RAONI MAÍRA RESENDE

# DESENVOLVIMENTO DE UMA INTERFACE HUMANO-ROBÔ UTILIZANDO VISÃO COMPUTACIONAL E SISTEMAS A EVENTOS DISCRETOS

Belo Horizonte
03 de agosto de 2006

Universidade Federal de Minas Gerais
Instituto de Ciências Exatas
Programa de Pós-Graduação em Ciência da Computação

# DESENVOLVIMENTO DE UMA INTERFACE HUMANO-ROBÔ UTILIZANDO VISÃO COMPUTACIONAL E SISTEMAS A EVENTOS DISCRETOS

Dissertação apresentada ao Curso de Pós-Graduação em Ciência da Computação da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

RAONI MAÍRA RESENDE

Belo Horizonte
03 de agosto de 2006

# Resumo

Este trabalho apresenta uma nova abordagem para o desenvolvimento de interfaces visuais humano-robô baseadas em visão computacional. O método proposto se baseia na utilização de uma linguagem composta por gestos simples, onde um gesto isolado não possui qualquer significado, mas uma palavra, composta por uma seqüência de gestos realizados na ordem correta, gera uma resposta. Dessa maneira, pode ser definida uma gramática e associada uma ação do robô a cada uma dessas palavras, permitindo ao operador humano comandar o robô de maneira intuitiva.

Considerando apenas gestos simples, o sistema de visão computacional privilegia a robustez se baseando em técnicas qualitativas, menos precisas porém mais robustas. A saída do sistema de visão computacional é processada por um sistema a eventos discretos estocástico que detecta a execução de comandos. O reconhecimento é realizado de maneira contínua, sendo o início e o fim de cada gesto identificados implicitamente pelo modelo. Foram utilizados dois tipos de sistemas a eventos discretos: cadeias de Markov e Modelos Ocultos de Markov (MOMs, do inglês, *Hidden Markov Models*). Para ambos os tipos foi desenvolvida uma metodologia para a construção automática do modelo.

Os testes realizados comprovam a eficácia do método mesmo em ambientes complexos e com o executor dos gestos se movendo. Foi comprovada a robustez do método e a baixa incidência de falsos positivos, principalmente para os MOMs. A utilização de MOMs no contexto proposto apresentou desempenho superior ao das cadeias de Markov. Também foi identificada a importância da escolha correta dos comandos da gramática, para evitar uma deterioração significativa da taxa de reconhecimento à medida que a quantidade de comandos aumenta.

# Abstract

This work presents a computer vision human-robot interface based on gesture recognition. A grammar composed of strings of simple gestures is defined so that an isolated gesture has no meaning to the system, and only a sequence of gestures performed in the correct order, according to the grammar, will issue a command to the robot. Each of these words, made of gestures, can be associated to a command, which will be issued to the robot everytime it's recognized. Through the use of this interface, a human operator can control a robot in a natural and intuitive way.

Since only simple gestures are considered, the computer vision system is based on qualitative techniques that exhibits robustness properties. The output of the computer vision system is passed on to a stochastic discrete event system which is responsible for the commands recognition. Continuous gesture recognition is performed and the model implicitly identifies the beginning and the end of each gesture. Two types of discrete event systems were employed: Markov chains and Hidden Markov Models (HMM). The models were built automatically for both of these types.

Experimental results shows that the proposed methodology yields robust recognition with low ocurrence of false positives even in complex backgrounds and with the operator moving. The HMMs outperformed the Markov chains in the proposed methodology context. The commands must be correctly chosen to avoid serious degradation of the performance as the number of considered commands increase.

# Agradecimentos

Agradeço primeiramente aos meus pais, Maria José Mendes e Heliéser José Resende e à minha avó Ana Mendes Ferreira, pois eles são os responsáveis por toda a caminhada até esse ponto.

À minha esposa Danielle pelo companheirismo e à minha filha Fernanda pela alegria e por ter me emprestado os brinquedos para que eu pudesse realizar diversos experimentos.

Agradeço ao meu orientador Guilherme Pereira pela infindável compreensão, disponibilidade e paciência. Ao Luiz Chaimowicz pelo aprendizado e orientação no projeto orientado, essencial nessa caminhada. Ao Mário pela discussão inicial, responsável por que eu me enveredasse por essa área, e pelos primeiros passos no mestrado. Ao Carceroni por sempre ter despendido toda a atenção necessária, sendo tão criterioso quanto necessário.

Agradeço à ATAN por todo o apoio durante os momentos em que mais precisei.

Agradeço também às demais pessoas que foram tão importantes na minha formação acadêmica, profissional e pessoal, em ordem cronológica, Renato Mesquita, Ana Liddy, Marcelo Szuter, Guilherme Salles, Leandro Barbosa, Ebenezer Oliveira e todos os demais colegas da ATAN com quem tive a oportunidade de trabalhar.

Aos colegas do VERLab, principalmente ao Pedro Shiroma, pela disponibilidade e introdução ao *Pioneer*.

Aos colegas presentes de longa data Fabrício Vivas e Sinval Nascimento.

# Sumário

# Lista de Figuras

# Lista de Tabelas

# Lista de Símbolos

$\Gamma$ função de estados ativos.

$a_{ij}$ probabilidade de atingir o estado $j$ oriundo do estado $i$.

$\alpha_t(i)$ probabilidade de observação da seqüência parcial $O = O_0 O_1 \cdots O_t$ até o instante $t$ e do estado $i$ no instante $t$, dado o modelo $\lambda$.

$\beta_t(i)$ probabilidade de observação da seqüência parcial de $t+1$ a $T-1$ dado o estado $i$ no instante $t$ e o modelo $\lambda$.

$b_j(k)$ probabilidade de se observar o símbolo $v_k$ no estado $j$.

$N_{real}$ número de comandos que deveriam ter sido identificados (*ground truth*).

$\gamma_t(i)$ probabilidade de se estar no estado $i$ no instante $t$ dadas a seqüência de observações $O$ e o modelo $\lambda$.

$q_0$ estado inicial.

$\lambda$ conjunto completo de parâmetros de um MOM.

