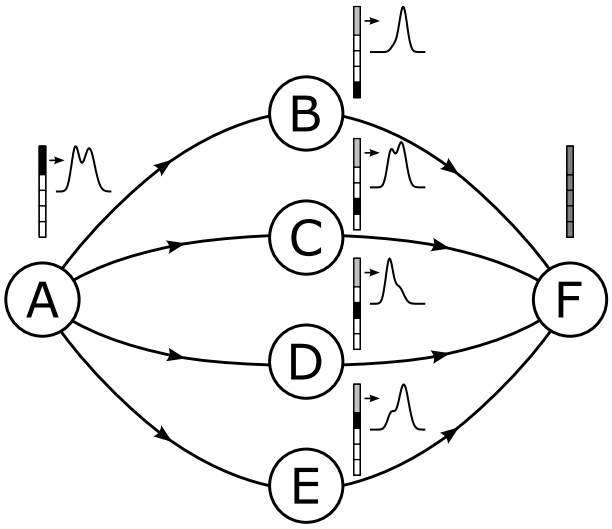


Figura 3.2: Exemplo de estimação paralela. O subespaço de parâmetros atribuídos a cada filtro é ilustrado com as regiões em preto. As distribuições dos subespaços herdados dos filtros antecedentes (cinza claro) provavelmente serão diferentes entre os filtros convergentes. A combinação final deve ser condicionada a todos estes filtros. A região cinza escuro no filtro **F** representa sua amplitude de estimação limitada.

das partículas e da distribuição da observação em cada plano. A estimação sobre X pode apenas mover as partículas paralelamente ao eixo X e, de forma semelhante, a estimação sobe Y pode apenas mover as partículas paralelamente ao eixo Y. Usando como guia a projeção sobre o plano XY, e recordando que a estimação é feita em cada filtro com a mesma função de verossimilhança, podemos facilmente chegar as distribuições desenhadas ao longo do eixo Z. Estas representam a distribuição do subespaço do parâmetro Z resultante de cada estimação independente dos parâmetros representados por X e Y. Claramente as distribuições são incompatíveis no sentido que cobrem regiões com pouca ou nenhuma intersecção do espaço de parâmetros. Este é um caso extremo que indica que parâmetros muito correlacionados devem ser estimados ao mesmo tempo. O mesmo efeito pode ocorrer, numa escala tolerável para estimação em paralelo, quando parâmetros que influenciam ramos diferentes da função de observação são levemente dependentes através das distribuições dos parâmetros que os antecedem na cadeia cinemática, como ilustrado na figura 3.2. O método usado para combinar as distribuições tem que considerar esta possibilidade.

A solução que demos ao problema de combinar as distribuições estimadas em paralelo foi empregar a operação de cruzamento, típica de algoritmos genéticos, para construir a nova distribuição posterior. Na figura 3.2, podemos descrever as subregiões de

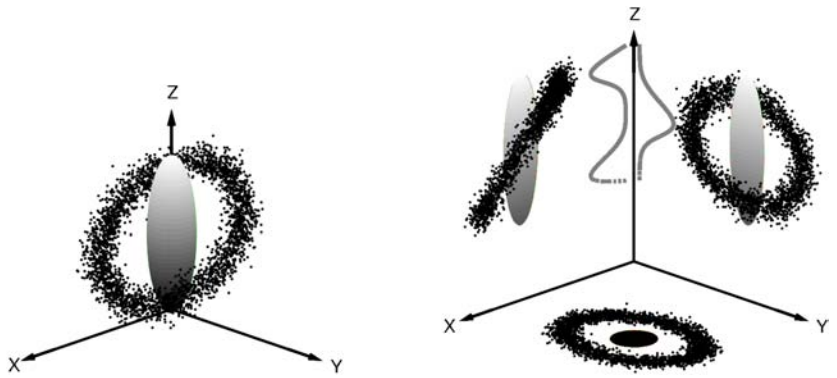


Figura 3.3: A distribuições inconsistentes surgem quando parâmetros correlacionados são estimados separadamente. Na figura existem 3.000 partículas nas proximidades de uma observação com distribuição elipsoidal. O eixo Z representa o distribuição do filtro antecessor. Os eixos X e Y representam os parâmetros sendo estimados no momento. As curvas desenhadas representam a distribuição de Z após a estimação de cada uma.

parâmetros de nossa partícula como $\mathbf{x} = \{\mathbf{x}^A, \mathbf{x}^B, \mathbf{x}^C, \mathbf{x}^D, \mathbf{x}^E\}$. A operação de cruzamento de múltiplos pais irá selecionar subregiões de cada filtro convergente, de **B** até **E**, para montar as partículas da distribuição posterior. A operação deve tender a selecionar de cada filtro, apenas as subregiões mais atualizadas. Uma forma de determinar quais subregiões serão selecionadas de cada filtro, e em que quantidades, será apresentada na próxima seção.

A operação de cruzamento pode gerar um conjunto inconsistente de partículas. Para tratar isto, fazemos de **F** nosso filtro agregador. Ele irá rejeitar partículas que não são compatíveis com a observação e irá realizar um ajuste final. O resultado será uma distribuição posterior consistente.

Durante a execução do SHPF, os filtros não entram na fila de execução, para fazer sua estimação, até que tenha sua distribuição inicial completamente montada pelos filtros antecessores. Os algoritmos que formam o Filtro de Partículas com Hierarquia de Subespaço podem ser encontrados na seção A.2 dos apêndices.

3.2 Determinando o que Cada Filtro Deve Propagar

Nesta seção apresentamos uma forma automática para determinar que subregiões das partículas que cada filtro especializado deve propagar. Mostraremos como usar o vetor de amplitudes, o número de partículas de cada filtro e a topologia para gerar uma estrutura contendo esta informação.

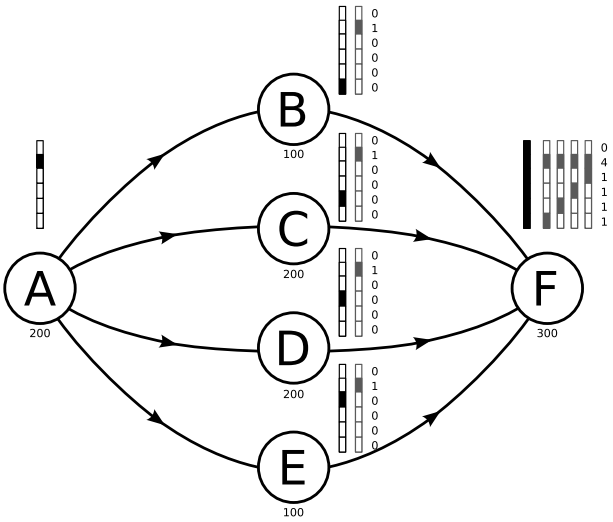


Figura 3.4: Máscaras de amplitude são propagadas do filtro inicial **A** até o filtro final **F**. Os vetores de ocupação são formados contando a quantidade de cada região marcada presente nas máscaras. Abaixo de cada filtro está a quantidade de partículas que deve receber durante a execução do SHPF.

O procedimento é dividido em duas etapas. A primeira, ilustrada na figura 3.4, consiste em propagar as **máscaras de amplitude** e gerar os **vetores de ocupação**. A máscara de amplitude é uma máscara binária gerada a partir do vetor de amplitudes de forma a assumir o valor um para regiões do vetor que são diferentes de zero e zero para as demais. Estas máscaras são propagadas por todos os filtros do grafo, sendo que, um filtro só propaga sua máscara para os demais após receber as máscaras de todos os antecessores. O vetor de ocupação grava a quantidade das regiões marcadas pelas máscaras de amplitude recebidas. Quando o filtro recebe uma máscara de seu antecessor, ele atualiza o vetor de ocupação, incrementando o valor do vetor equivalente a todas as regiões marcadas, e então combina a máscara recebida com a sua própria. Após receber as máscaras de seus antecessores, o filtro tem sua máscara completa, contendo todas as regiões que ele estima e todas as regiões estimadas por seus antecessores, ou seja, herdadas.

A segunda etapa, consiste em usar as máscaras de amplitude, o vetor de ocupação $Occ_{(\rho,j)}$, o número de antecessores $\Pi_{(\rho)}$ e o número de partículas $n_{(\rho)}$ de cada filtro de forma a determinar a proporção de cada subregião que deverá enviar durante a operação de cruzamento. Esta etapa pode ser feita filtro a filtro, sem nenhuma ordem específica. Caso o filtro estime ou herde uma região $\mathbf{x}^j$ (sua máscara estará com esta região marcada) ele propagará $n_{(\rho)}/Occ_{(\rho,j)}$ destas regiões, onde ρ indica o filtro adjacente para o qual a subregião será enviada. Caso o valor para determinada subregião do vetor de ocupação

do filtro adjacente seja zero, implicando que nenhum filtro antecessor é responsável por estimá-la, o número destas subregiões a serem enviadas é $n_{(\rho)}/\Pi_{(\rho)}$. O filtro não envia nenhuma quantidade das demais subregiões. A exemplo do filtro **B** na figura 3.4, as quantidades de regiões, olhando a máscara de amplitude de cima para baixo, são: $n_{(\mathbf{F})}/\Pi_{(\mathbf{F},1)} = 75$, $n_{(\mathbf{F})}/Occ_{(\mathbf{F},2)} = 75$, 0, 0, 0, $n_{(\mathbf{F})}/Occ_{(\mathbf{F},6)} = 300$.

3.3 Encontrando Subespaços

Nesta seção, introduzimos um procedimento para separar os parâmetros em grupos que formarão os filtros especializados e, ao mesmo tempo, uma topologia proposta. Uma possibilidade, comumente encontrada em trabalhos de redes neurais e controle [3], é usar a matriz Jacobiana, $\mathrm{J}(\mathbf{X}) = \frac{\partial h_i(\mathbf{X})}{\partial \mathbf{x}^j}$, da função de observação h_k. Se o elemento $\mathbf{j}_{ij}$ da matriz é sempre zero, não importando a entrada, significa que o parâmetro $\mathbf{x}^j$ não afeta o elemento h_i do vetor de observação. Queremos agrupar os parâmetros que atuam sobre o mesmo conjunto de elementos da observação e, com isto, ter uma idéia grosseira de que grupos podem ser estimados separadamente. Quando a estrutura do modelo é clara, podemos imediatamente extrair os grupos, mas em situações onde o modelo é obtido através de treinamento, ou é demasiadamente complexo, um algoritmo é necessário.

Usamos a matriz Jacobiana para gerar uma nova matriz binária, que chamaremos de **matriz de observalidade**, cujos elementos têm valor um se são diferentes de zero para algum valor do vetor de parâmetros **X**. Caso contrário, o valor zero é atribuído. Quando geramos a função h_k manualmente, a matriz de observalidade pode ser obtida por simples inspeção.

Para extrair os grupos automaticamente, associamos a cada coluna da matriz um par de valores: a contagem de uns na coluna, $\mathbf{o}_j$, que representa o número de elementos da observação que cada parâmetros influencia, e o número formado pelo padrão binário da coluna, $\mathbf{v}_j$. Quando as colunas são ordenadas por $\mathbf{o}_j$, e por $\mathbf{v}_j$ como chave secundária, a matriz de observalidade resultante vai ter estrutura similar a

$$\begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \tag{3.1}$$

cujas colunas com mesmo $\mathbf{v}_j$ pertencem ao mesmo grupo. Neste exemplo, nós temos, da esquerda para a direita, os grupos {63}, {12}, {3}, {16}, {8}, {2}, rotulados de acordo com $\mathbf{v}_j$.

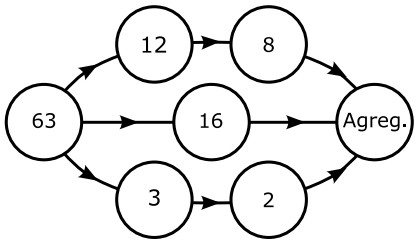


Figura 3.5: Agrupamentos de parâmetros sugerido pelo algoritmo e a topologia extraída da função de observação h_k.

Para decidir quais grupos podem ser estimados em paralelo, fazemos uma simples comparação binária entre os $\mathbf{v}_j$'s. Aqueles grupos que não têm bits em comum, ou seja, ramos de funções de observação sobrepostos, são candidatos a serem estimados em paralelo. A ordem da estimação serial pode ser tida como a de $\mathbf{o}_j$ decrescente. A figura 3.5 ilustra as relações extraídas da matriz exemplo anterior. Notamos este resultado é apenas uma sugestão, tendo em vista que grupos sem sobreposição podem ainda ser altamente correlacionados através dos filtros antecedentes. É tarefa do projetista do filtro escolher uma das possíveis topologias. Além das topologias, o projetista deve escolher o número de partículas e definir as amplitudes de busca da função de transição de estado em cada filtro. Acreditamos que estes procedimentos podem ser automatizados através de técnicas de aprendizado [40] e temos planos de investigar esta possibilidade em trabalhos futuros.

3.4 Trabalhos Relacionados

Nesta seção expomos os trabalhos relacionados a rastreamento visual de objetos. Material sobre técnicas que usam abordagens muito diferentes da nossa pode ser encontrado no trabalho de Pavlovic [43], no de Huang e Wu [62] e no, mais recente, trabalho de Stenger [53].

Sistemas lineares com distribuição de observações e estado Gaussianas, podem ser rastreados por uma fórmula fechada, é o caso do filtro de Kalman [38]. Sistemas que são modelados com dinâmica não linear mas cujo estado pode ser modelado por uma curva Gaussiana, vêem sendo estimados com sucesso através do Unscented Kalman Filter [59]. Em muitas outras aplicações de visão computacional, a distribuição dos valores possíveis para um dado parâmetro assume formas arbitrárias, exigindo um ferramental mais poderoso para modelá-las.

O filtro de partículas é uma ferramenta para estimação não paramétrica que vem sendo usada extensivamente nas comunidades de visão [41, 12] e aprendizado de máquina [15, 30].

É conhecido por muitos nomes como **Condensation** [28], filtro **Bootstrap** [24] e **Monte Carlo Seqüencial** [16]. Yukito Iba [27] faz uma coletânea de métodos similares ao filtro de partículas que vêm sendo sistematicamente redescobertos em diversas áreas. Como sabemos, entretanto, o filtro sofre com o problema da dimensionalidade, isto é, a necessidade de partículas para rastrear adequadamente um modelo cresce exponencialmente com o número de parâmetros que se deseja rastrear.

MacCormick e Isard [35, 36] introduziram o método de **amostragem particionada** para rastreamento de múltiplos objetos, que pode ser estendido para objetos articulados. O objetivo é evitar o custo exponencial em partículas para aplicações com muitas dimensões. Eles dividiram em partições o espaço de amostragem assumindo que é possível decompor a função de transição de estado da seguinte forma:

$$f(\mathbf{x}_k|\mathbf{x}_{k-2}) = \int f_B(\mathbf{x}_k|\mathbf{x}_{k-1}) f_A(\mathbf{x}_{k-1}|\mathbf{x}_{k-2}) d\mathbf{x}_{k-1}. \tag{3.2}$$

Isto é, é possível aplicá-las em série sobre as partículas. f_A e f_B são funções de transição de estado referentes a dois objetos diferentes. O método deles é similar ao nosso para topologias seriais, com a diferença que realizam a etapa de atualização (seção 2.6) no último filtro apenas. Eles propuseram também a **amostragem particionada ramificada** para melhorar a precisão do filtro, que, diferentemente da nossa proposta, calcula para cada ramo (em ordem diferente) todas as funções de transição de estado. As distribuições de cada ramo são simplesmente unidas antes da etapa de atualização. Ilustram seu método através de aplicações para rastreamento de pessoas e de mãos sem marcadores usando ***active contours***. Neste caso, não rastreiam um modelo tridimensional da mão, no lugar, rastreiam os parâmetros do contorno. Logo, suas demonstrações são limitadas ao rastreamento bidimensional.

Okuma *et al.* [41] descrevem o ***boosted particle filter***, também para rastreamento de múltiplos alvos. O filtro de partículas comum funciona mal quando múltiplos alvos válidos são observados. O filtro tende a preferir uma **moda**[2] sobre as demais, concentrando as partículas apenas nela, e com isto, eventualmente, irá perder o rastreamento dos outros objetos de interesse. A abordagem deles consiste em unir uma mistura de filtros de partícula, proposta por Vermaak *et al.* [56] com o algoritmo de Viola e Jones [57] para detecção de objetos. A mistura de filtros associa um filtro de partículas para cada moda. Os filtros interagem apenas na avaliação da função de verossimilhança. Como a etapa de reconfiguração é independente, a abordagem evita o problema de perda das partículas

[2]Moda é o ponto com maior valor de probabilidade da distribuição. Distribuições com apenas uma moda, como a Gaussiana, são chamadas de unimodais. Algumas distribuições apresentam mais de um destes pontos e são chamadas de multimodais. Geralmente consideramos distribuições com picos múltiplos de probabilidade com magnitude semelhante como sendo multimodais, mesmo que apenas um destes tenha o maior valor de probabilidade.

de uma moda com menor probabilidade. O algoritmo de Viola e Jones emprega uma cascata de classificadores fracos, treinados para minimizar falsos negativos, para gradualmente eliminar regiões da imagem que não correspondem ao objeto de interesse. Os classificadores são aplicados em ordem de complexidade, isto faz com que apenas poucas regiões sejam avaliadas pelos classificadores mais complexos e torna o algoritmo muito eficiente. O algoritmo é usado para gerar a distribuição proposta do filtro de partículas e para gerenciar as modas. Os autores mostram resultados interessantes no rastreamento de jogadores de hóquei no gelo. Planejamos investigar a aplicação do algoritmo de Viola e Jones para detecção de partes da mão.

Deutscher e Reid [13] apresentaram o ***annealed particle filter***. O método usa uma série de funções de peso que gradualmente restringem a região de busca para regiões de maior probabilidade do espaço de hipóteses. O método inicia a busca em uma região abrangente do espaço e, ao final do processo, mantém partículas nos picos de probabilidade de regiões, possivelmente, separadas. Eles defendem que o método realiza uma divisão do espaço de hipóteses automaticamente. Introduziram também a operação de cruzamento a fim de combinar as melhores partes das diferentes regiões, e, desta forma, gerar uma nova distribuição que, mais eficientemente, combina o rastreamento realizado em paralelo dos subespaços separados. Este método emprega o cruzamento tradicional de pares de partículas. A nossa proposta utiliza o cruzamento de múltiplas partículas. Fazemos isto porque nossa partição do espaço é discreta e conhecemos certas características das distribuições por construção. O cruzamento, no entanto, tem o mesmo objetivo. Não temos dados comparativos da performance dos algoritmos, mas imaginamos que a operação de *annealing* completa, necessária para cada iteração do filtro, seja onerosa. Os autores ilustraram o método rastreando uma pessoa, sem marcadores, em três dimensões, com até 34 graus de liberdade, usando múltiplas câmeras e fundo de cena sem objetos que distraiam o rastreador. Como observação, usaram cantos e silhuetas.

Em sua tese de doutorado, Stenger [53] propõe uma técnica de busca hierárquica para o rastreamento. O método hierárquico proposto, ao contrário do método por partículas, é determinístico. Inicia com uma grade de baixa resolução e gradualmente divide regiões da grade que apresentam maior probabilidade, rejeitando as demais regiões. O método tem a vantagem de permitir a inicialização automática, entretanto, depende de padrões gerados previamente e, portanto, sofre com a discretização do espaço de hipóteses. O autor defende que a técnica pode ser usada em conjunto com o filtro de partículas, de forma a fornecer a distribuição proposta, ao invés de usar a obtida a partir de uma função de transição de estados. A implementação para rastreamento da mão apresentada é baseada na forma da mão extraída pelo contorno. A função de verossimilhança testa os padrões via ***chamfer***. Na correspondência *chamfer*, o padrão pesquisado é emparelhado com a transformada de distância dos contornos da imagem. Com esta configuração, ele conseguiu rastrear a mão

sobre fundos de cena bem menos restritos que trabalhos anteriores. Entretanto só foram demonstrados movimentos com certa de seis graus de liberdade no máximo.

Existem formas de rastrear a mão na literatura que empregam paradigmas diferentes do filtro Bayesiano. Em sua dissertação de mestrado, Dorner [14] usa otimização através do método de Newton para determinar qual a pose da mão que melhor se adequa à observação. Ela usou uma luva pintada, com 20 marcadores individuais, para capturar movimentos com 26 graus de liberdade. No entanto, apenas a luva, e eventualmente uma mão, é enquadrada na imagem e o fundo da cena é construído de forma a ser o mais neutro possível. Este método é mais suscetível a mínimos locais que o método Bayesiano empregado aqui, principalmente no momento de inicialização do sistema. O filtro de partículas (seção 2.6) mantém, naturalmente, partículas em regiões prováveis, o que possibilita que o sistema considere diferentes hipóteses até que surja informação que beneficie uma sobre a outra. O método de otimização pelo método de Newton, e similares, apenas transfere a informação do instante anterior por meio de uma sugestão para o ponto inicial de busca para o próximo quadro. O sistema apresentado aqui tenta rastrear 15 graus de liberdade com seis marcadores idênticos. Precisamos de toda a informação disponível da distribuição anterior para evitar que o sistema assuma configurações imprevisíveis a cada quadro.

Capítulo 4

Implementação, Validação e Alguns Experimentos

Neste capítulo, descrevemos o sistema de rastreamento desenvolvido no decorrer do mestrado e no qual o Filtro de Partículas com Hierarquia de Subespaços foi inserido, bem como a forma como os dados foram adquiridos. Apresentamos os resultados obtidos com o nosso filtro usando topologias diferentes e o comparamos ao filtro de partículas tradicional.

4.1 Sistema de Rastreamento

Desenvolvemos um sistema de rastreamento que recebe como entrada uma seqüência de imagens, ou dados sintéticos, e tem como saída as medianas das distribuições dos parâmetros do modelo da mão. O resultado é apresentado de forma gráfica, com o modelo estimado sobreposto à imagem original. A figura 4.1 mostra uma tela do sistema. Internamente, o sistema pode ser dividido como descrito no capítulo 1, entretanto, a última etapa, a de reconhecimento, não foi usada na maioria dos experimentos. Nas seções 4.3 e 4.4, descrevemos as duas etapas iniciais do sistema. Na seção 4.8, descrevemos um experimento com um sistema de reconhecimento completo, minimamente implementado.

4.2 Detalhes de Implementação

O sistema foi completamente implementado usando C++ [54], GTK+ [60] e OpenCV [4], de forma que é possível executá-lo em qualquer plataforma suportada por esta linguagem e bibliotecas. Todas as ferramentas podem ser executadas em modo visual interativo, ou em lote pela linha de comando. Nos primeiros meses de desenvolvimento, tentamos usar o DirectShow [46] para extrair e processar os vídeos. Dada a desnecessária comple-

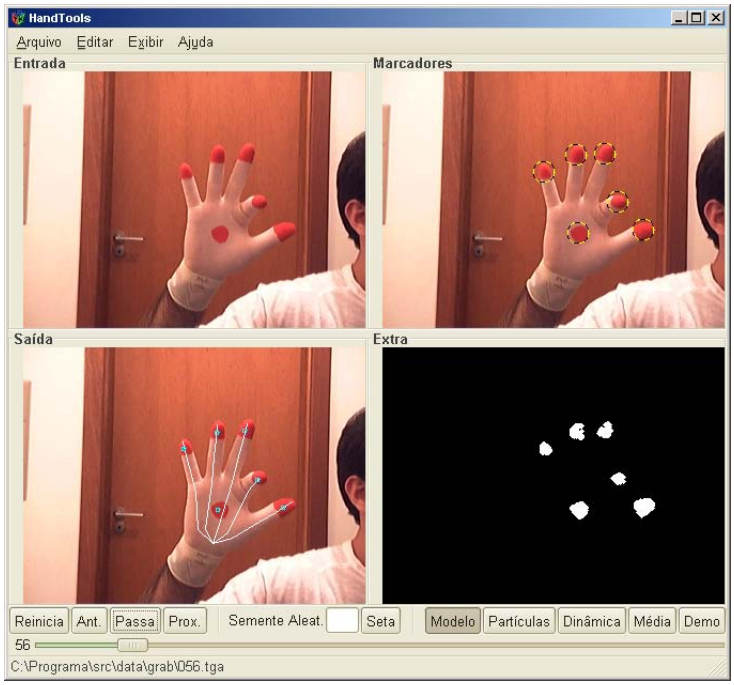


Figura 4.1: Uma tela do sistema de rastreamento implementado no decorrer deste trabalho. O quadro "Extra" mostra as regiões segmentadas pela etapa de extração de características. O quadro "Marcadores" mostra a posição dos marcadores determinada a partir das regiões segmentadas. O quadro "Saída" ilustra os parâmetros estimados desenhando o modelo sobre a imagem original.

xidade e inerente restrição a um tipo de sistema operacional, migramos para o GTK+. A troca acelerou o desenvolvimento e diversificou a base suportada. O OpenCV facilitou a construção posterior do gerador de seqüências sintetizadas, facilitou a adaptação do rastreador para estas seqüências e possibilitou o uso portável de vídeo extraído de arquivos ou diretamente da câmera. Embora tenhamos usado esta biblioteca exclusivamente para entrada e saída, ela possui muitos algoritmos úteis para visão computacional e está cada vez mais confiável.

4.3 Etapa de Extração de Características

A primeira etapa a ser realizada é a extração de características[1] dos quadros do vídeo. Estas características serão usadas como observações no filtro implementado na segunda

[1]Quando usamos dados sintéticos, esta etapa é desnecessária uma vez que o gerador de seqüências sintéticas foi desenvolvido com foco no rastreamento e já fornece as características pré-processadas.

etapa. Escolhemos como características, as posições, em coordenadas de imagem[2], das pontas dos dedos e do centro da palma. Para tornar a extração de características mais simples, possibilitando que focássemos o trabalho na segunda etapa, usamos uma luva com marcadores. Os detalhes do ambiente de aquisição e da luva são apresentados na seção 4.5.

Precisamos extrair a posição dos seis marcadores vermelhos. Consideramos a posição de um marcador como sendo seu centro de massa, dado pelo primeiro momento das posições dos *pixels* que o formam. Existem alguns inconvenientes na escolha desta cor, por exemplo, os lábios e as orelhas podem apresentar matiz suficientemente vermelha para serem confundidos por marcadores. Devido a isto, consideramos um marcador apenas regiões com pontos vermelhos adjacentes com área acima de um determinado patamar Λ. Levando estes aspectos em consideração, e após muita experimentação, construímos um extrator de características que realiza, em cada quadro, os seguintes passos:

- Inicialmente normalizamos a imagem. Regiões com valores das componentes RGB abaixo de um patamar pré-estabelecido são convertidas para zero a fim de evitar instabilidade. Para cada *pixel*, o vetor $\hat{P}_i$, formado pelas três componentes de cor, é normalizado de forma a ter módulo um. Fazemos isto para diminuir a influência da intensidade da cor e privilegiar a matiz e saturação.

- Com a imagem normalizada, procuramos vetores com direção próxima a de um vetor $\hat{R}$ que codifica a cor com matiz e saturação desejada. A comparação é feita por uma subtração simples de vetores. Caso $|\hat{R} - \hat{P}_i| < \varepsilon$, sento ε um patamar de similaridade pré-determinado, a cor é considerada a mesma. No final do processo uma máscara é gerada indicando quais *pixels* possuem a cor desejada.

- A máscara gerada pelo processo anterior é suscetível a ruído no processo de aquisição do vídeo. O efeito disto é que *pixels* com a cor desejada podem deixar de ser marcados, e *pixels* com cores diferentes podem acabar sendo marcados. Aplicamos operações morfológicas [23] de erosão e dilatação à máscara a fim de uniformizá-la. Em ambas usamos o elemento cruz.

- Uma vez que temos a máscara uniformizada, buscamos as regiões formadas por *pixels* com adjacência-8 [23] e cujo número de total *pixels* seja superior a Λ, pré-estabelecido. Para isto usamos uma estrutura de conjuntos disjuntos [9]. A imagem é percorrida de cima a baixo, do canto esquerdo ao direito, linha a linha. A cada novo ponto marcado encontrado, verificamos se os *pixels* imediatamente à esquerda, acima e na diagonal superior-esquerda estão marcados. Caso afirmativo seus conjuntos

[2]Coordenadas de imagem representam a distância a um referencial, geralmente o canto superior esquerdo da imagem, em número de *pixels*.

são unidos. Obtemos então uma lista de regiões marcadas disjuntas. No fim do processamento, basta calcular o centro de massa das regiões cuja quantidade de *pixels* é superior a Λ e temos as posições dos marcadores.

Temos alguns comentários sobre algumas técnicas que testamos e dispensamos no decorrer do desenvolvimento do método descrito acima. A aplicação de um filtro Gaussiano, no intuito de suavizar a imagem, minimizando o efeito do ruído [34], é comumente o primeiro tratamento dado a imagem em aplicações de visão. Nos nossos experimentos, o uso do filtro não resultou em melhoras expressivas a ponto de compensar o tempo de processamento gasto. Devido a isto, dispensamos o filtro em favor de operações morfológicas que se mostraram mais adequadas ao nosso problema. Testamos a conversão para HSV [21] para minimizar a influência da iluminação, mas não obtivemos bons resultados nos testes feitos antes de obter a câmera definitiva. Uma vez que encontramos uma alternativa, não voltamos atrás para verificar se seria mais eficiente. Além disto, o método de normalização descrito acima gera uma imagem única que codifica a saturação e matiz. Isto nos permitiu visualizar de forma mais simples regiões de ruído da imagem, o que facilitou a escolha definitiva da cor e da câmera a serem usadas.

Tentamos otimizar as duas primeiras operações desta etapa usando um cubo RGB [44] que dispensa a normalização e comparação ao usar a cor do *pixel* diretamente como índice de uma tabela pré-processada. O ganho em performance conseguido foi desprezível, a qualidade da máscara obtida, ligeiramente inferior, (usamos um cubo com $2^{(7+7+7)}$ posições, ao invés do completo com $2^{(8+8+8)}$) e o custo, em memória, relativamente alto.

Ao final desta etapa, ilustrada na figura 4.2, obtemos as observações necessárias para alimentar o filtro da etapa de estimativa de parâmetros.

4.4 Etapa de Estimativa de Parâmetros

Esta etapa se resume ao filtro de partículas. Implementamos tanto o tradicional quanto o Filtro de Partículas com Hierarquia de Subespaços. Já determinamos nossa observação, precisamos descrever o modelo[3], a função de transição de estado, a função de observação, a distribuição inicial e a fase de ajuste. Descreveremos todos estes itens nesta seção. As topologias usadas nos experimentos são descritas na sessão 4.6.

O modelo que escolhemos tem quinze **graus de liberdade** (DoF), cinco para a posição tridimensional do pulso, ângulos[4] de **giro** e **guinada**, e dois para os ângulos de **inclinação**

[3]Na cronologia real o modelo e as observações foram decididos em conjunto.

[4]O efeito do ângulo de giro é o de supinação e pronação, ou seja, a rotação da mão em relação ao eixo do antebraço. Com o antebraço ereto, de forma que a mão esteja acima do cotovelo e com a palma na direção do observador, o ângulo de guinada está associado ao movimento de *tchau*, se feito a partir do pulso, e o ângulo de inclinação tem o efeito de colocar a palma para baixo, aproximando as pontas dos

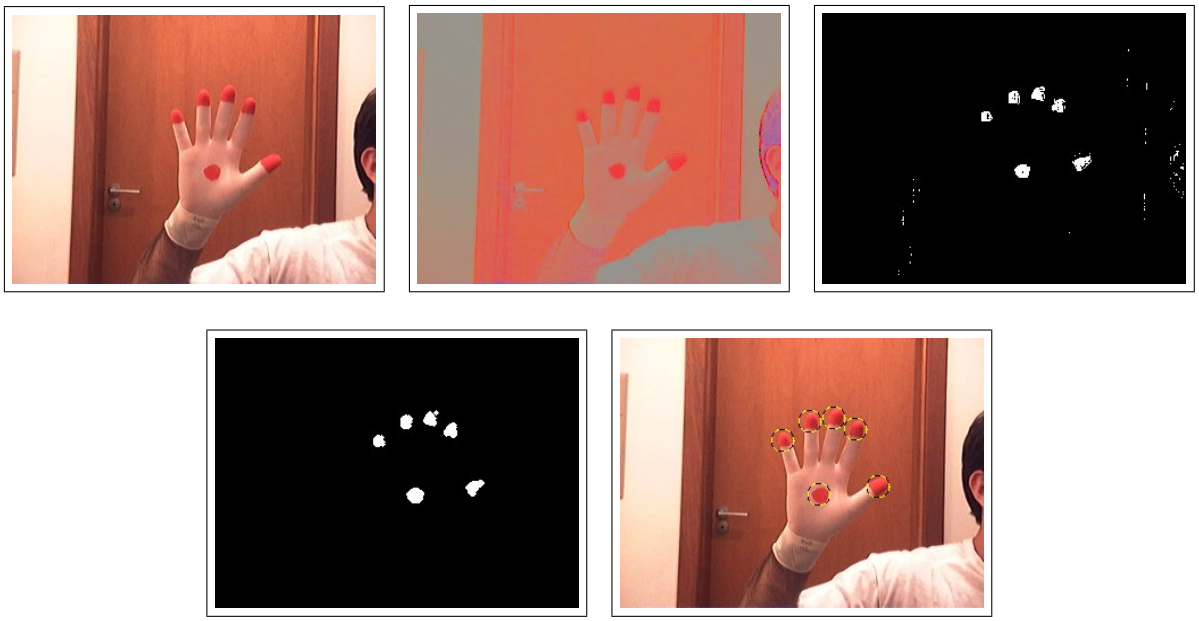


Figura 4.2: Etapa de extração de características. Da esquerda para direita, na linha de cima temos: a imagem original, a imagem normalizada, a máscara de cor. Na linha de baixo temos: a máscara após operações morfológicas e a imagem com centro das regiões marcados.

e guinada de cada dedo. No nosso modelo, as observações são as posições de seis marcadores não rotulados, um no centro da palma e um na ponta de cada dedo. O modelo possui algumas restrições quanto aos limites de movimento dos dedos, implementadas de forma que poses que não obedecem as restrições são associadas a probabilidade zero. A figura 4.3 ilustra o modelo. Esta configuração é claramente mal condicionada, já que, com as seis posições, podemos apenas determinar unicamente doze parâmetros. Para tratar parcialmente o problema, alguns parâmetros são associados a amplitudes de busca muito limitadas, apenas o suficiente para lidar com o **ajuste inicial** e mudanças pequenas durante o rastreamento. Ainda assim, é possível que os valores estimados formem poses da mão consideravelmente diferentes dos dados de entrada. Apesar disto, ainda podemos realizar experimentos interessantes com uma configuração tão simples.