$X_m$ conjunto de estados marcados ou finais.

$q_k$ estado no instante $k$.

$v_k$ k-ésimo símbolo do conjunto de símbolos distintos $V$.

$V$ conjunto de símbolos distintos observáveis do modelo.

$\xi_t(i,j)$ probabilidade de se estar no estado $i$ no instante $t$ e no estado $j$ no instante $t+1$, dado o modelo e a seqüência de observações.

$A$ matriz de probabilidades de transição.

$B$ matriz da distribuição probabilística de observação dos símbolos nos estados.

$E$ conjunto de eventos.

$f$ função de transição.

$h$ altura da imagem em pixels.

$T$ comprimento da seqüência de observações.

$T_{avg}$ tamanho médio das seqüências utilizadas para treinar o modelo.

$T_{max}$ tamanho da maior seqüência utilizada para treinar o modelo.

$T_{min}$ tamanho da menor seqüência utilizada para treinar o modelo.

$w$ largura da imagem em pixels.

$X$ conjunto de estados.

MOMs Modelos Ocultos de Markov.

AEFD autômato de estados finitos determinístico.

D número de erros de deleção.

GSMS *Generalized Semi-Markov Scheme*.

H número de comandos identificados corretamente.

I número de erros de inserção.

MEF Máquina de Estados Finitos.

S número de erros de substituição.

SED Sistema a Eventos Discretos.

TR Taxa de Reconhecimento.

# Capítulo 1

# Introdução

A Robótica pessoal e de serviço é atualmente uma das áreas de pesquisa mais ativas em robótica, com grande potencial de crescimento de acordo com a aceitação da sociedade. Robôs de serviço interagem diretamente com as pessoas, tornando a existência de interfaces naturais e fáceis de usar fundamental. O foco de grande parte dos trabalhos passados é predominantemente navegação e manipulação e pouco foi o interesse em sistemas robóticos equipados com interfaces flexíveis, que permitam ao usuário controlar o robô por meios naturais. Um robô com interface natural possibilitaria toda uma nova gama de aplicações, como por exemplo um robô de limpeza, guiado para limpar locais específicos e pegar o lixo [Waldherr et al., 2000]. Diferentemente de robôs industriais, que são encontrados principalmente em linhas de produção, os robôs de serviço são utilizados pelas massas, milhões de usuários em todos os tipos de locais, do hospital à casa, do restaurante ao escritório [Pransky, 1996]. Diversos especialistas apontam que em um futuro próximo, entre cinco e dez anos, robôs de serviço se tornarão um mercado multibilionário [Bishop, 2006].

A utilização crescente de robôs nas mais diversas aplicações, algumas destas exigindo alta interatividade, demanda interfaces intuitivas com o homem, uma vez que estas podem reduzir o risco de erro em aplicações críticas, considerando uma interface natural de acordo com os reflexos condicionados do ser humano. Um exemplo disso seriam os robôs cirúrgicos [Pransky, 1996].

A criação de interfaces naturais homem-robô é um assunto que desperta cada vez mais o interesse da comunidade científica mundial devido também à extensa gama de aplicações que ela possibilita. Entretanto podemos considerar que estas ainda se encontram em seu estágio inicial. À medida que aumenta a capacidade dos robôs de realizar mais tarefas de maneira autônoma se torna necessário avaliar como deve ser a interface humano-robô

para acomodar essa evolução [Scholtz, 2002].

Na robótica atual, principalmente na robótica industrial, existe pouca ou nenhuma interação direta entre homem e robô. Os robôs são programados para determinada atividade e se limitam a executar tais atividades, o que gera uma segmentação do trabalho entre homens e robôs. Essa exclusividade de humanos ou robôs acaba limitando as aplicações dos robôs, uma vez que o uso de robôs acaba se tornando viável somente em ambientes totalmente controlados. As tarefas devem ser inteiramente desempenhadas pelo robô, situações em que as capacidades de decisão e planejamento de um supervisor são necessárias não podem surgir ou devem ser resolvidas remotamente, devido à ausência de interface com o ser humano. Sistemas robóticos projetados para interagir com seres humanos abrem um novo leque de aplicações abrangendo tarefas complexas e não repetitivas que requerem supervisão humana e podem ser executadas por robôs, porém devem ser coordenadas pelos operadores (autonomia supervisionada). Tais sistemas precisariam permitir uma interação natural entre o robô e o operador tornando a interface visual fundamental. Para isso o sistema precisa ser capaz de localizar o operador no ambiente e rastrear os seus movimentos em tempo real.

A área de busca e resgate urbano é uma área em que humanos devem interagir com robôs e com a informação produzida por eles. Pesquisas baseadas em dados reais apontam que os erros de execução (i.e., controle do robô) são muito maiores que os erros de intenção (i.e, onde e como utilizar o robô) [Murphy e Casper, 2002], o que indica que uma interface natural poderia reduzir os erros nessa tarefa.

Além disso, interfaces baseadas em visão computacional possuem outras aplicações menos óbvias, como por exemplo os computadores usáveis (*wearable computers*). A utilização desse tipo de interface nesses dispositivos pode fornecer usabilidade mesmo quando o usuário está em movimento [Kölsch et al., 2004].

Este trabalho propõe uma interface visual humano-robô baseada em visão computacional. A comunicação é feita por uma linguagem composta por gestos simples, definindo uma gramática onde um gesto isolado não possui qualquer significado, mas uma palavra, composta por uma seqüência de gestos realizados na ordem correta, gera uma resposta. Uma ação do robô pode ser associada a cada uma dessas palavras, permitindo ao operador humano comandar o robô de maneira intuitiva.

Dois módulos principais compõem o sistema, o de visão computacional e o de processamento estocástico dos gestos. Buscou-se a utilização de métodos estocásticos, uma vez que estes fornecem níveis de confiança em vez de apenas sucesso ou falha, o que é importante quando se trabalha com problemas difíceis para os quais não há método imune a falhas disponível [Siskind e Morris, 1996].