Precisamos definir as funções de transição de estado e de observação. No nosso sistema, empregamos uma função de transição de estado que trabalha apenas por dispersão. As tentativas de usar modelos com dinâmica de primeira ordem resultaram em oscilações indesejáveis. Felizmente, o modelo por dispersão se mostrou adequado para nossa aplicação.

dedos na direção dos olhos, e para cima, afastando.

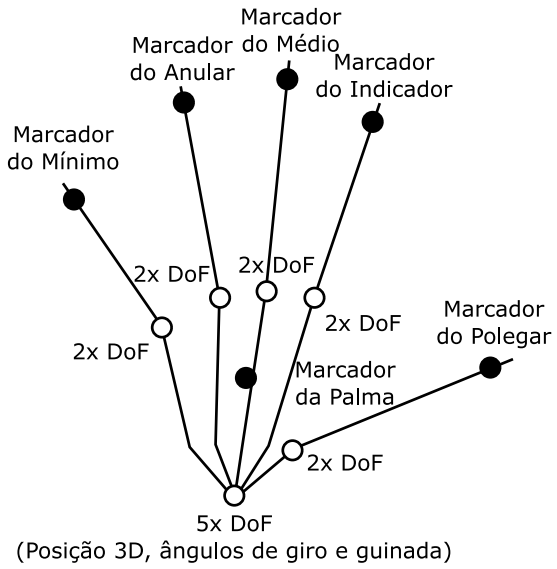


Figura 4.3: Um modelo simples da mão com quinze graus de liberdade. Os círculos preenchidos representam a localização relativa das observações, os círculos vazios mostram onde os parâmetros atuam no modelo.

O controle da dispersão é dado pelos limites impostos pelo modelo, para evitar disposições impossíveis (em casos normais) da mão e pelas amplitudes definidas em cada filtro especializado do filtro por hierarquia de subespaços.

A função de observação gera, a partir dos valores dos parâmetros em cada partícula, a projeção das posições, em coordenadas de imagem, dos marcadores. Para gerar os pesos das partículas precisamos comparar as duas observações. Para isto, os marcadores gerados e os adquiridos são emparelhadas através de um método guloso. Se o número de marcadores adquiridos é menor ou igual a seis, para cada marcador adquirido, o algoritmo de emparelhamento busca qual marcador gerado está mais próximo, excluindo da busca os já emparelhados. Se o número de marcadores adquiridos é maior que seis, a busca se inverte e é feita a partir do conjunto gerado. Desta forma podemos lidar com situações com observações degeneradas.

Precisamos de uma distribuição inicial para o filtro e uma forma de adaptá-la para as características iniciais de cada seqüência de vídeo. Iniciamos com a suposição que a mão, no primeiro momento, está espalmada, com a palma na direção da câmera e no centro da imagem. Esta pose tem a vantagem de manter as marcas distantes uma das outras.

A posição inicial não é crítica mas, infelizmente, a alta ambiguidade das nossas observações impede a realização de uma busca mais abrangente, de forma robusta, para permitir uma pose inicial mais genérica. Temos que fixar os valores dos parâmetros dos

ângulos dos dedos e procurar uma distribuição adequada para o ajuste inicial apenas nos parâmetros de posição e rotação, que dificilmente serão as mesmas entre duas seqüências quaisquer. A distribuição que maximiza a entropia, e gera médias com as características esperadas, é a distribuição uniforme[5]. Teoricamente é a distribuição mais adequada, mas podemos usar uma alternativa, que funcionou adequadamente nos nossos experimentos. Geramos um conjunto de partículas exatamente iguais, todas correspondentes a mão centrada e espalmada, em seguida aplicamos pequenas variações aleatórias na posição e usamos esta distribuição como a nossa distribuição inicial.

Decidida a disposição inicial, precisamos ajustar o filtro à seqüência. O rastreamento é sempre iniciado com a mão espalmada na direção da câmera. É deixada assim por alguns instantes. Durante este tempo, o filtro com hierarquia de subespaços opera apenas nos filtros especializados que correspondem aos parâmetros de posição e rotação da mão, isto é, transformações rígidas. O filtro de partículas tradicional, de forma semelhante, pode entrar em um modo de operação alternativo, rastreando apenas estes parâmetros. No nosso sistema mal condicionado e com observações ambíguas, o ajuste inicial é fundamental. Sem limitar os parâmetros que são estimados quando as partículas estão concentradas em uma região de baixa probabilidade, o filtro pode assumir estados completamente imprevisíveis e nunca se recuperar. Isto ocorre mesmo que a observação real esteja simplesmente transladada em relação as observações sintetizadas. Este ajuste pode ser realizado sempre que a função de verossimilhança retornar valores abaixo de um patamar pré-estabelecido.

4.5 Configuração para Aquisição de Dados Reais

Nesta seção, apresentamos alguns aspectos práticos em relação a configuração do ambiente e equipamentos para a aquisição das seqüências de vídeo. Em particular, damos detalhes das características da luva, câmeras usadas e iluminação.

O primeiro ponto a discutir é o ambiente disponível para aquisição dos dados. Como não tínhamos um estúdio disponível, ou lugar para montar um, toda a aquisição era realizada em ambientes cuja modificação era limitada. Tínhamos que trabalhar com as condições de fundo de cena disponíveis, isto é, a parede logo em frente à câmera, que por sua vez, está presa ao computador.

Como não podíamos usar o fundo de cena com um tom uniforme, que facilitaria a extração de características, e ainda queríamos tornar esta etapa simples, decidimos usar uma luva marcada. Usamos uma luva de médico comum, encontrada em farmácias, marcada com tinta guache. Ambos os componentes são baratos e encontrados facilmente.

[5]No caso geral, usar apenas a restrição mínima e a restrição de média define uma distribuição exponencial, mas como estamos considerando que a média é exatamente o centro do espaço de parâmetros, o multiplicador de Lagrange associado à média vai a zero, e o resultado é a distribuição uniforme.

Durante o decorrer do trabalho, testamos padrões de cores diferentes, mas o que possibilitou a segmentação mais precisa foi o que as pontas dos dedos e o centro da palma são marcados com a mesma tinta vermelha. Foi a matiz para a qual a câmera apresentou menos ruído e que gerou menos confusão com outros objetos na cena. Infelizmente o uso de marcadores com mesma cor e de observações apenas baseadas em cores tornam o sistema pouco robusto a oclusão. Portanto, a disposição dos marcadores foi decidida de forma que fiquem consideravelmente distantes um dos outros na maioria dos experimentos e que correspondam a características aparentemente simples de extrair de uma mão sem luva. Implementar a primeira etapa de forma a dispensar a luva é uma das propostas para trabalhos futuros.

Toda câmera apresenta variação temporal aleatória na cor dos *pixels*. Esta variação é normalmente chamada de, e modelada como, ruído. Ao longo do trabalho conseguimos obter três câmeras: uma *Camcorder* VHS da **JVC**, uma *webcam* **SamSung AnyCam MPC-M10** e uma *webcam* **Logitech QuickCam PRO 3000**. A QuickCam foi a única a fornecer imagem com nível de ruído baixo o suficiente, nas nossas condições de iluminação, para efetuar a extração características de forma estável. Infelizmente, foi também a última câmera a ser obtida. Todos os dados reais, mostrados ou citados neste trabalho, foram adquiridos através dela.

A iluminação é um fator prático importante. Se não fosse mantida fixa, teríamos que recalibrar o extrator para cada nova seqüência. A iluminação também afeta o nível de ruído das câmeras. Pouca luz é, em geral, acompanhada de muito ruído. A única alteração possível no nosso ambiente era a escolha das lâmpadas. Testamos florescentes, halógenas e incandescentes, todas sem refletores. A configuração que forneceu as imagens com menor ruído foi a que usamos duas lâmpadas incandescentes de $60W$ que, coincidentemente, são as de menor custo. Todos os experimentos foram realizados à noite, uma vez que a variação do clima e da posição do sol causam variação na iluminação.

4.6 Validação com Dados Sintéticos

Para validação, geramos seqüências sintéticas envolvendo rotação tridimensional da mão, movimento de agarrar, translação e movimento individual dos dedos. Usamos três topologias diferentes, como ilustrado na figura 4.4, e diferentes amplitudes de busca. Comparamos os resultados com o filtro de partículas tradicional com três quantidades diferentes de partículas: 1.000, 5.000 e 10.000. Temos uma topologia serial e uma paralela do SHPF, cada uma com 1.000 partículas. Como nossa configuração tem alguns graus de liberdade livres, que podem induzir a desvios imprevisíveis, é impraticável comparar **estados escondidos**, isto é, os valores dos parâmetros de dois filtros quaisquer ou entre um filtro e os valores verdadeiros. Ao invés disto, comparamos o quão próxima está a observação

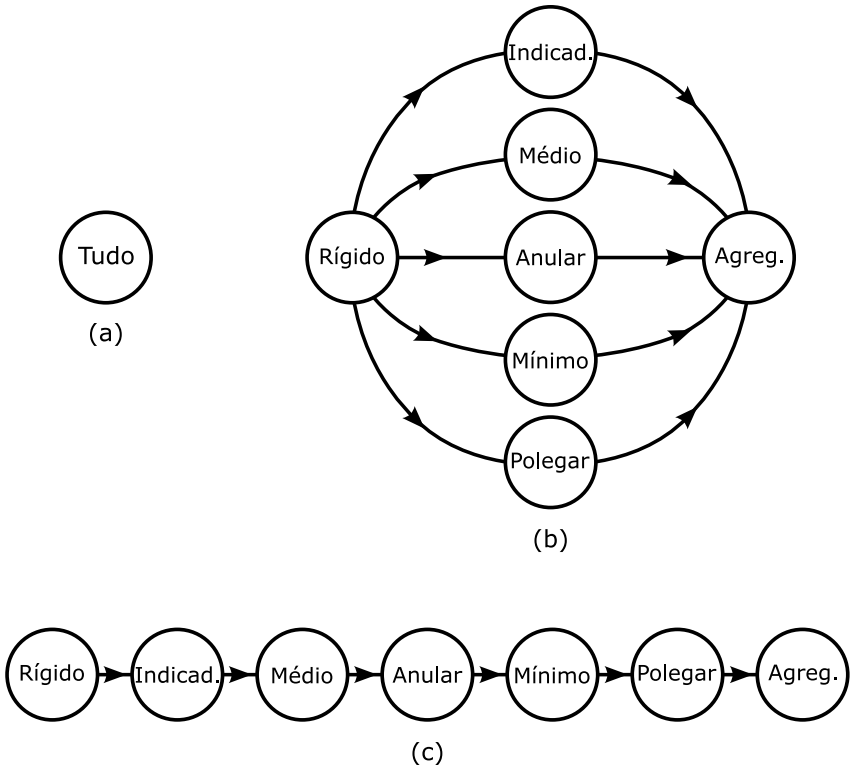


Figura 4.4: Topologias diferentes do filtro com hierarquia de parâmetros. A figura (a) ilustra o filtro de partículas padrão estimando todos os parâmetros ao mesmo tempo. (b) e (c) são as topologias serial e paralela. Cada filtro, a menos do agregador, é rotulado de acordo com o conjunto de parâmetros do modelo que estima.

gerada a partir das medianas estimadas pelo filtro do valor gerado sinteticamente. Todas as seqüências possuem movimento relativamente rápido, por exemplo, o movimento de "agarrar e soltar", em algumas seqüências, é feito em cerca de um segundo de vídeo.

A figura 4.5 mostra as curvas da soma das diferenças quadradas (SSD) entre as observações geradas por cada filtro e as observações fornecidas pela seqüência sintetizada. Ambos relatam o erro decorrido durante o período de duas operações de "agarrar e girar". No apêndice B, apresentamos mais gráficos de erro, bem como imagens ilustrando as seqüências sintetizadas. Como esperado, o erro do filtro com hierarquia de subespaços foi, na maior parte do tempo, inferior ao do filtro tradicional para o mesmo número de partículas. Em algumas situações, o SHPF, convergiu melhor para os dados de entrada que o filtro tradicional com uma ordem de magnitude a mais de partículas.

A topologia paralela apresentou uma resposta mais estável que a serial para o mesmo número de partículas. Isto é razoável, uma vez que a operação de cruzamento força uma

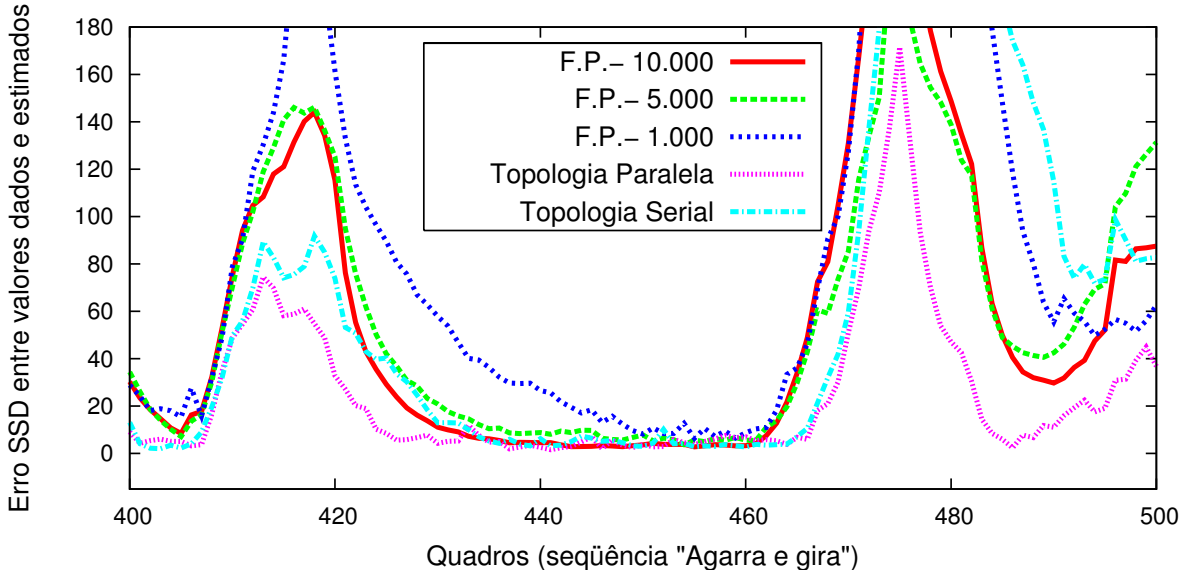


Figura 4.5: Gráficos SSD entre as observações sintetizadas e cinco topologias do filtro diferentes. A seqüência contém duas operações "agarra e gira".

distribuição mais abrangente de partículas no espaço de parâmetros. Esta propriedade deixa o filtro paralelo menos suscetível a mínimos locais que o serial. A filtragem seqüencial, do caso serial, tende a contrair a distribuição a medida que passa por cada filtro especializado. A abordagem mais abrangente da topologia paralela é particularmente importante quando reduzimos o número de partículas. As topologias SHPF, em geral, requerem menos partículas que o filtro tradicional. Filtros com número maior de partículas tendem a oferecer uma estimativa por mediana mais estável visualmente, no sentido de sofrer menos com o efeito de "tremedeira" (variações aleatórias perceptíveis ao longo dos quadros). No entanto, uma estimativa final mais estável visualmente, não implica em um menor erro de rastreamento.

Os parâmetros dos filtros estão resumidos na tabela 4.1. A tabela 4.2 resume o erro médio obtido em cada seqüência sintética. A performance ruim do filtro serial na seqüência "Quatro", foi devida a uma divergência grande no rastreamento que se deu após o quadro 400. Neste caso, dois dedos da mão trocaram de posição. No geral, os filtros SHPF têm performance equiparável, como mostram as figuras do apêndice B. Em alguns dos resultados, não mostrados, onde testamos parâmetros diferentes, observamos uma performance ligeiramente superior do filtro serial e algumas performances melhores para o filtro tradicionais de 10.000 partículas. Entretanto, a maioria dos nossos testes foi condizente

com os dados mostrados no apêndice.