A interface é baseada em algoritmos tradicionais de visão computacional, simples e eficientes, uma vez que a tarefa alocada para o sistema de visão, da maneira que foi formulada, pode ser considerada simples. Sendo simples, pode ser resolvida por técnicas qualitativas que são robustas. Dessa maneira, o módulo de visão computacional foca em identificar apenas as características importantes, necessárias para atingir o seu objetivo, em vez de tentar reconstruir o mundo.

Para o processamento dos gestos foram utilizados reconhecedores de linguagens estocásticos, sendo selecionadas as cadeias de Markov, por se tratarem de máquinas de estados finitos probabilísticas e Modelos Ocultos de Markov (MOMs, do inglês, *Hidden Markov Models*). MOMs foram utilizados largamente ao longo das últimas decádas para o reconhecimento de padrões e classificação e formam hoje a base para uma grande gama de soluções para análise de dados e modelagem estatística. MOMs são utilizados nas mais diversas aplicações como, por exemplo, guiar mísseis [Nilubol et al., 1998], predição de crises no Oriente Médio [Schrodt, 2000] e pesquisas biomédicas envolvendo seqüências de genes [Kulp et al., 1996, Yada e Hirosawa, 1996, Pedersen et al., 1996].

## 1.1 Contribuições

As principais contribuições deste trabalho são:

- desenvolvimento de uma interface visual utilizando visão computacional baseada na utilização de sistemas a eventos discretos estocásticos para o reconhecimento contínuo de gestos;
- desenvolvimento de uma metodologia para o processamento estocástico de gestos utilizando tanto cadeias de Markov quanto MOMs;
- utilização de uma nova abordagem para o cálculo da probabilidade de uma seqüência de observações ser reconhecida utilizando MOMs.

## 1.2 Organização da dissertação

Este trabalho está organizado da maneira que será descrita a seguir. O Capítulo 2 apresenta os trabalhos relacionados e o Capítulo 3 apresenta os conceitos necessários para o correto entendimento da metologia proposta, que por sua vez é apresentada no Capítulo 4. O Capítulo 5 apresenta os resultados obtidos e o Capítulo 6, as conclusões e propostas de trabalhos futuros.

# Capítulo 2

# Revisão Bibliográfica

As interfaces baseadas em visão computacional (IBVC) começaram a ser largamente exploradas em meados da década de noventa, quando foram desenvolvidos desde *mouses* baseados em gestos [Fukumoto et al., 1994, Quek et al., 1995] até interfaces para jogos baseadas em visão computacional [Freeman et al., 1996]. Devido às limitações de processamento computacional, essas interfaces baseavam-se em técnicas simples para que pudessem responder em tempo real aos gestos do usuário, como histogramas de orientação e momentos da imagem [Horn, 1986].

Para avaliar a qualidade e usabilidade de uma IBVC devem ser consideradas quatro características principais, de acordo com Kölsch [Kölsch et al., 2004]: velocidade, exatidão, precisão e robustez. De acordo com Sheridan e Ferrell [Sheridan e Ferrell, 1963] o tempo de latência máximo entre a ocorrência de um evento e a resposta do sistema deve ser de 45 ms para que o usuário não perceba nenhum atraso. A partir de 300 ms de atraso as interfaces passam a parecer lentas, pondendo provocar oscilações e resultando num efeito conhecido como o sintoma "mova e espere".

As interfaces baseadas em gestos podem ser classificadas por diversos parâmetros. Os gestos a serem identificados podem ser estáticos, identificados pela posição e forma, ou dinâmicos, identificados por sua trajetória. A abordagem pode ser *bottom-up* baseada nas características de baixo nível da imagem ou *top-down* como métodos de reconstrução geométrica do corpo. O reconhecimento pode ainda ser realizado em três dimensões ou simplificadamente em duas dimensões.

A maioria dos sistemas que emprega a abordagem *top-down* utiliza um modelo geométrico do corpo humano. Partes do corpo são modeladas, por exemplo, como cilindros [Hogg, 1983] e *super quadrics* [Horowitz e Pentland, 1991], entre outros, e os parâmetros do modelo são determinados a partir das imagens utilizando análise espaço-temporal

[Yamamoto, 1991], propagação de restrições [O'Rourke e Badler, 1980] ou análises modais [Horowitz e Pentland, 1991]. A reconstrução da postura do corpo humano (i.e., extração dos parâmetros do modelo) fornece uma grande gama de informações caso a reconstrução tenha sucesso, como por exemplo, os ângulos das juntas. Entretanto os algoritmos de reconstrução não são robustos para imagens reais, pois estas em geral são muito ruidosas para permitir um casamento fácil do modelo [Yamato et al., 1992]. Além disso, métodos baseados na detecção de arestas e formas não são recomendados para a identificação e rastreamento em *backgrounds* complexos pois sua exatidão é determinada pelo contraste existente entre os planos de fundo e frontal, o que não pode ser garantido nesse caso [Kölsch et al., 2004].

Um exemplo de abordagem *top-down* é o rastreamento e identificação de movimentos da mão em três dimensões (3D) realizado por Davis e Shah [Davis e Shah, 1994a]. A mão foi modelada por cinco cilindros que eram ajustados à falange distal dos dedos da mão. A trajetória dos cilindros era calculada no espaço 3D podendo ser utilizada no reconhecimento de gestos. Com informações 3D é possível saber a localização exata dos dedos, em coordenadas do mundo, a qualquer momento. Esse conhecimento pode ser explorado sem a preocupação de ambigüidade, como ocorre com a informação bidimensional (2D), uma vez que uma trajetória 2D pode corresponder a diversas trajetórias 3D devido à transformação perspectiva. Além disso, a utilização de modelos 3D e parâmetros de movimento evita a necessidade da correspondência do movimento para mapear características à sua correta trajetória 2D [Rangarajan e Shah, 1991]. A informação 3D pode ser utilizada para remover essas incertezas que surgem em 2D. O algoritmo implementado por Davis e Shah exige que a mão seja o objeto dominante na imagem devendo também iniciar em posição pré-definida.

Entretanto, de acordo com o paradigma da visão qualitativa proposto por Aloimonos [Aloimonos, 1990], a reconstrução do corpo não é essencial para o reconhecimento das ações humanas. A visão computacional geralmente é tratada como um problema de recuperação [Horn, 1986], buscando recuperar (reconstruir) o mundo e suas propriedades, como a posição 3D e formatos dos objetos, de modo a realizar ações e tomar decisões baseados nessas informações. Aloimonos questionou a tradicional abordagem reconstrucionista e propôs o paradigma da visão qualitativa-objetiva (*purposive-qualitative*) como uma alternativa. Nesse paradigma a visão não é vista como um fim, mas sim como parte de um processo maior que visa, por exemplo, realizar determinadas tarefas. A abordagem deve ser objetiva para formular as soluções corretas (simples) para os problemas e qualitativa para obter soluções robustas.

Desde então abordagens *bottom-up*, que utilizam heuristicamente características

de baixo nível extraídas de imagens reais, têm sido objeto de numerosos estudos [Yamato et al., 1992] sendo utilizadas, por exemplo, para contagem de pessoas a partir da identificação de regiões candidatas. Em geral, as características de baixo nível não fornecem descrições tão ricas quanto as representações baseadas em modelos, mas seu processo de extração é mais simples e robusto que os procedimentos de ajuste dos modelos.

Freeman [Freeman et al., 1996] utilizou a abordagem *bottom-up* para implementar interfaces para jogos baseadas em visão computacional. Foram implementadas interfaces baseadas em técnicas como momentos da imagem, histogramas de orientação e fluxo óptico, que a partir dos gestos do usuário controlava os jogos. Aplicações gráficas interativas têm a grande vantagem de permitir a exploração do *feedback* visual da tela, pois o usuário vê imediatamente o resultado do seu gesto e pode alterá-lo caso necessário. Por exemplo, se um usuário se inclina para fazer uma curva no jogo e vê que a inclinação não foi suficiente, ele pode se inclinar um pouco mais.

Uma desvantagem da abordagem *bottom-up* é que a descrição de categorias de ações em representações de baixo nível é mais difícil que em representações de alto nível (baseadas em modelos), pois as relações entre as dimensões do vetor de características e a descrição de alto nível das categorias não é explícita. Além disso, as dimensões do vetor de características podem ser muito grandes para serem entendidas intuitivamente [Yamato et al., 1992].

Gestos estáticos são definidos como o reconhecimento de uma posição da mão ou do corpo enquanto gestos dinâmicos são identificados por trajetórias do braço ou da mão. Um dos principais problemas no caso dos gestos estáticos é a identificação da mão. A maioria dos trabalhos restringe fortemente o ambiente de utilização exigindo *background* uniforme [Segen e Kumar, 1998], *background* estático, luvas coloridas ou marcadores na mão [Cipolla et al., 1993, Davis e Shah, 1994c]. Detecção de mãos em quaisquer *backgrounds* foi realizado, por exemplo, por Triesch e Malsburg [Triesch e von der Malsburg, 1996] obtendo uma taxa de acerto de 86.2% e por Cui e Weng [Cui e Weng, 1999], porém nenhum desses métodos é capaz de realizar a detecção em tempo real, requisito das interfaces com o usuário. Alguns trabalhos apresentam métodos capazes de reconhecer a postura da mão independentemente do ângulo de visão [Wu e Huang, 2000, Rosales et al., 2001, Ong e Bowden, 2004], porém esses artigos não fazem nenhuma ressalva quanto à execução em tempo real.

Para o reconhecimento de gestos dinâmicos é feito em geral o rastreamento de um objeto e sua trajetória é processada para identificar ou não a ocorrência de um gesto. O aprendizado e reconhecimento de gestos dinâmicos, mesmo em duas dimensões, é

complexo, uma vez que os dados amostrados da trajetória de qualquer gesto variam para cada execução do mesmo. Isso é justificado por várias razões como freqüência de amostragem, erros de rastreamento, ruído e, principalmente, variações na execução do gesto pelo ser humano, tanto espacial quanto temporal [Hong et al., 2000a].

Recentemente algumas abordagens baseadas em sistemas a eventos discretos foram propostas para a modelagem e reconhecimento de gestos. Uma das principais vantagens dessa abordagem é que a mesma não requer um grande conjunto de dados para treinar o modelo [Hong et al., 2000a]. Bobick e Wilson [Bobick e Wilson, 1997] utilizaram uma abordagem baseada em estados para representar e reconhecer gestos onde cada gesto foi modelado como uma seqüência de estados em um espaço de configurações. Os dados para treinamento foram alinhados temporalmente e segmentados manualmente. Primeiro, diversas amostras de um gesto são utilizadas para computar sua *principal curve* [Hastie e Stuetzle, 1989] que é parametrizada pelo comprimento do arco. O mapeamento de cada ponto da amostra do gesto exemplo para um comprimento de arco ao longo da curva é um produto desse cálculo da curva. A seguir, segmentos de reta de comprimento uniforme são utilizados para aproximar a curva discretizada. Cada segmento de reta é representado por um vetor e todos os segmentos de reta são agrupados em *clusters*. Um estado é definido para indicar o *cluster* ao qual um segmento de reta pertence. Um gesto então é definido por uma seqüência de estados. O processo de reconhecimento consiste em avaliar se uma trajetória passa ou não pelos estados na ordem correta. Cada gesto é uma trajetória única no espaço.