4.7 Experimentos com Dados Reais

Em seqüências de vídeo reais, as observações, algumas vezes, são inconsistentes. Marcadores podem sumir, aparecer em lugares indesejáveis, fundirem-se e duplicar-se. Seqüências reais apresentam também menos consistência quanto a pose e posição iniciais. A etapa de ajuste que implementamos mostrou-se adequada para lidar com posições tridimensionais e ângulos diferentes do esperado pela distribuição inicial. Infelizmente, quando dois ou mais marcadores estão muito próximos a ponto de fundirem-se, a correspondência que o modelo assume é aleatória, podendo acarretar na troca de marcadores entre os dedos, isto é, os dedos se cruzam. Por isto, tentamos manter os marcadores afastados nos experimentos.

A figura 4.6 mostra quatro quadros de uma seqüência real de quinze segundos. As duas imagens da esquerda mostram o processo de ajuste inicial. As duas imagens da esquerda são dois quadros de um movimento de "agarrar". No apêndice B, mostramos algumas seqüências com mais detalhes. Com nossa configuração, e modelo com quinze parâmetros, o SHPF com 1.000 partículas foi capaz de rastrear a 30 quadros por segundo, em um Pentium 4 3.2Ghz. O tempo de execução do filtro é aproximadamente proporcional ao número de partículas. Logo, nos nossos experimentos, conseguimos rastrear cerca de oito vezes mais rápido e com a mesma precisão que o filtro tradicional.

4.8 Sistema de Reconhecimento Visual de Gestos

Realizamos um pequeno experimento de manipulação de objetos virtuais. Para isto, implementamos minimamente a terceira etapa do sistema de reconhecimento, a homônima, de reconhecimento. Nossa implementação relata, ao sistema acoplado ao reconhecedor, quando a mão está aberta ou fechada, sua posição bidimensional e ângulo de guinada. O sistema acoplado consiste de uma área bidimensional com objetos virtuais que podem ser movidos ou girados. Definimos que a mão fecha quando o ângulo de inclinação do polegar, e de mais algum outro dedo da mão, é menor que um valor α, e que a mão abre, caso seja maior que um valor β, e fazemos $\alpha < \beta$. Caso haja algum objeto sobre o centro da palma quando a mão é fechada, este objeto passa ao estado de "agarrado" e pode ser movido e girado, de acordo com os movimentos da mão. Quando a mão abre, os objetos passam todos para o estado "solto".

A figura 4.7 ilustra o ambiente virtual em conjunto com o sistema de reconhecimento. Os dois quadrados cinza, que aparecem no quadro inferior direito, podem ser agarrados, movidos e girados. No momento de obtenção da imagem, o usuário está movendo, em

Figura 4.6: Quadros de uma seqüência real. Os dois da esquerda foram extraídos durante o passo de ajuste inicial. Os dois à direita demonstram um movimento de "agarrar". Os parâmetros recuperados são ilustrados com o modelo desenhado sobre a imagem original.

	Filtros							
Parâmetros	Rígido	Ind.	Méd.	Anular	Mín.	Poleg.	Agreg.	F.P. 10.000
X	4.0	0	0	0	0	0	1.0	4.0
Y	4.0	0	0	0	0	0	1.0	4.0
Z	0.04	0	0	0	0	0	0.019	0.04
Guinada	0.028	0	0	0	0	0	0.0	0.028
Giro	0.0208	0	0	0	0	0	0.0	0.0208
Ind. Guin.	0	0.04	0	0	0	0	0.019	0.04
Ind. Incl.	0	0.083	0	0	0	0	0.019	0.083
Médio Guin.	0	0	0.04	0	0	0	0.019	0.04
Médio Incl.	0	0	0.083	0	0	0	0.019	0.083
Anular Guin.	0	0	0	0.04	0	0	0.019	0.04
Anular Incl.	0	0	0	0.083	0	0	0.019	0.083
Mínimo Guin.	0	0	0	0	0.04	0	0.019	0.04
Mínimo Incl.	0	0	0	0	0.083	0	0.019	0.083
Polegar Guin.	0	0	0	0	0	0.04	0.019	0.04
Polegar Guin.	0	0	0	0	0	0.04	0.019	0.04
Partículas	200	100	100	100	100	100	300	10000
σ^2 da obs.	6.0	2.0	2.0	2.0	2.0	2.0	2.0	6.0

Tabela 4.1: Quantidade de partículas, vetores de amplitudes e variância da observação dos filtros que formam as topologias serial e paralela e do filtro tradicional.

		Filtros				
Seqüência	Quadros	Serial	Paralelo	F.P. 1.000	F.P. 5.000	F.P. 10.000
“Agarra e solta”	640	55.37	29.11	104.95	86.87	82.17
“Agarra e gira”	1232	27.16	23.03	85.69	58.17	58.04
“Indicador incl.”	217	4.93	4.57	15.21	11.89	12.02
“Indicador guin.”	272	5.74	5.38	32.86	23.57	21.66
“Translação”	87	4.80	5.34	19.30	16.41	16.02
“Quatro”	524	135.71	6.91	33.35	25.88	23.99

Tabela 4.2: Erro médio em várias seqüências sintéticas. Ambas topologias do SHPF estão usando 1000 partículas no total.

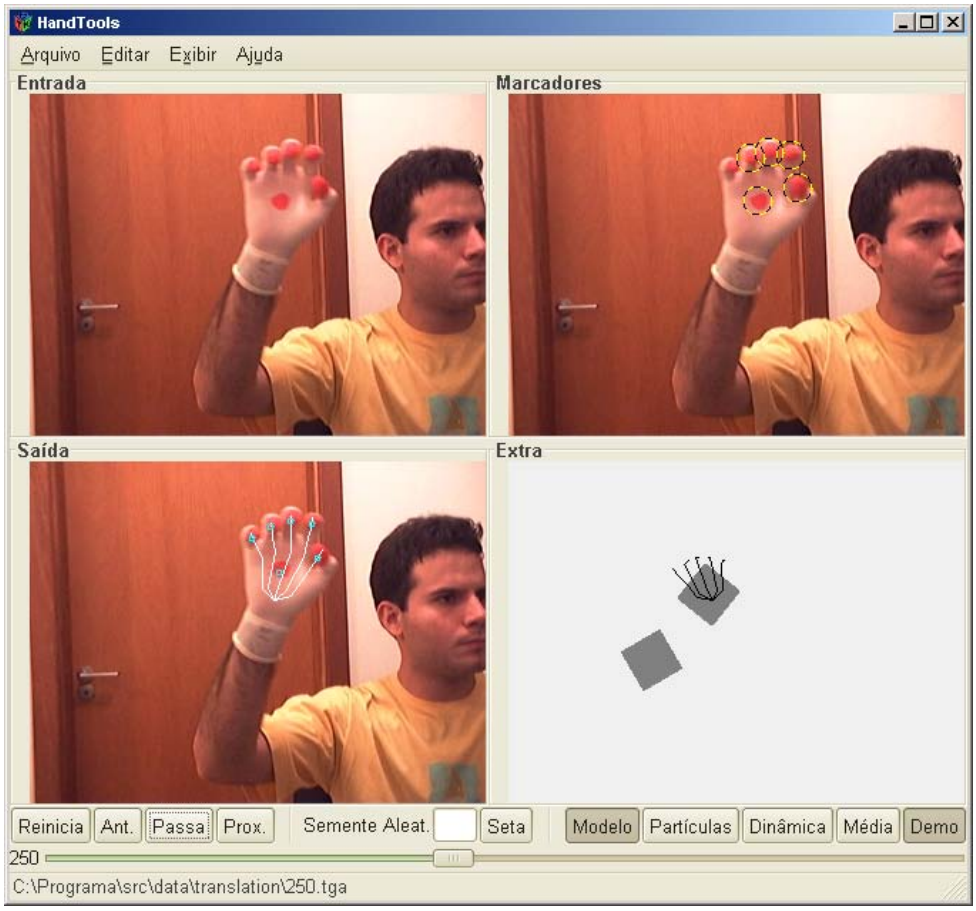


Figura 4.7: A figura ilustra uma pequena aplicação de manipulação de objetos virtuais feita para testar uma implementação simples da etapa de reconhecimento. Os quadrados do quadro "Extra" podem ser movidos e girados pelos usuários.

círculos, um dos quadrados. O experimento funcionou adequadamente dentro dos limites do sistema de reconhecimento. Expandir este sistema para suportar outros tipos de interação é uma das propostas para trabalhos futuros.

Capítulo 5

Conclusões e Trabalhos Futuros

Em algumas aplicações, a interface gestual visual é mais adequada que interfaces disponíveis atualmente em termos de custo ou em termos de praticidade. Porém, em outras, principalmente as que dependem de gestos manipulativos de longa duração, a viabilidade da interface gestual ainda é duvidosa. Os exemplos que demos no início deste trabalho suportam estas afirmações.

Uma solução exata para o rastreamento da mão é atualmente impraticável pela complexidade devido a auto-oclusão e a ambiguidade dos métodos atuais de extração de características. Optamos, então, por um método de estimativa de parâmetros Bayesiano. Este é teoricamente mais adequado para o problema que técnicas de otimização por permitir a passagem de mais informação entre quadros, e desta forma, ser mais robusto à ambiguidade.

A teoria da probabilidade Bayesiana é mais abrangente que a teoria freqüencista. Os principais fatores que adiaram sua aceitação no meio acadêmico foram, de certa forma, resolvidos. Esta interpretação da probabilidade permite a compreensão mais clara de alguns aspectos envolvidos no desenvolvimento deste trabalho, em particular, o tratamento das distribuições iniciais. Resta saber que benefícios trará para a análise posterior do filtro, área na qual a teoria freqüencista, e seus **processos estocásticos**, tem décadas de vantagem.

A principal contribuição deste trabalho foi a introdução do Filtro de Partículas com Hierarquia de Subespaços que apresenta convergência mais rápida que o filtro tradicional e, portanto, exige um número menor de partículas para rastrear modelos hierárquicos. Mostramos o funcionamento de uma topologia serial e uma paralela para o SHPF e discutimos suas características. Introduzimos a operação de cruzamento de múltiplos pais para combinar distribuições de probabilidade, a princípio, incompatíveis. O uso desta operação resultou em uma busca mais abrangente no espaço de estados e uma conseqüente melhora na convergência do filtro. Neste sentido, as topologias paralelas para o filtro são mais van-

tajosas que as seriais. Mostramos como uma estrutura de filtros de partículas na forma de um grafo acíclico orientado pode ser construída a partir do Jacobiano da função de observação do sistema.

Implementamos um ambiente de reconhecimento com foco no rastreamento da mão. O ambiente consiste de aplicações independentes: um rastreador para seqüências de vídeo, um rastreador para seqüências sintetizadas, um gerador de seqüências e um pequeno ambiente virtual. O ambiente é portável e pode ser feito disponível em todas as plataformas suportadas pelas bibliotecas que empregamos. A abordagem multi-plataforma acelerou a detecção de erros de implementação, tendo em vista que alguns erros não reportados em uma plataforma, podem causar falha instantânea em outra. O ambiente foi todo desenvolvido com bibliotecas de código aberto. Não só se mostraram mais práticas de usar que bibliotecas proprietárias, como sua documentação é mais abrangente e fácil de ser localizada.

Validamos, por meio de seqüências sintéticas, a melhora em termos de convergência em relação ao filtro de partículas tradicional. Realizamos experimentos com seqüências adquiridas com uma *webcam* para validar o sistema em aplicações de vídeo real. A principal característica destas aplicações é a ambiguidade das observações. O rastreamento se mostrou estável e testamos o funcionamento do sistema de reconhecimento minimamente implementado. Neste, fomos capazes de agarrar, mover e girar objetos virtuais.

Nosso sistema é limitado a gestos simples, tendo em vista que a aproximação de dois marcadores pode facilmente confundir o rastreador. Nossas observações são ambíguas e insuficientes para o rastreamento de todos os graus de liberdade da mão. Entretanto foi possível desenvolver um sistema que funciona a 30 quadros por segundo, é capaz de rastrear 15 graus de liberdade com seis marcadores e permite a manipulação de objetos virtuais.

5.1 Trabalhos Futuros

Como sugestões para extensões e trabalho futuros podemos citar:

- A construção de um modelo matemático para o SHPF para permitir a análise teórica dos limites de convergência, das melhorias em termos de número de partículas e das características das topologias.

- A re-implementação da primeira etapa do sistema de reconhecimento de forma a dispensar a luva. Embora imaginamos uma alternativa que resultará em observações semelhantes usando Viola e Jones, seria interessante validar o SHPF com observações de outros tipos, como contornos, que estão presentes na literatura. As duas melho-

rias principais que buscamos é o tratamento da oclusão e ser capaz de rastrear todos os graus de liberdade da mão.

- Uma abordagem de aprendizado de máquina pode ser empregada no SHPF para construir automaticamente a topologia, decidir o número de partículas e as amplitudes de busca em cada filtro.

- O uso de técnicas de reconfiguração dinâmica sobre o modelo do sistema, de forma a torná-lo mais robusto a mudanças das características da observação. É o caso, por exemplo, quando marcadores desaparecem inesperadamente devido a problemas de oclusão ou resolução da imagem.

- Expandir o sistema de reconhecimento e ambiente virtual para permitir outras formas de interação.

Apêndice A

Algoritmos

Neste apêndice, apresentamos a versão genérica dos algoritmos para o filtro de partículas tradicional e o filtro de partículas com hierarquia de subespaços.

A.1 O Filtro de Partículas Tradicional

Considere um vetor com as partículas P, uma estrutura com as observações O, o escalar $n = \textit{número\_de\_partículas}$ e um vetor com os pesos das partículas W de forma que

$$W_{(i)} = \sum_{j=1}^{i} w_j \quad \text{e} \quad w_n = 1. \tag{A.1}$$

As funções de transição e de verossimilhança devem ser definidas para cada aplicação. P deve possuir uma distribuição válida no início do processamento. P' corresponde à distribuição posterior do filtro. A busca binária empregada, é uma variação que retorna a primeira posição na qual o elemento poderia ser inserido sem alterar a ordem. $P_{(i)}$ simboliza a i-ésima partícula do vetor P.

Filtro_de_Partículas (P, O, P')
1. *Função_de_Transição (P)*
2. *Verossimilhança* (P, O, W)
3. *Reconfiguração* (P, W, P')

Reconfiguração (P, W, P')
1. **Para todo** i **de** 1 **até** *número_de_partículas***:**
2. $\quad r \Leftarrow$ *Uniforme* $(0, 1)$
3. $\quad k \Leftarrow$ *Busca_Binária* (W, r)
4. $\quad P'_{(i)} \Leftarrow P_{(k)}$

A.2 O Filtro de Partículas com Hierarquia de Subespaços

O algoritmo do SHPF é dividido em duas partes. Geramos a estrutura *TP*, de Transferência de Partículas, contendo as quantidades de todas as regiões que os filtros devem transportar para os adjacentes, de forma que $TP_{(\rho,\ell,k)}$ é a quantidade de regiões $\mathbf{x}^k$ que o filtro ρ deve enviar ao filtro ℓ. Criamos esta estrutura a partir da amplitude de busca de cada região, determinada pela matriz *Amplitude*, e a partir da topologia do grafo, determinada pela lista de adjacência, *Adj*, e o número de antecedentes de cada filtro, Π. No grafo, temos m filtros de partículas e o número de partículas de cada um é determinado pelo escalar $n_{(k)}$. Por conveniência, *nAdj* é um vetor que guarda o número de filtros adjacentes de cada filtro.

A função *Pré_Processamento* gera a estrutura *TP*. Para isto constrói uma máscara de amplitudes binária, *Masc*, a partir das amplitudes de busca, de forma a determinar quais regiões são rastreadas por cada filtro, e propaga as máscaras para os filtros adjacentes. O vetor de ocupação, *Occ*, determina a quantidade total de cada região que foram enviadas pelos filtros antecedentes e *MdP* armazena as máscaras propagadas pelos antecedentes. A partir destas máscaras é possível determinar que filtro tem prioridade de envio de uma certa região, e que regiões devem ser preenchidas por mais de um filtro.

Cada filtro ρ tem seu vetor partículas $P_{(\rho)}$ e seu vetor de pesos $W_{(\rho)}$, de forma que

$$W_{(\rho,i)} = \sum_{j=1}^{i} w_j \quad \text{e} \quad w_{n_{(\rho)}} = 1. \tag{A.2}$$

As observações são adquiridas na forma da estrutura O. D é uma matriz que indica o preenchimento de um conjunto de partículas, isto é, o valor de $\beta = D_{(\rho,j)}$ indica que β partículas do filtro ρ têm seu elemento $\mathbf{x}^j$ preenchido.

Um filtro só pode ser executado após seus antecedentes terem preenchido seu conjunto de partículas inicial. Na ocasião do último antecedente ter preenchido, o filtro é colocado na fila Q para ser executado. Durante a execução, as funções de transição e verossimilhança são avaliadas e, depois, a etapa de reconfiguração é feita para cada filtro adjacente, transferindo as regiões das partículas de acordo com *TP*.