Hong et al. [Hong et al., 2000b] utilizaram MEFs (Máquinas de Estados Finitos) para reconhecer gestos dinâmicos. As posições 2D da cabeça e das mãos foram utilizadas como características, sendo localizadas utilizando segmentação por cor e rastreamento. Cada gesto foi modelado como uma seqüência de estados no espaço espacial-temporal (*spatial-temporal*), sendo cada estado modelado como uma Gaussiana multi-dimensional. Hong considerou que as trajetórias de um gesto são um conjunto de pontos distribuídos espacialmente, podendo portanto ser representados por um conjunto de regiões espaciais Gaussianas. Um limiar foi selecionado para representar a variância espacial permitida para cada estado, determinando a variância espacial do gesto. O número de estados e seus parâmetros foram calculados utilizando *k-means dynamic clustering* nos dados de treinamento do gesto, sem informação temporal (as informações espaciais e temporais dos dados de treinamento são primeiro desacopladas). O resultado desse primeiro treinamento é utilizado para a segmentação e alinhamento dos dados. O modelo então é treinado com a informação temporal alinhada e a informação espacial é atualizada. Esse procedimento fornece a seqüência de estados final que representa o gesto. Cada seqüên-

cia de estados é uma MEF reconhecedora de um gesto. Toda essa etapa de treinamento é realizada off-line. Quando um novo vetor de características é computado, cada MEF decide se continua no estado atual ou vai para o próximo estado baseado nos parâmetros espaciais e temporais. Quando o estado final é atingido o gesto é considerado reconhecido. Caso o estado final de mais de uma MEF seja atingido ao mesmo tempo, aquele com a menor distância média acumulada é considerado o vencedor. O reconhecimento é realizado on-line. Essa técnica foi testada com sucesso em gestos como acenar com a mão esquerda, acenar com a mão direita, desenhar um círculo, desenhar um oito, entre outros. A complexidade computacional do processamento foi reduzida por meio do armazenamento do contexto nas MEFs, sendo processados apenas os novos dados a cada quadro.

Posteriormente, Hong [Hong et al., 2000a] utilizou uma variação do algoritmo de casamento de padrões Knuth-Morris-Pratt(KMP) para acelerar o processamento, considerando o reconhecimento de gestos como um casamento de cadeia de caracteres entre uma seqüência de dados e a seqüência de estados da MEF.

Davis e Shah [Davis e Shah, 1994b] utilizaram uma máquina de estados finitos para modelar quatro fases diferentes de um gesto genérico qualitativamente. Se a mão está em movimento para a posição do gesto são criadas trajetórias dos dedos usando correspondência dos pontos das pontas dos dedos do movimento no plano da imagem. Vetores são utilizados então para aproximar as trajetórias e os gestos desconhecidos são casados com os gestos de uma biblioteca usando esses vetores. Foram reconhecidos sete gestos, que representam as ações esquerda, direita, para cima, para baixo, agarrar, girar e parar, sem a utilização de hardware especial.

Siskind e Morris [Siskind e Morris, 1996] propuseram uma metodologia para classificação de eventos visuais utilizando uma abordagem de máxima verossimilhança. Seu trabalho se baseou na modelagem das características dos perfis de movimentação, variando ao longo do tempo, de objetos que participam em diferente eventos simples. Os eventos considerados foram pegar, deixar, empurrar, puxar, soltar e lançar (*pick up, put down, push, pull, drop, throw*). Em vez de formular lógicas detalhadas e modelos geométricos de classes de eventos manualmente, os parâmetros de um modelo genérico são determinados empiricamente a partir dos dados de treinamento. O rastreador utilizado usa um conjunto de técnicas baseadas em cor e movimento, operando quadro a quadro e rastreando de maneira independente objetos coloridos e que se movimentam. Cada região identificada no quadro é modelada como uma elipse, abstraindo-se assim dos dados da imagem. Cada elipse é representada por um vetor de características. Então a partir desse vetor de características adota-se a abordagem de máxima verossimilhança para o reco-

nhecimento de eventos. Utiliza-se técnicas de aprendizado supervisionado para treinar um modelo gerador para cada classe de eventos a partir de um conjunto de exemplos de treinamento para cada classe. O modelo gerador obtido é utilizado para classificar as novas observações em classes existentes. Um dos vídeos do conjunto de treinamento é escolhido para ser o evento canônico. Esse vídeo deve conter o mesmo número de seqüências de elipses tais quais existem objetos participando no evento. Essas seqüências são utilizadas como referência para identificar os conjuntos de elipses nas demais seqüências de treinamento. Os vídeos foram ajustados manualmente para que o início e o fim do vídeo coincidissem com o início e fim do evento e também para que todos os vídeos representando o mesmo evento tivessem o mesmo número de quadros. O experimento utilizou um conjunto de 72 vídeos, sendo 12 para cada gesto. Desses, 6 foram utilizados para treinamento e 6 para validação, escolhidos aleatoriamente.

As principais diferenças da abordagem de Siskind e Morris para aquela proposta nesta dissertação são o reconhecimento contínuo on-line, sem necessidade da indicação do início e fim de cada evento e a utilização de Modelos Ocultos de Markov (MOMs, do inglês, *Hidden Markov Models*) discretos no lugar dos contínuos. O método de rastreamento utilizado por Siskind e Morris também restringe que os objetos participantes do evento sejam praticamente os únicos na cena e exige pós-processamento para a identificação dos objetos que estão em toda a seqüência.

Vogler e Metaxas [Vogler e Metaxas, 1998] apresentaram um arcabouço para o reconhecimento isolado e contínuo da linguagem de sinais americana (ASL - *American Sign Language*). Os dados foram obtidos por um sistema de rastreamento 3D e apresentados como entrada para MOMs realizarem o reconhecimento. Três câmeras posicionadas ortogonalmente eram responsáveis pela identificação e rastreamento em 3D. O algoritmo de rastreamento baseou-se no filtro de Kalman objetivando evitar problemas causados por oclusão. Para o treinamento foram utilizados dados de visão computacional e de um aparato de captura de movimento, devido à grande quantidade de dados necessária para treinar os MOMs e o alto custo computacional do sistema de visão. O reconhecimento contínuo é complexo pois as fronteiras entre os gestos individuais não são claras. A utilização de MOMs permite capturar as variações estatísticas tanto na posição quanto na duração dos movimentos, bem como a segmentação dos dados. A identificação das fronteiras entre os símbolos foi realizada utilizando informações geométricas e da trajetória percorrida juntamente com o algoritmo de Viterbi [Viterbi, 1967] para identificar o início de cada gesto. Vogler utilizou a topologia esquerda para direita por acomodar bem variações no comprimento e duração dos gestos. O número de estados do modelo foi determinado experimentalmente. Foi considerado um vocabulário de 53 gestos, utili-

zando 486 frases, totalizando 2345 sinais. Dessas frases, 389 (80%) foram utilizadas para treinamento e 97 (20%) para validação, escolhidas aleatoriamente. O reconhecimento obteve taxa de acerto de 89,91%.