Podemos usar *filtro_limite* para manter o SHPF na fase de ajuste, por exemplo, limitando o rastreamento nos primeiros estágios, que geralmente formam as transformações rígidas. Durante a operação normal do filtro, *filtro_limite* $= m$, assumindo que m coincide com o último filtro do grafo. O vetor de partículas do primeiro filtro, $P_{(1)}$, deve possuir uma distribuição válida no início do processamento. A distribuição final estará disponível no mesmo vetor $P_{(1)}$. Para simplificar a escrita do programa, consideramos que os valores lógicos assumem os valores numéricos 1 e 0, para *verdadeiro* e *falso*, respectivamente.

Filtro_de_Partículas_Parcial (P, W, O, ρ)

1. *Função_de_Transição* $(P_{(\rho)})$
2. *Verossimilhança* $(P_{(\rho)}, O, W_{(\rho)})$

Reconfiguração_Guiada $(P, W, TP, D, \rho, \ell)$

1. **Para todo** i **de** 1 **até** *Máx* $(TP_{(\rho,\ell)})$**:**
2. $\quad r \Leftarrow$ *Uniforme* $(0, 1)$
3. $\quad k \Leftarrow$ *Busca_Binária* $(W_{(\rho)}, r)$
4. $\quad$ **Para todo** j **de** 1 **até** *tamanho_da_partícula***:**
5. $\quad\quad$ **Se** $\{TP_{(\rho,\ell,j)} > 0\}$**:**
6. $\quad\quad\quad P_{(\ell,D_{(\ell,i)},j)} \Leftarrow P_{(\rho,k,j)}$
7. $\quad\quad\quad TP_{(\rho,\ell,j)} \Leftarrow TP_{(\rho,\ell,j)} - 1$
8. $\quad\quad\quad D_{(\ell,j)} \Leftarrow D_{(\ell,j)} + 1$

Filtro_de_Partículas_com_Hierarquia_de_Subespaços $(P, O,$ *Adj*, $TP,$ *filtro_limite*$)$

1. $D \Leftarrow 0$
2. $W \Leftarrow 0$
3. $Q \leftarrow \{1\}$
4. **Enquanto** $\{Q \neq$ **vazio**$\}$**:**
5. $\quad \rho \leftarrow Q$
6. $\quad$ *Filtro_de_Partículas_Parcial* (P, W, O, ρ)
7. $\quad$ **Para todo** ℓ **em** $Adj_{(\rho)}$**:**
8. $\quad\quad$ *Reconfiguração_Guiada* $(P, W, TP, D, \rho, \ell)$
9. $\quad\quad$ **Se** $\{\nexists\, \alpha \in D_{(\ell)} | \alpha \neq n_{(\ell)}\}$**:**
10. $\quad\quad\quad Q \leftarrow \{\ell\}$
11. *Reconfiguração_Guiada* $(P, W, FP, D,$ *filtro_limite*$, 1)$

Pré_Processamento (TP, Adj, $nAdj$, Π, $Amplitude$)

1. **Para todo** i **de** 1 **até** m**:**
2. $\quad \pi_{(i)} \Leftarrow \Pi_{(i)}$
3. $\quad$ **Para todo** k **de** 1 **até** *tamanho_da_partícula***:**
4. $\quad\quad Masc_{(i,k)} \Leftarrow \{Amplitude_{(i,k)} > 0\}$
5. $\quad\quad Occ_{(i,k)} \Leftarrow 0$

6. $Q \leftarrow \{1\}$
7. **Enquanto** $\{Q \neq$ **vazio**$\}$**:**
8. $\quad \rho \leftarrow Q$
9. $\quad$ **Para todo** ℓ **de** 1 **até** $\Pi_{(\rho)}$**:**
10. $\quad\quad$ **Para todo** k **de** 1 **até** *tamanho_da_partícula***:**
11. $\quad\quad\quad Masc_{(\rho,k)} \Leftarrow \{Masc_{(\rho,k)} \vee MdP_{(\rho,\ell,k)}\}$
12. $\quad\quad\quad Occ_{(\rho,k)} \Leftarrow Occ_{(\rho,k)} + MdP_{(\rho,\ell,k)}$
13. $\quad$ **Para todo** ℓ **em** $Adj_{(\rho)}$**:**
14. $\quad\quad MdP_{(\ell)} \Leftarrow Masc_{(\rho)}$
15. $\quad\quad \pi_{(\ell)} \Leftarrow \pi_{(\ell)} - 1$
16. $\quad\quad$ **Se** $\{\pi_{(\ell)} = 0\}$**:**
17. $\quad\quad\quad Q \leftarrow \{\ell\}$

18. **Para todo** i **de** 1 **até** m**:**
19. $\quad$ **Para todo** j **de** 1 **até** $nAdj_{(i)}$**:**
20. $\quad\quad$ **Para todo** k **de** 1 **até** *tamanho_da_partícula***:**
21. $\quad\quad\quad$ **Se** $\{Masc_{(i,k)} > 0\}$**:**
22. $\quad\quad\quad\quad TP_{(i,j,k)} \Leftarrow n_{(j)} / Occ_{(Adj_{(i,j)},k)}$
23. $\quad\quad\quad$ **Senão, se** $\{Occ_{(Adj_{(i,j)},k)} = 0\}$**:**
24. $\quad\quad\quad\quad TP_{(i,j,k)} \Leftarrow n_{(j)} / \Pi_{(j)}$
25. $\quad\quad\quad$ **Senão:**
26. $\quad\quad\quad\quad TP_{(i,j,k)} \Leftarrow 0$

27. **Para todo** k **de** 1 **até** *tamanho_da_partícula***:**
28. $\quad TP_{(m,1,k)} \Leftarrow n_{(1)}$

Apêndice B

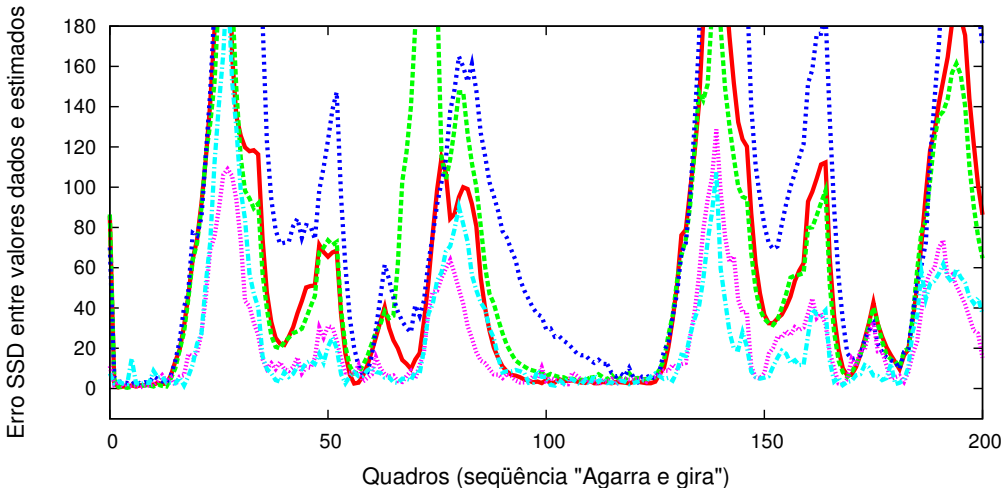
Resultados Experimentais Complementares

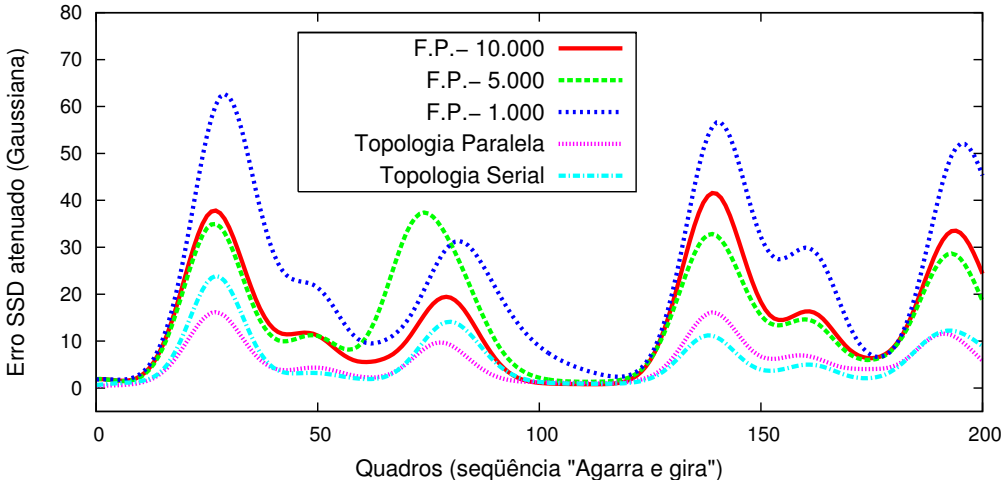
Neste apêndice, listamos mais gráficos comparativos entre os filtros e relacionamos algumas seqüências de vídeo citadas no trabalho.

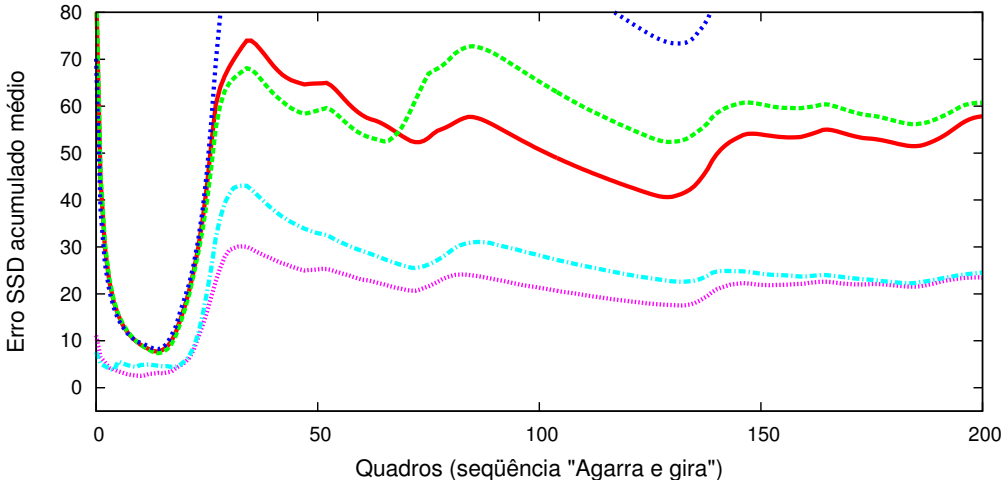
Para cada seqüência sintética mostramos três gráficos. O primeiro ilustra o erro SSD entre a posição estimada dos marcadores e a posição real no vídeo. O segundo é uma versão atenuada do primeiro através da aplicação de uma Gaussiana com $\sigma = 20$. Com este gráfico fica mais fácil perceber a performance local de cada filtro. O terceiro gráfico é uma média do erro SSD acumulado pelos quadros. Com este gráfico podemos perceber o quanto cada filtro acumula de erro ao longo do tempo, e podemos ter uma estimativa relativa de performance global.

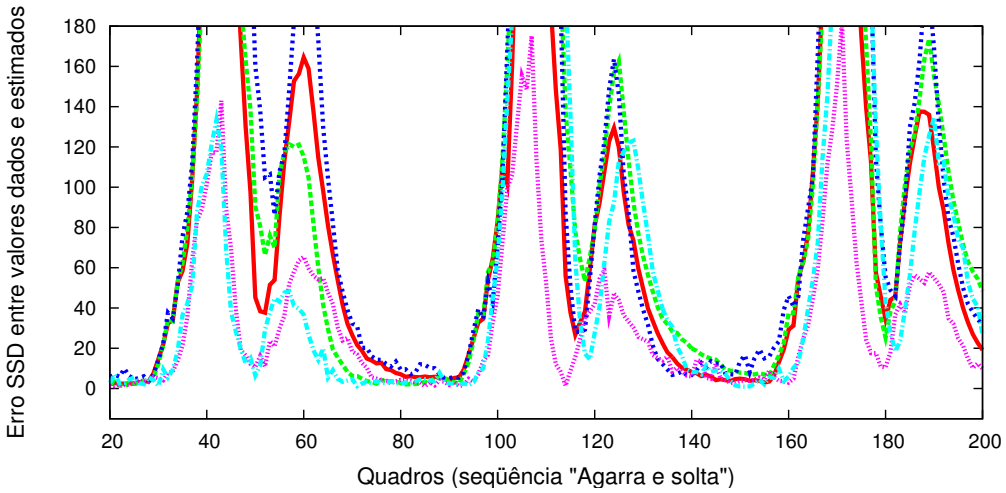
Após os gráficos de erro, mostramos quadros de seqüências de vídeo capturadas, no momento da inicialização (B.1 e B.2), durante o rastreamento (B.3 e B.4) e durante o uso do sistema de reconhecimento (B.5).

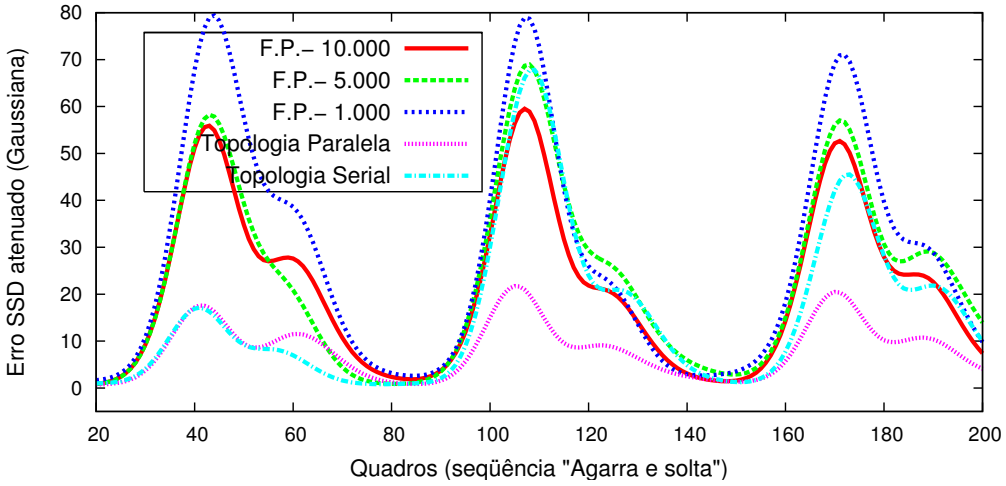
Por fim, mostramos quadros de duas seqüências sintetizadas, "Quatro" (B.6) e "Agarra e gira" (B.7).

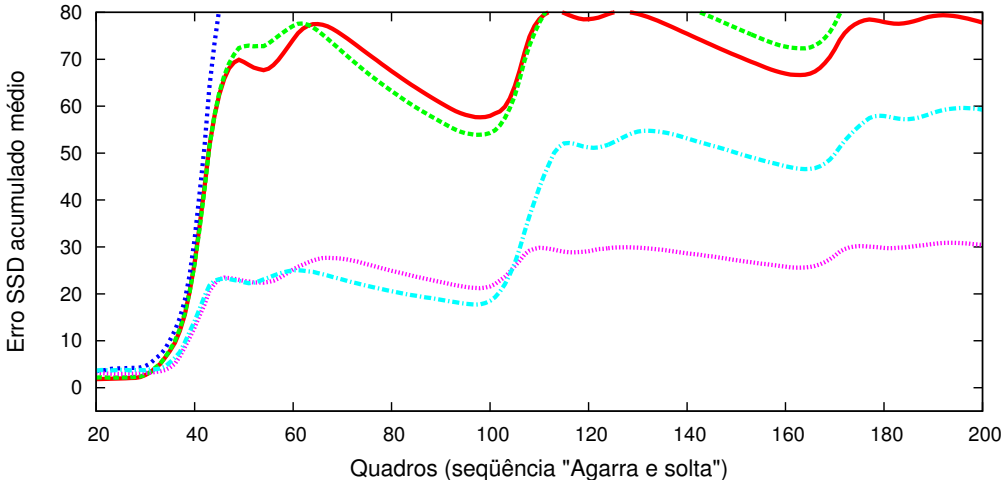

Erro SSD entre valores dados e estimados
180
160
140
120
100
80
60
40
20
0
0
50
100
150
200
Quadros (seqüência "Agarra e gira")