A abordagem proposta nesta dissertação difere da de Vogler pois sugere procedimento para gerar os modelos automaticamente, inclusive determinando o seu número de estados. Também não necessita da informação de posição, pois o rastreamento 3D é complexo. Além disso, não exige segmentação para identificar o início de cada gesto, sendo essa identificação realizada implicitamente pelo modelo.

Redes Neurais também foram utilizadas no reconhecimento de gestos. Por exemplo, Yang e Narendra [Yang e Ahuja, 1999] utilizaram redes neurais com atraso de tempo (TDNN - *time-delay neural network*) para o reconhecimento de 40 gestos da linguagens de sinais americana (ASL - *American Sign Language*). TDNN é uma rede neural multi-camadas *feedforward* que utiliza atrasos entre as camadas para representar relações temporais entre os eventos no tempo. O sistema se baseou em algoritmos para reconhecimento e extração de padrões de movimento utilizando trajetórias. Foram utilizados 80% dos dados para treinamento e 20% para validação, obtendo taxa de reconhecimento de 96.21% nos dados de validação.

Este capítulo apresentou os trabalhos relacionados identificando as principais diferenças entre estes e a metodologia proposta neste trabalho. O próximo capítulo apresenta os conceitos necessários para o entendimento da metodologia proposta.

# Capítulo 3

# Conceitos Preliminares

Este capítulo tem por objetivo introduzir os conceitos necessários para o entendimento da metodologia proposta nesta dissertação. Ele é dividido em três seções principais: sistemas a eventos discretos, cadeias de Markov e MOMs. Os textos das seções tem caráter apenas introdutório sendo recomendadas as referências citadas para mais informações e detalhes. As primeiras seções são breves e a que trata de *Hidden Markov Models* (Modelos Ocultos de Markov) é mais detalhada por ser um assunto ainda não tão disseminado quanto os demais.

## 3.1 Sistemas a Eventos Discretos

Um Sistema a Eventos Discretos (SED) deve satisfazer as seguintes condições: o espaço de estados deve ser um conjunto discreto e o mecanismo de transição entre estados deve ser dirigido por eventos. Portanto um SED pode ser definido como um sistema de estados discretos, dirigido por eventos, cuja evolução dos estados depende da ocorrência de eventos discretos assíncronos durante o tempo [Cassandras e Lafortune, 1999].

O comportamento de um SED pode ser modelado por uma linguagem, se considerarmos o conjunto de eventos $E$ como o "alfabeto" e as seqüências (finitas) de eventos como as "palavras". Um evento pode ser definido como algo que ocorre instantaneamente e que causa transições de um valor de estado para outro.

A utilização de autômatos é um formalismo para a modelagem de eventos discretos, assim como a utilização de redes de Petri. Ambos os formalismos representam linguagens utilizando uma estrutura de transição de estados. A diferença entre os dois é como eles representam a informação do estado.

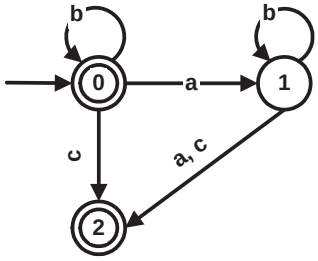


Figura 3.1: Exemplo de autômato com três estados onde $X = \{0, 1, 2\}$, $E = \{a, b, c\}$, $q_0 = 0$ (marcado pela seta) e $X_m = \{0, 2\}$.

Um autômato é um dispositivo capaz de representar uma linguagem de acordo com regras bem definidas. Um autômato determinístico, representado por $G$, é uma tupla de seis elementos

$$G \equiv (X, E, f, \Gamma, q_0, X_m)\,, \tag{3.1}$$

onde:
$X$ é o conjunto de estados;
$E$ é o conjunto finito de eventos associados com as transições em $G$;
$f : X \times E \to X$ é a função de transição; $f(x, e) = y$ significa que há uma transição do estado $x$ para o estado $y$ definida pelo evento $e$;
$\Gamma : X \to 2^E$ é a função de eventos ativos (ou factíveis); $\Gamma(x)$ é o conjunto de todos os eventos $e$ para os quais $f(x, e)$ é definida, denominada *conjunto de eventos ativos* (ou factíveis) de $G$ em $x$;
$q_0$ é o estado inicial;
$X_m \in X$ é o conjunto de estados marcados ou finais.

A Figura 3.1 exemplifica essas definições.

Se $X$ é um conjunto finito, $G$ é denominado um autômato de estados finitos determinístico (AEFD) . Esse modelo também é conhecido como *Generalized Semi-Markov Scheme* (GSMS) .

## 3.2 Cadeias de Markov

A análise de cadeias de Markov fornece um arcabouço para o estudo de diversos SEDs de interesse prático, variando de jogos de azar e do mercado de ações até o projeto de sistemas de computadores e redes de comunicação [Cassandras e Lafortune, 1999].

Considere um sistema que pode ser descrito como estando a qualquer instante em um

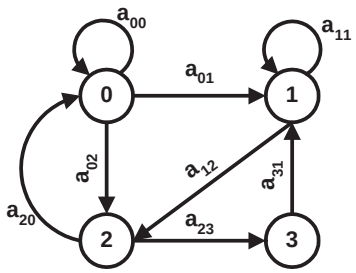


Figura 3.2: Cadeia de Markov com quatro estados (0 a 3).

estado do conjunto de $N$ estados distintos $0, 1, \cdots, N-1$, como ilustrado na Figura 3.2, onde $N = 4$. Em instantes discretos, uniformemente espaçados, o sistema passa por uma transição (possivelmente para o mesmo estado) de acordo com um conjunto de probabilidades associadas ao estado. Os instantes de tempo associados às transições são definidos como $t = t_0, t_1, \cdots$, e o estado no instante $t$ é representado por $q_t$. Em geral, uma descrição completa desse sistema requer a especificação do estado atual no instante $t$, bem como todos os estados predecessores. Para o caso especial de uma cadeia de Markov discreta de primeira ordem, essa descrição probabilística é truncada para apenas o estado atual e seu predecessor, i.e.,

$$P[q_t = S_j | q_{t-1} = S_i, q_{t-2} = S_k, \cdots] = P[q_t = S_j | q_{t-1} = S_i]. \tag{3.2}$$

O processo descrito constitui um modelo de Markov observável uma vez que a saída do processo é o conjunto de estados em cada instante de tempo, onde cada estado corresponde a um evento observável.

A principal característica das cadeias de Markov é que seu comportamento estocástico é definido pelas probabilidades de transição da forma $P[q_{k+1} = x' | q_k = x]$, onde $x$ é o estado atual e $x'$ é o próximo estado, para todos os valores de $x$, $x'$. Dadas essas probabilidades de transição e uma distribuição para o estado inicial é possível determinar a probabilidade de se estar em qualquer estado em qualquer instante.

Uma cadeia de Markov é definida por [Cassandras e Lafortune, 1999]:

1. Um espaço de estados $X$.
2. A distribuição inicial de probabilidades $p_0(x) = P[q_0 = x]$, $\forall\, x \in X$.
3. As probabilidades de transição $p(x', x)$ onde $x$ é o estado atual e $x'$ é o próximo estado.

As probabilidades de transição podem ser representadas na forma de uma matriz, denominada matriz de probabilidades de transição, $A$,

$$A \equiv [a_{ij}(k)], \qquad i, j = 0, \ldots, N-1 \tag{3.3}$$

onde $a_{ij}(k)$ é a probabilidade de se atingir o estado $j$ oriundo do estado $i$ no instante $k$, formalmente

$$a_{ij}(k) \equiv P[q_{k+1} = j | q_k = i]. \tag{3.4}$$

Como $a_{ij}(k)$ é uma probabilidade temos $0 \leq a_{ij}(k) \leq 1$ e $\sum_{\forall j} a_{ij}(k) = 1$ para qualquer estado $i$.

O funcionamento das cadeias de Markov ficará mais claro quando a metodologia for apresentada (Seção 4.3).

## 3.3 Modelos Ocultos de Markov

Modelos Ocultos de Markov (MOMs, do inglês, *Hidden Markov Models*)[1] são um tipo de modelo estocástico, também conhecidos como fontes de Markov ou funções probabilísticas de cadeias de Markov. MOMs tem sido largamente utilizados ao longo das duas últimas décadas para o reconhecimento de padrões e classificação. Com a popularidade, diversos livros contendo abordagens baseadas na utilização de MOMs foram publicados em diversas áreas como biomedicina [Baldi e Brunak, 1998, Durbin et al., 1998], onde são utilizados, por exemplo, para classificação de proteínas em classes, e inteligência artificial [Russell e Norvig, 2002], onde são utilizados, por exemplo, para o reconhecimento de fala e gestos.

A teoria básica foi publicada por Baum et al. entre 1966 e 1972 [Baum e Petrie, 1966, Baum e Egon, 1967, Baum e Sell, 1968, Baum et al., 1970, Baum, 1972] e foi implementada para o reconhecimento de fala por diversos autores na década de 1970 como Baker [Baker, 1975]. Entretanto, a popularização dos MOMs ocorreu somente no início da década de 1980 com a publicação de diversos tutoriais.

Por brevidade essa seção será focada nos três problemas fundamentais para o projeto de MOMs, a saber: o cálculo da probabilidade de uma seqüência de observações dado um MOM específico; a determinação da melhor seqüência de estados do modelo; e o ajuste dos parâmetros do modelo de modo a melhor representar as observações fornecidas.

[1] O texto dessa seção é baseado principalmente nas referências [Rabiner, 1989, Rabiner e Juang, 1986, Russell e Norvig, 2002, Young et al., 2005].

### 3.3.1 Motivação

A limitação de cada estado corresponder a um evento observável presente nas cadeias de Markov e autômatos de estados finitos determinístico é muito restritiva para que o modelo se torne aplicável a diversos problemas. Nessa seção o conceito de modelo de Markov é estendido para considerar os casos em que a observação é uma função probabilística do estado, sendo o modelo resultante um MOM, que é um processo duplamente estocástico embutido onde o processo estocástico subjacente não é observável (é oculto) e pode somente ser observado por meio de um outro conjunto de processos estocásticos que produzem a seqüência de observações.

Podemos exemplificar modelando o problema do lançamento de uma moeda no qual há um quarto com uma barreira (e.g., uma cortina) de modo que não é possível ver o que ocorre do outro lado, onde uma pessoa está lançando uma moeda (ou diversas moedas diferentes). Essa pessoa não irá dizer exatamente o que está fazendo, apenas o resultado de cada lançamento. Dessa maneira uma seqüência de lançamentos *ocultos* será realizada, com o resultado consistindo numa série de caras e coroas. Uma seqüência de observações poderia ser, por exemplo:

$$\mathbf{O} = O_0 O_1 O_2 \cdots O_{T-1} \tag{3.5}$$

$$= CaCaCoCoCoCaCoCoCa \cdots Ca \tag{3.6}$$

onde $Ca$ representa cara, $Co$ representa coroa e $T$ é o comprimento da seqüência.

Para construir um MOM para modelar a seqüência de caras e coroas observada devemos primeiro decidir o que os estados do modelo representam e então quantos estados o modelo deve ter. Uma possibilidade seria assumir que apenas uma moeda está sendo lançada, utilizando um modelo de 2 estados onde cada um desses estados corresponderia a um dos lados da moeda (i.e., cara ou coroa). Esse modelo é mostrado na Figura 3.3(a). Nesse caso o modelo de Markov é observável, sendo portanto um MOM degenerado, e o único parâmetro necessário para especificar o modelo completamente é a probabilidade de um dos lados.

Uma outra maneira seria modelar como na Figura 3.3(b) onde são utilizados também dois estados, porém cada estado corresponde a uma moeda com probabilidades diferentes para cara e coroa. Cada estado é caracterizado por uma distribuição de probabilidades de caras e coroas e as transições entre os estados são dadas por uma matriz de transição. O mecanismo físico que determina as transições entre os estados que são ativadas poderia ser um conjunto de lançamentos independentes ou qualquer outro evento probabilístico.