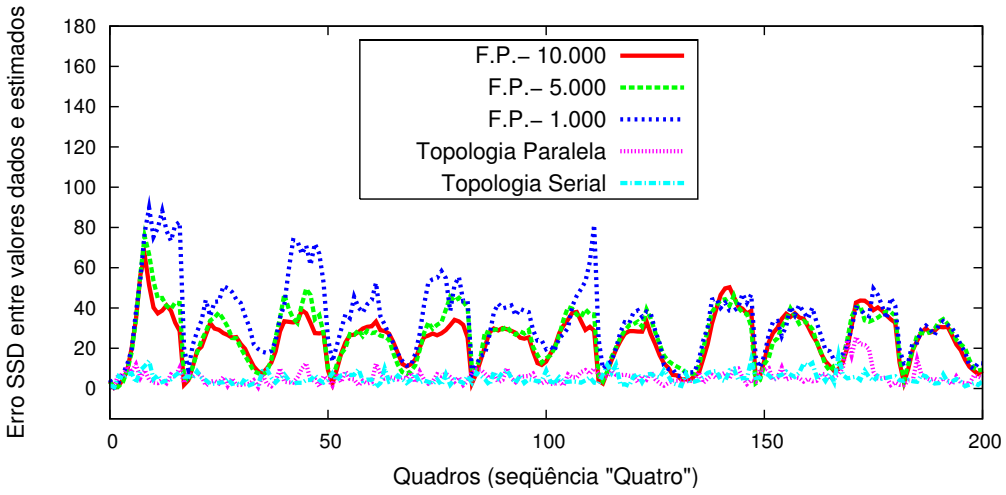

Erro SSD atenuado (Gaussiana)
80
70
60
50
40
30
20
10
0
F.P.– 10.000
F.P.– 5.000
F.P.– 1.000
Topologia Paralela
Topologia Serial
0
50
100
150
200
Quadros (seqüência "Agarra e gira")

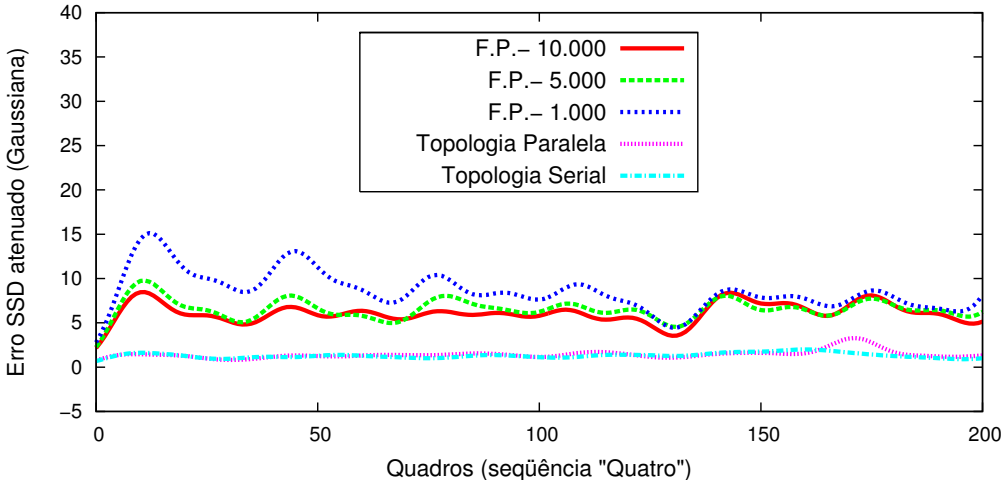

Erro SSD acumulado médio
80
70
60
50
40
30
20
10
0
0
50
100
150
200
Quadros (seqüência "Agarra e gira")

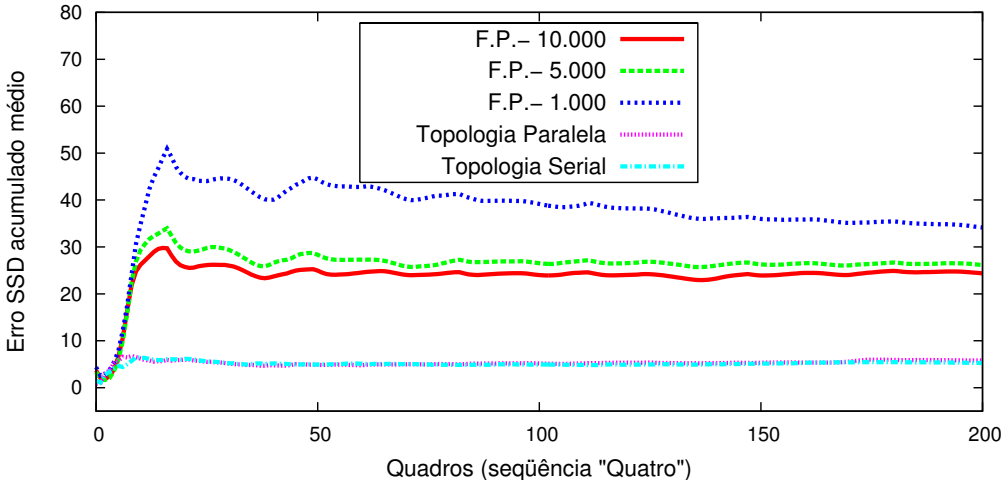

Erro SSD entre valores dados e estimados
Quadros (seqüência "Agarra e solta")

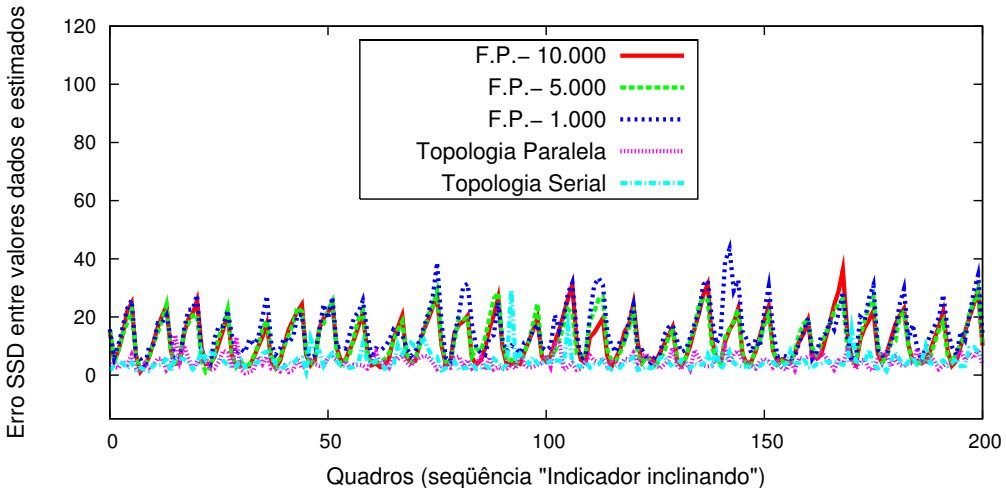

Erro SSD atenuado (Gaussiana)
F.P.– 10.000
F.P.– 5.000
F.P.– 1.000
Topologia Paralela
Topologia Serial
Quadros (seqüência "Agarra e solta")

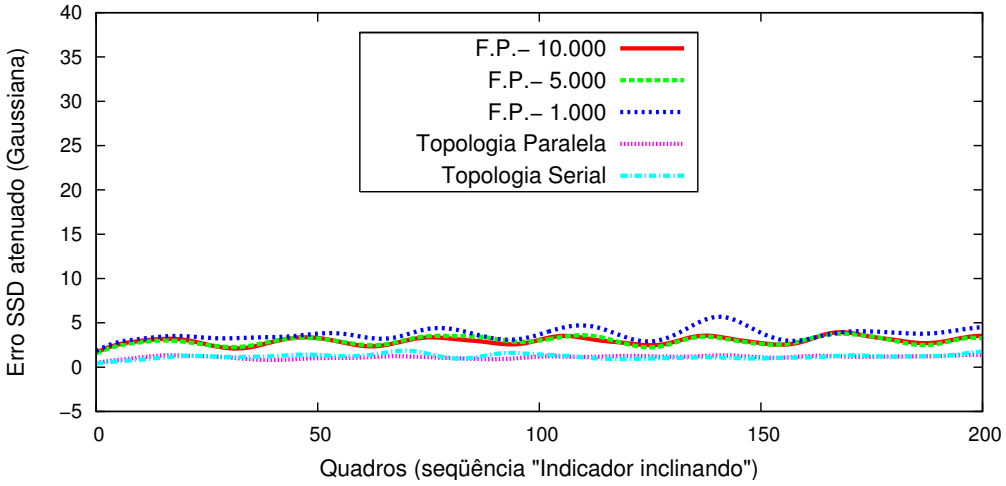

Erro SSD acumulado médio
Quadros (seqüência "Agarra e solta")

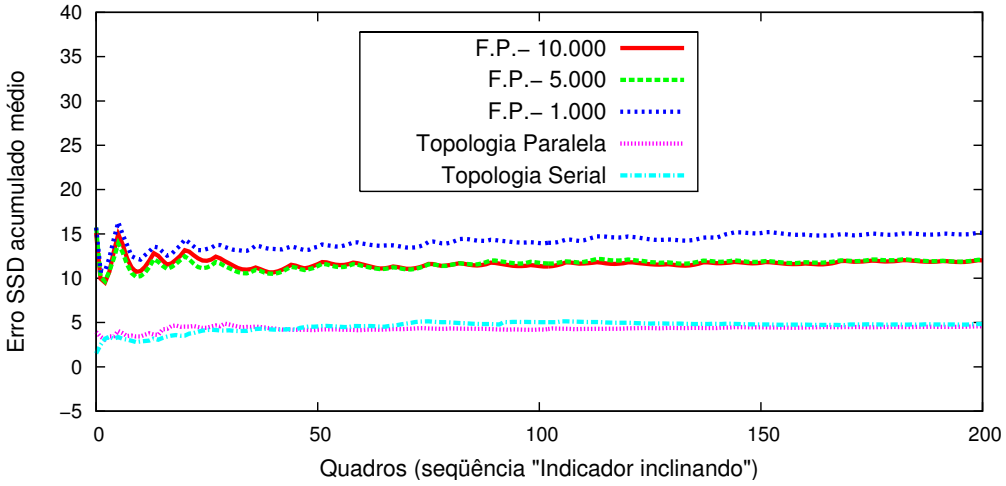

Erro SSD entre valores dados e estimados
Quadros (seqüência "Quatro")
F.P.– 10.000
F.P.– 5.000
F.P.– 1.000
Topologia Paralela
Topologia Serial
0
20
40
60
80
100
120
140
160
180
0
50
100
150
200

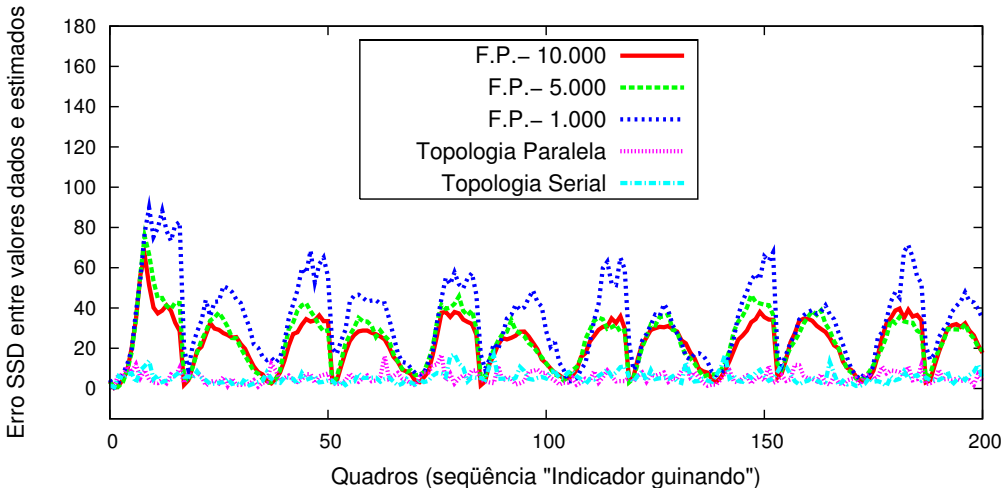

Erro SSD atenuado (Gaussiana)
Quadros (seqüência "Quatro")
F.P.– 10.000
F.P.– 5.000
F.P.– 1.000
Topologia Paralela
Topologia Serial
−5
0
5
10
15
20
25
30
35
40
0
50
100
150
200

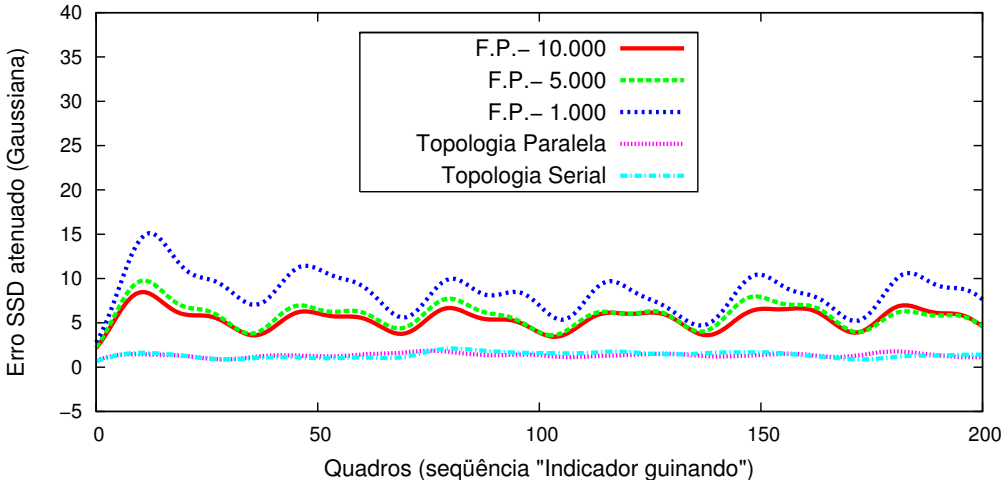

Erro SSD acumulado médio
Quadros (seqüência "Quatro")
F.P.– 10.000
F.P.– 5.000
F.P.– 1.000
Topologia Paralela
Topologia Serial
0
10
20
30
40
50
60
70
80
0
50
100
150
200

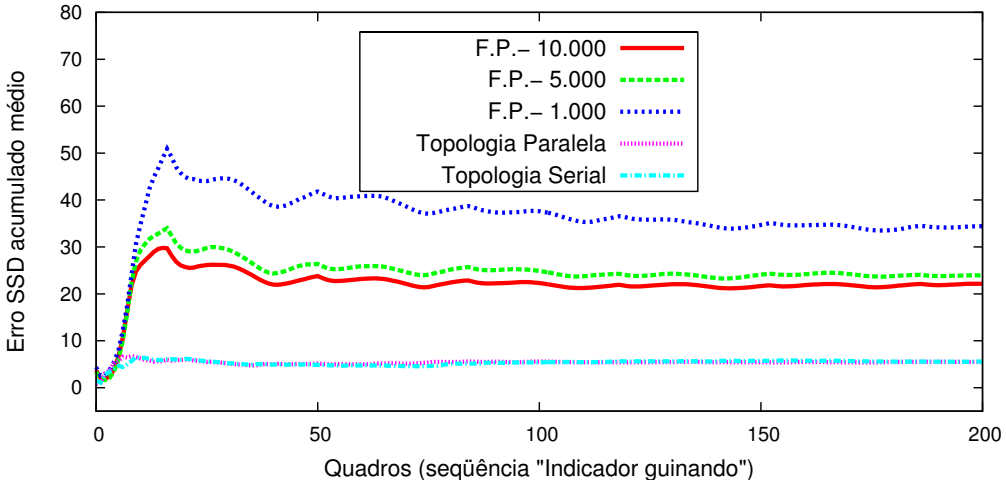

Erro SSD entre valores dados e estimados
Quadros (seqüência "Indicador inclinando")
F.P.– 10.000
F.P.– 5.000
F.P.– 1.000
Topologia Paralela
Topologia Serial
120
100
80
60
40
20
0
0
50
100
150
200

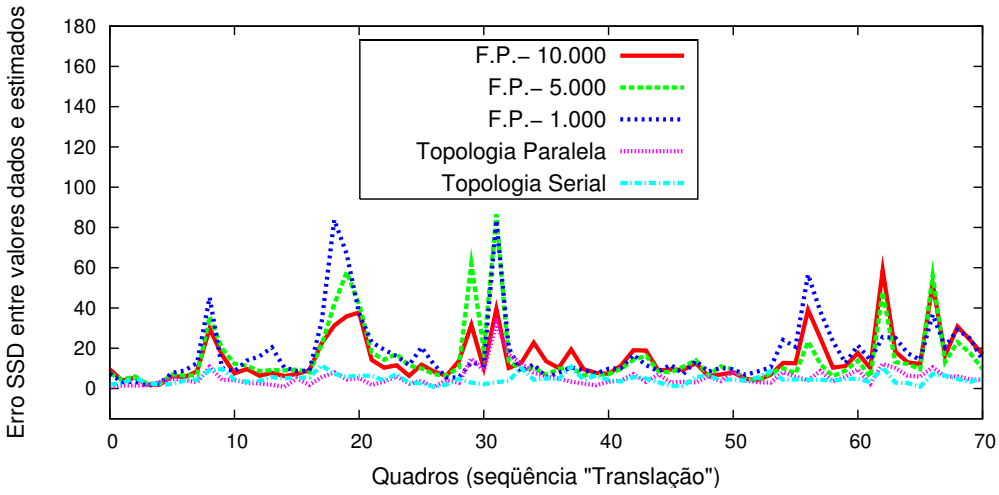

Erro SSD atenuado (Gaussiana)
Quadros (seqüência "Indicador inclinando")
F.P.– 10.000
F.P.– 5.000
F.P.– 1.000
Topologia Paralela
Topologia Serial
40
35
30
25
20
15
10
5
0
–5
0
50
100
150
200

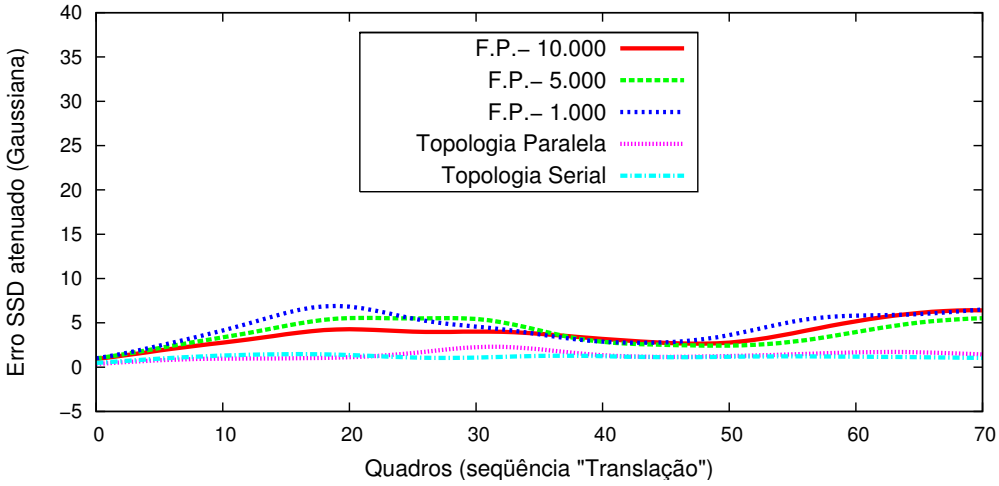

Erro SSD acumulado médio
Quadros (seqüência "Indicador inclinando")
F.P.– 10.000
F.P.– 5.000
F.P.– 1.000
Topologia Paralela
Topologia Serial
40
35
30
25
20
15
10
5
0
–5
0
50
100
150
200


Erro SSD entre valores dados e estimados
Quadros (seqüência "Indicador guinando")
F.P.– 10.000
F.P.– 5.000
F.P.– 1.000
Topologia Paralela
Topologia Serial
0
20
40
60
80
100
120
140
160
180
0
50
100
150
200


Erro SSD atenuado (Gaussiana)
Quadros (seqüência "Indicador guinando")
F.P.– 10.000
F.P.– 5.000
F.P.– 1.000
Topologia Paralela
Topologia Serial
−5
0
5
10
15
20
25
30
35
40
0
50
100
150
200


Erro SSD acumulado médio
Quadros (seqüência "Indicador guinando")
F.P.– 10.000
F.P.– 5.000
F.P.– 1.000
Topologia Paralela
Topologia Serial
0
10
20
30
40
50
60
70
80
0
50
100
150
200


Erro SSD entre valores dados e estimados
Quadros (seqüência "Translação")
F.P.– 10.000
F.P.– 5.000
F.P.– 1.000
Topologia Paralela
Topologia Serial


Erro SSD atenuado (Gaussiana)
Quadros (seqüência "Translação")
F.P.– 10.000
F.P.– 5.000
F.P.– 1.000
Topologia Paralela
Topologia Serial

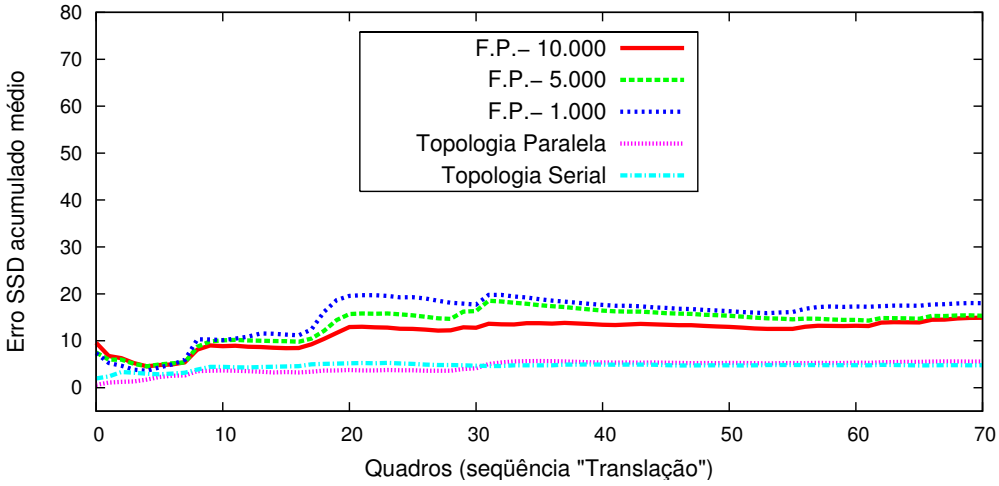

Erro SSD acumulado médio
Quadros (seqüência "Translação")
F.P.– 10.000
F.P.– 5.000
F.P.– 1.000
Topologia Paralela
Topologia Serial

Figura B.1: Seqüência usada no ajuste inicial do vídeo “Dedo”. Cada quadro mostra a imagem original com a projeção da estimação sobreposta.

Figura B.2: Seqüência usada no ajuste inicial do vídeo “Agarra”. Cada quadro mostra a imagem original com a projeção da estimação sobreposta.

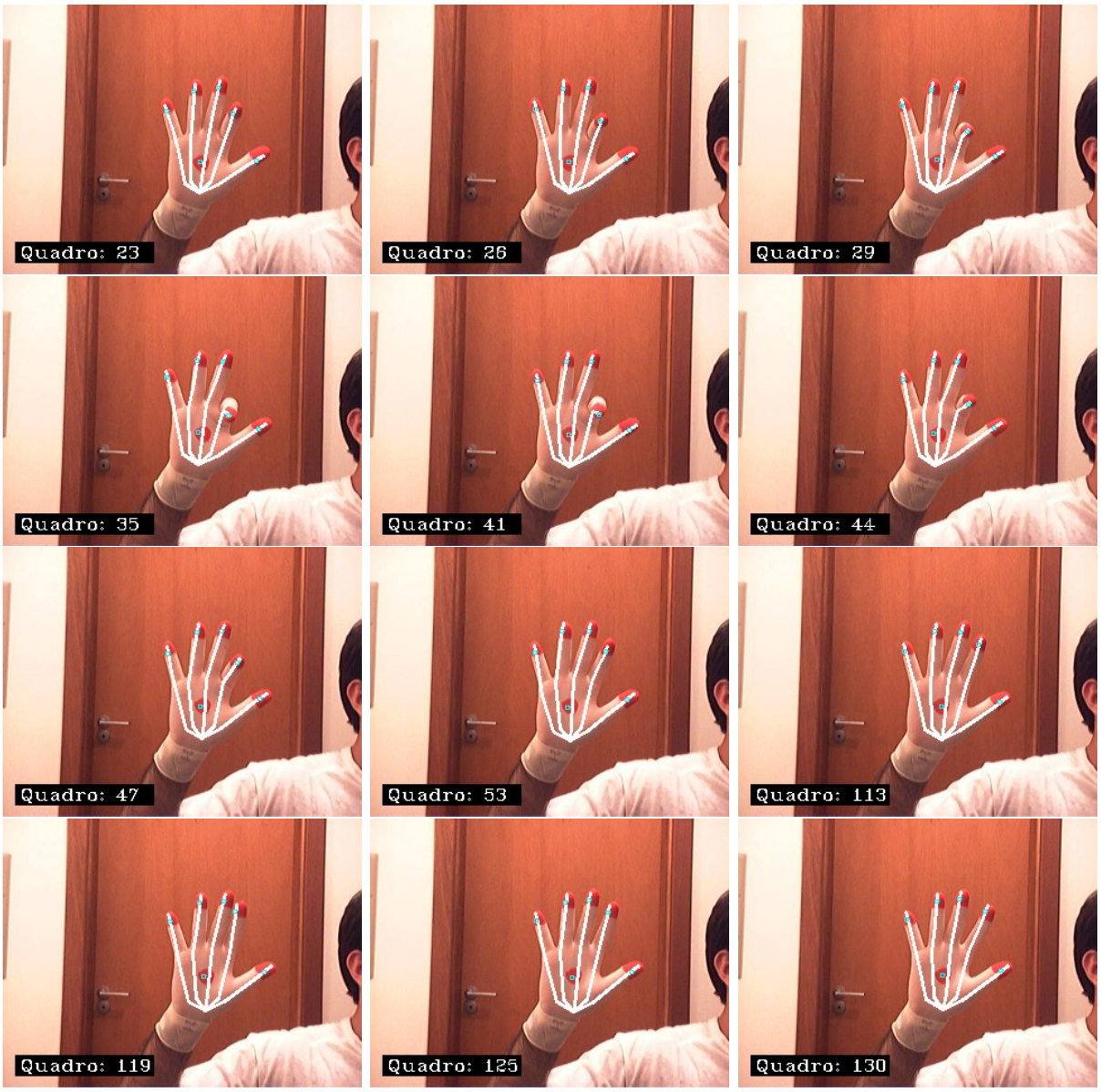


Figura B.3: Seqüência do vídeo “Dedo” mostrando inclinação e guinada do dedo indicador.

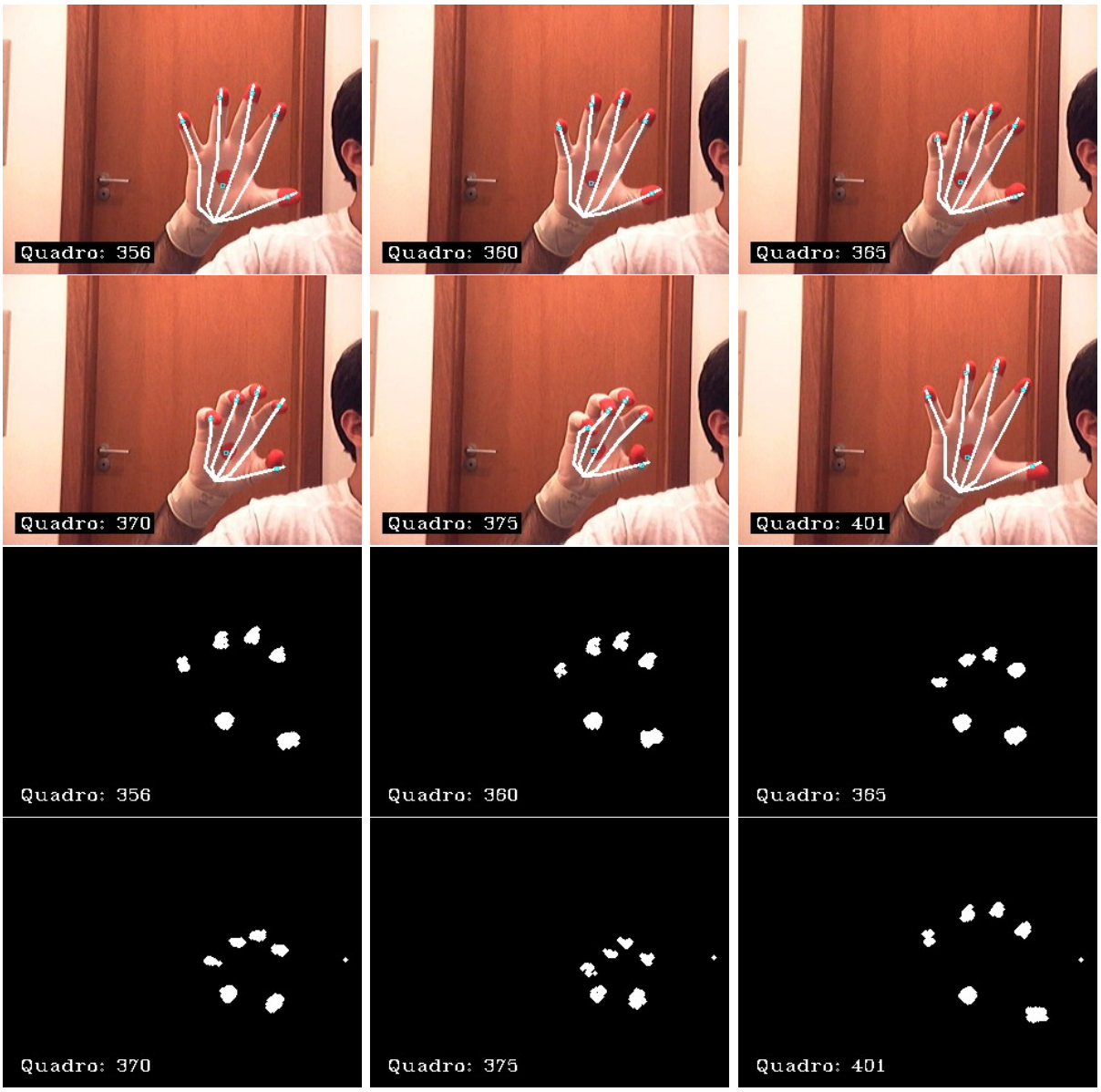


Figura B.4: Seqüência do vídeo “Agarra” mostrando o movimento de agarrar. Abaixo estão as observações correspondentes a cada quadro.

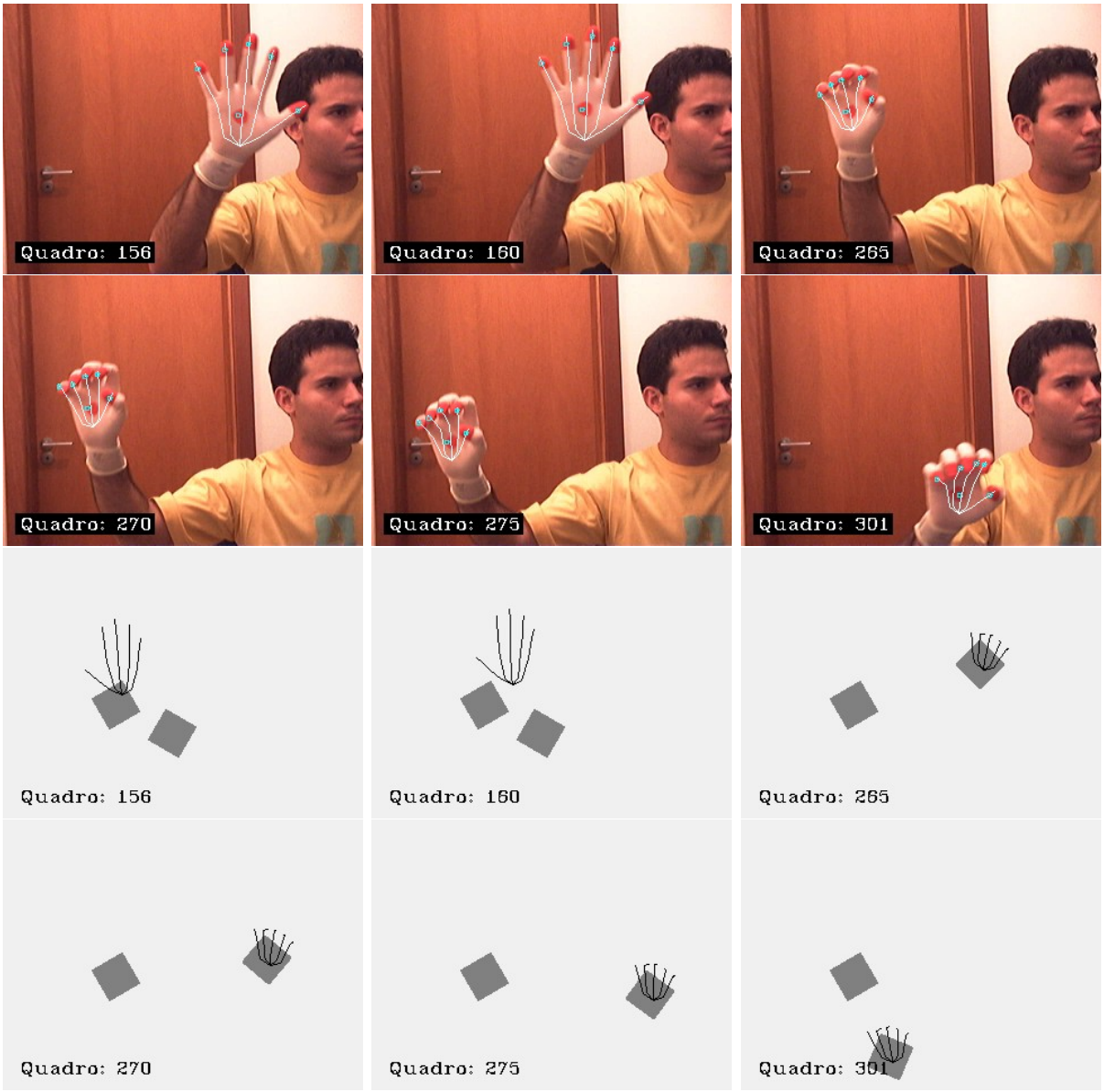


Figura B.5: Seqüência do vídeo "Translação" mostrando a interação com o ambiente virtual. A mão dá voltas pela tela, agarra um dos objetos, e novamente dá voltas pela tela.

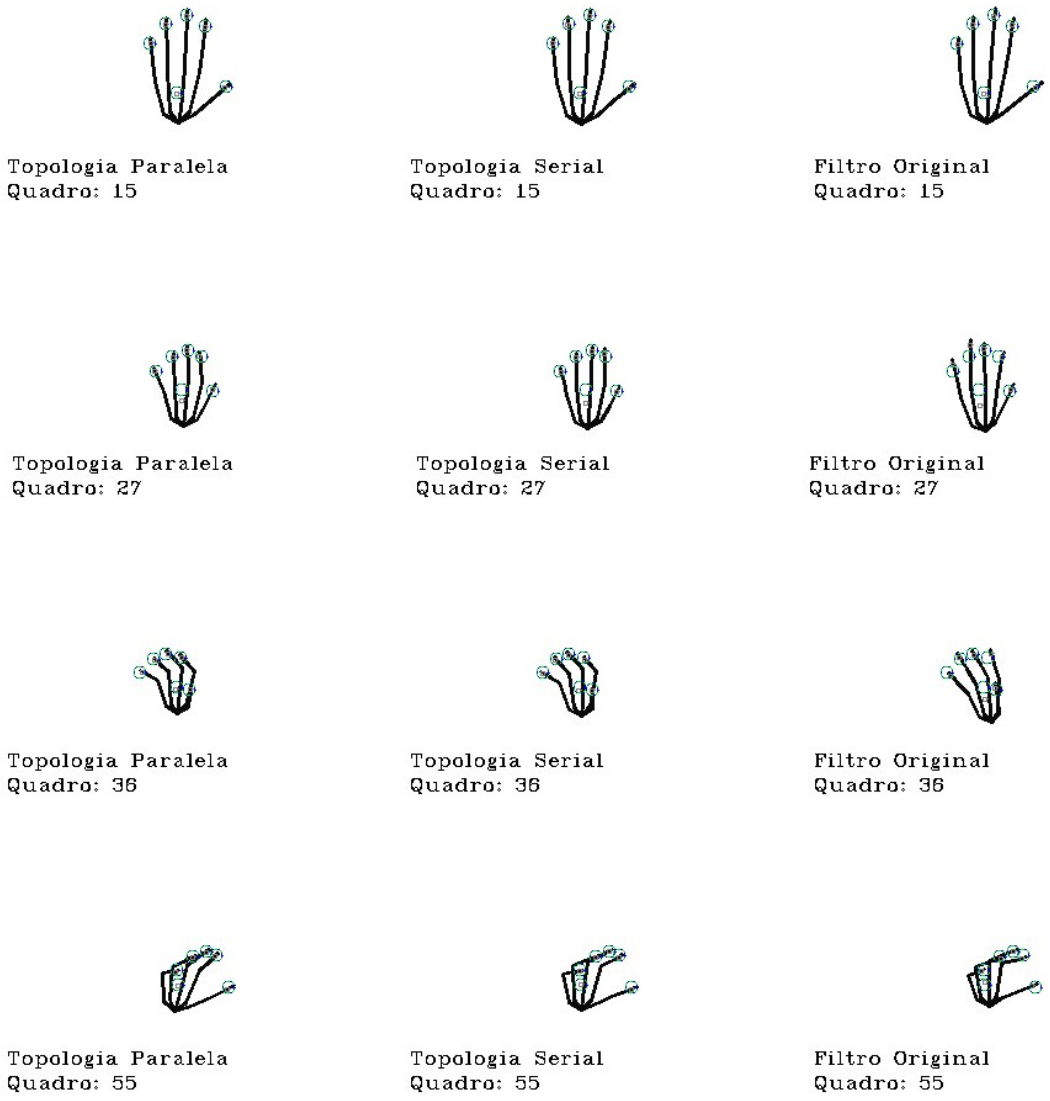


Figura B.6: Seqüência do vídeo sintetizado "Agarra e gira" rastreado com as topologias paralela, serial e com o filtro de partículas tradicional com 1000 partículas. As circunferências marcam as posições das observações.

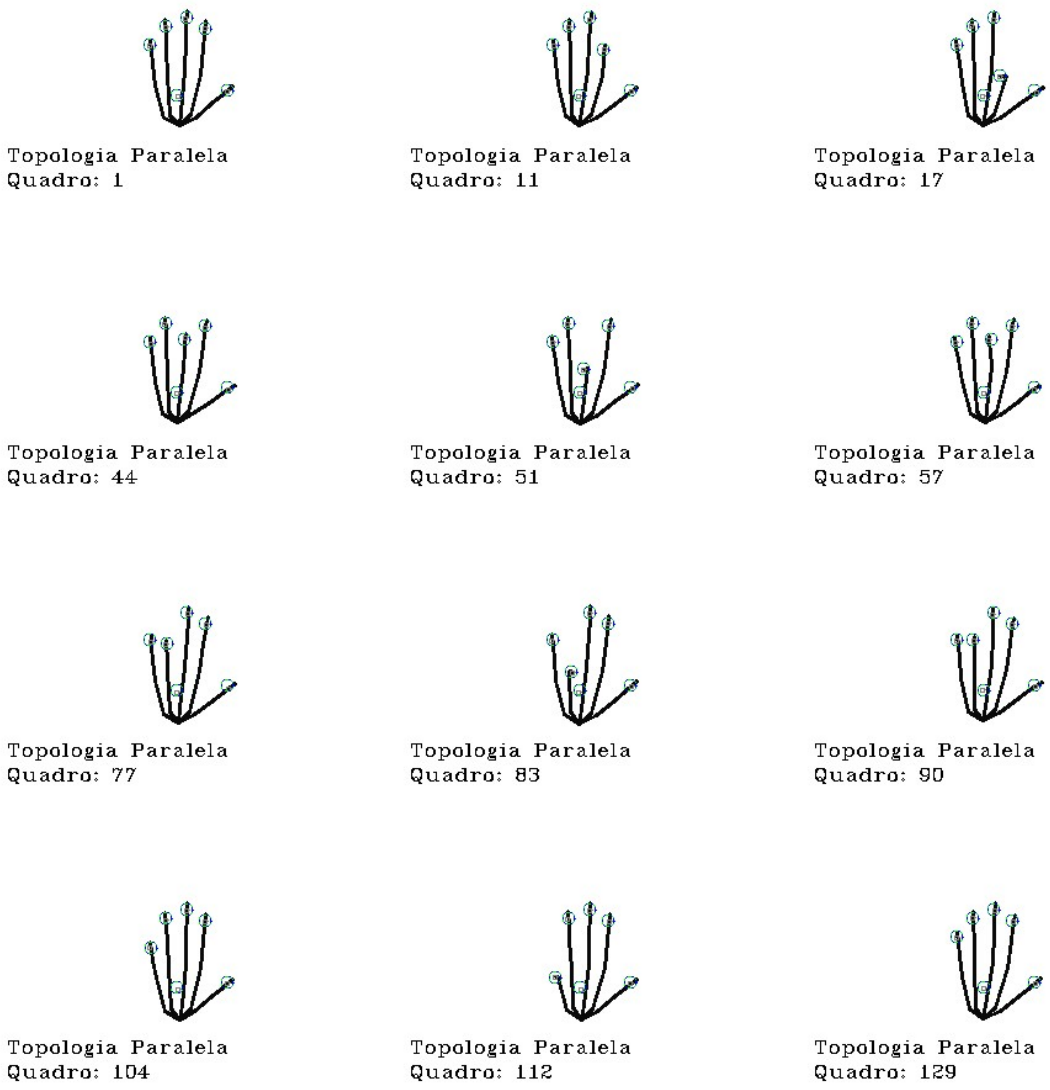


Figura B.7: Seqüência do vídeo sintetizado “Quatro”, rastreado apenas com a topologia paralela. As circunferências marcam as posições das observações.

Referências Bibliográficas

[1] B. D. O. Anderson and J. B. Moore. *Optimal Filtering*. Prentice-Hall, 1979.

[2] C. Bagnoli and H. C. Smith. The Theory of Fuzz Logic and its Application to Real Estate Valuation. *Journal of Real Estate Research*, 16:169–199, 1998.

[3] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1996.

[4] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 25(11):120, 122–125, Nov 2000.

[5] M. Brand, N. M. Oliver, and A. Pentland. Coupled hidden Markov models for complex action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 994–999, June 1997.

[6] G. L. Bretthorst. *Bayesian Spectrum Analysis and Parameter Estimation*, volume 48 of *Lecture Notes in Statistics*. Springer, 1988.

[7] G. L. Bretthorst. An Introduction to Parameter Estimation Using Bayesian Probability Theory. In *Proceedings of Maximum Entropy and Bayesian Methods*, pages 53–79, 1989.

[8] R. T. Collins, R. Gross, and J. Shi. Silhouette-Based Human Identification from Body Shape and Gait. In *Proceedings of the 5th IEEE International Conference on Automatic Face and Gesture Recognition*, pages 351–356, 2002.

[9] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 1990.

[10] R. T. Cox. *The Algebra of Probable Inference*. The Johns Hopkins Press, 1961.

[11] Luiz Henrique de Figueiredo and Jorge Stolfi. Affine Arithmetic: Concepts and Applications. *Numerical Algorithms*, 37:147–158, 2004.

[12] F. Dellaert, W. Burgard, D. Fox, and S. Thrun. Using the CONDENSATION Algorithm for Robust, Vision-based Mobile Robot Localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 588–594, June 1999.

[13] J. Deutscher and I. Reid. Articulated Body Motion Capture by Stochastic Search. *International Journal of Computer Vision*, 61(2):185–205, 2005.

[14] B. Dorner. Chasing the colour glove: Visual hand tracking. Master's thesis, Simon Fraser University, June 1994.

[15] A. Doucet, N. de Freitas, K. P. Murphy, and S. J. Russell. Rao-Blackwellised Particle Filtering for Dynamic Bayesian Networks. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, pages 176–183, 2000.

[16] A. Doucet, S. Godsill, and C. Andrieu. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, 10(3):197–208, 2000.

[17] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley Publishing, second edition, 2000.

[18] W. Feller. *An Introduction to Probability Theory and It's Applications*, volume 1. Wiley Publishing, third edition, 1968.

[19] R. P. Feynman, R. Leighton, E. Hutchings, and A. R. Hibbs. *Surely You're Joking, Mr. Feynman!: Adventures of a Curious Character*. W. W. Norton, 1985.

[20] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[21] J. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes. *Computer Graphics. Principles and Practice*. Addison-Wesley, second edition, 1996.

[22] D. Geer. Will Gesture-Recognition Technology Point the Way? *IEEE Computer*, 37(10):20–23, 2004.

[23] R. C. Gonzales and R. E. Woods. *Digital Image Processing*. Addison-Wesley, 1992.

[24] N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In *Proceedings-F Radar and Signal Processing*, volume 140, number 2, pages 107–113, 1993.

[25] A. G. Gray and A. W. Moore. 'N-Body' Problems in Statistical Learning. In T.K. Leen, T.G. Dietterich, and V. Tresp, editors, *Proceedings of the Neural Information Processing Systems*, pages 521–527, 2001.

[26] P. C. Gregory. *Bayesian Logical Data Analysis for the Physical Sciences*. Cambridge University Press, 2005.

[27] Y. Iba. Population Monte Carlo algorithms. *Transactions of the Japanese Society for Artificial Intelligence*, 16:279–86, April 2002.

[28] M. Isard and A. Blake. CONDENSATION — Conditional Density Propagation for Visual Tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998.

[29] E. T. Jaynes. *Probability Theory : The Logic of Science*. Cambridge University Press, April 2003.

[30] M. Klaas, N. de Freitas, and A. Doucet. Toward Practical N^2 Monte Carlo: The Marginal Particle Filter. In *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence*, 2005.

[31] H. Koike, Y. Sato, and Y. Kobayashi. Integrating paper and digital information on EnhancedDesk: a method for realtime finger tracking on an augmented desk system. *ACM Transactions on Computer-Human Interaction*, 8(4):307–322, 2001.

[32] Y. Kuno, T. Watanabe, Y. Shimosakoda, and S. Nakagawa. Automatic detection of human for visual surveillance system. In *Proceedings of the 13th International Conference On Pattern Recognition*, pages 865–869, 1996.

[33] T. J. Loredo. From Laplace to Supernova SN 1987A: Bayesian inference in astrophysics. In P. F. Fougere, editor, *Maximum Entropy and Bayesian Methods*, pages 81–142. Kluwer Academic Publishers, 1990.

[34] Yi Ma, S. Soatto, J. Kosecka, and S. S. Sastry. *An Invitation to 3-D Vision*. Springer, 2003.

[35] J. MacCormick and A. Blake. A Probabilistic Exclusion Principle for Tracking Multiple Objects. *International Journal of Computer Vision*, 39(1):57–71, 2000.

[36] J. MacCormick and M. Isard. Partitioned Sampling, Articulated Objects, and Interface-Quality Hand Tracking. In *Proceedings of the 6th European Conference on Computer Vision*, pages 3–19, 2000.

[37] T. M. Massie and J. K. Salisbury. The PHANToM Haptic Interface: A Device for Probing Virtual Objects. In *Proceedings of the 3rd Annual Symposium Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pages 295–301, 1994.

[38] P. S. Maybeck. *Stochastic Models, Estimating, and Control*. Academic Press, 1979.

[39] B. W. Miners, O. A. Basir, and M. Kamel. Knowledge-Based Disambiguation of Hand Gestures. In *Proceeding of the IEEE International Conference on Systems, Man and Cybernetics*, pages 201–206, 2002.

[40] R. E. Neapolitan. *Learning Bayesian Networks*. Prentice Hall, 2003.

[41] K. Okuma, A. Taleghani, N. de Freitas, J. J. Little, and D. G. Lowe. A Boosted Particle Filter: Multitarget Detection and Tracking. In *Proceedings of the 8th European Conference on Computer Vision*, pages 28–39, 2004.

[42] A. Papoulis. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill, fourth edition, 2002.

[43] V. I. Pavlovic, R. Sharma, and T. S. Huang. Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):677–695, 1997.

[44] E. A. Penharbel, R. C. Destro, F. Tonidandel, and R. A. C. Bianchi. Filtro de Imagem Baseado em Matriz RGB de Cores-Padrão para Futebol de Robôs. In *Anais do XXIV Congresso da Sociedade Brasileira de Computação (I EnRI)*, 2004.

[45] S. Perrin and M. Ishikawa. Fuzzy Features for Gesture Recognition Using High Speed Vision Camera. In *Anais do XVI Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens*, 2003.

[46] M. D. Pesce. *Programming Microsoft® DirectS]how® for Digital Video and Television, publisher = MS Press, year = 2003,*.

[47] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. In *Readings in speech recognition*, pages 267–296. Morgan Kaufmann, 1990.

[48] H. S. Sawhney and S. Ayer. Compact Representations of Videos Through Dominant and Multiple Motion Estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):814–830, 1996.

[49] C. E. Shannon. A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1):3–55, 2001.

[50] B. Shneiderman. Direct manipulation for comprehensible, predictable and controllable user interfaces. In *Proceedings of the 2nd International Conference on Intelligent User Interfaces*, pages 33–39, 1997.

[51] L. Sigal, S. Sclaroff, and V. Athitsos. Skin Color-Based Video Segmentation under Time-Varying Illumination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(7):862–877, 2004.

[52] A. F. M. Smith and A. E. Gelfand. Bayesian Statistics Without Tears: A Sampling-Resampling Perspective. *American Statistician*, 46(2):84–88, May 1992.

[53] B. Stenger. *Model-Based Hand Tracking Using a Hierarchical Bayesian Filter*. PhD thesis, University of Cambridge, March 2004.

[54] B. Stroustrup. *The C++ Programming Language*. Addison-Wesley, 2000.

[55] A. van Dam. Post-WIMP User Interfaces. *Communications of the ACM*, 40(2):63–67, 1997.

[56] J. Vermaak, A. Doucet, and P. Pérez. Maintaining Multi-Modality through Mixture Tracking. In *Proceedings of the 9th IEEE International Conference on Computer Vision*, pages 1110–1116, 2003.

[57] P. A. Viola and M. J. Jones. Rapid Object Detection using a Boosted Cascade of Simple Features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 511–518, 2001.

[58] C. P. Vogler. *American Sign Language Recognition: Reducing the Complexity of the Task with Phoneme-Based Modeling and Parallel Hidden Markov Models*. PhD thesis, University of Pennsylvania, 2003.

[59] E. A. Wan and R. van der Merwe. *Kalman Filtering and Neural Networks*, chapter Chapter 7 : The Unscented Kalman Filter, (50 pages). Wiley Publishing, 2001.

[60] M. Warkus. *The Official GNOME 2 Developer's Guide*. No Starch Press, 2004.

[61] A. Wexelblat. An approach to natural gesture in virtual environments. *ACM Transactions on Computer-Human Interaction*, 2(3):179–200, 1995.

[62] Y. Wu and T. S. Huang. Vision-Based Gesture Recognition: A Review. In *Proceedings of the International Gesture Workshop on Gesture-Based Communication in Human-Computer Interaction*, pages 103–115, 1999.