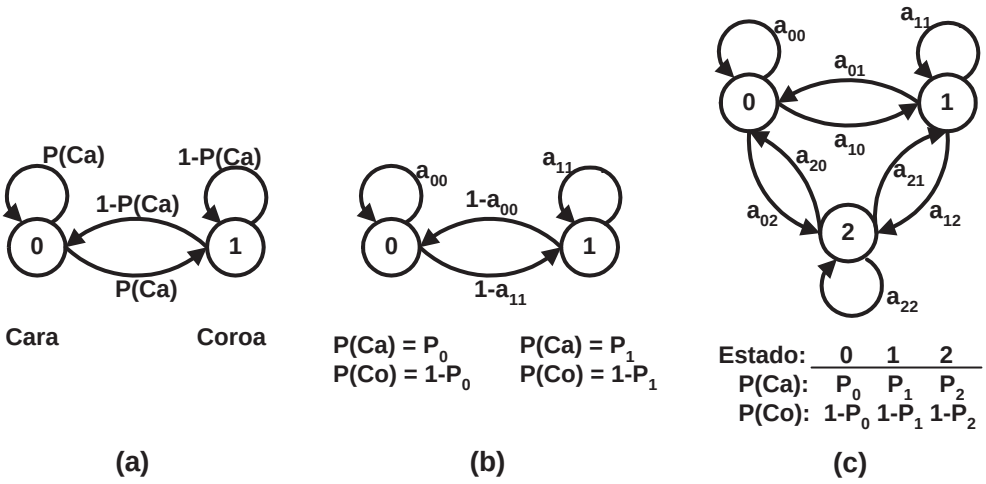


Figura 3.3: Três possíveis modelos de Markov para o experimento do lançamento de uma moeda. (a) Modelo de uma moeda. (b) Modelo de duas moedas. (c) Modelo de três moedas.

Uma terceira maneira de modelar a seqüência de lançamentos observada utilizando um MOM é apresentada na Figura 3.3(c). Nesse modelo são consideradas 3 moedas e a escolha entre as três é feita baseada em um evento probabilístico.

Dados os três modelos mostrados na Figura 3.3 para explicar a seqüência de caras e coroas observada surge uma dúvida natural: qual deles representa melhor essa seqüência. O modelo de uma moeda possui apenas um parâmetro desconhecido; o modelo de duas moedas possui quatro parâmetros desconhecidos; e o modelo de três moedas possui nove parâmetros desconhecidos. Com mais graus de liberdade os MOMs maiores seriam naturalmente mais capazes de modelar uma série de lançamentos que os modelos menores. Apesar disso ser teoricamente verdade, algumas considerações práticas impõem fortes restrições quanto ao tamanho dos modelos que podem ser utilizados. Além disso, utilizar o modelo de 3 moedas seria inapropriado no caso de apenas uma moeda estar sendo lançada uma vez que o evento físico propriamente dito não corresponderia ao modelo sendo utilizado.

### 3.3.2 Definição

Um MOM é caracterizado por:

1. $N$, o número de estados do modelo. Apesar dos estados serem ocultos, para a maioria das aplicações geralmente existe algum significado físico ligado aos estados

ou ao conjunto de estados do modelo. Os estados são representados por $X = \{0, 1, 2, \cdots, N-1\}$, e o estado no instante $t$ por $q_t$.

2. $M$, o número de símbolos distintos observáveis por estado (i.e., o tamanho do alfabeto). Os símbolos observados correspondem à saída física do sistema modelado. Para o experimento do lançamento da moeda os símbolos são simplesmentes cara e coroa. Os símbolos são representados pelo conjunto $V = \{v_0, v_1, \cdots, v_{M-1}\}$.

3. A distribuição de probabilidades de transição entre os estados $A = \{a_{ij}\}$ onde $a_{ij}$ é a probabilidade de atingir o estado $j$ oriundo do estado $i$, formalmente

$$a_{ij} = P[q_{t+1} = j | q_t = i], \qquad 0 \leq i, j < N. \tag{3.7}$$

Para o caso especial de qualquer estado poder ser atingido de qualquer outro estado em uma única transição temos $a_{ij} > 0$ para todo $i, j$ e o modelo é denominado *ergódico*. Para os demais tipos de MOMs temos $a_{ij} = 0$ para um ou mais pares $(i, j)$.

4. A distribuição probabilística de observação dos símbolos nos estados, $B = \{b_j(k)\}$, onde $b_j(k)$ é a probabilidade de se observar o símbolo $v_k$ no estado $j$, formalmente

$$b_j(k) = P[O_t = v_k | q_t = j], \qquad \begin{array}{l} 0 \leq j < N \\ 0 \leq k < M. \end{array}$$

5. A distribuição de estados inicial $\pi = \{\pi_i\}$, onde

$$\pi_i = P[q_0 = i], \qquad 0 \leq i < N \tag{3.8}$$

Dados valores apropriados para $N$, $M$, $A$, $B$ e $\pi$ o MOM pode ser utilizado como gerador para fornecer uma seqüência de observações

$$O = O_0 O_1 O_2 \cdots O_{T-1} \tag{3.9}$$

(onde cada observação $O_t$ é um símbolo de $V$ e $T$ é o número de observações na seqüência) como descrito a seguir:

1. Selecione um estado inicial $q_0 = i$ de acordo com a distribuição de estados inicial $\pi$. Nesse instante $t = 0$.

2. Selecione $O_t = v_k$ de acordo com a distribuição de probabilidades dos símbolos no estado $i$ (i.e., $b_i(k)$).

3. Vá para o novo estado $q_{t+1} = j$ de acordo com a distribuição de probabilidades de transição do estado $i$ (i.e., $a_{ij}$).

4. $t = t + 1$; retorne ao item 2 caso $t < T$, senão termine.

Esse procedimento pode ser utilizado tanto como gerador de observações quanto como modelo para mostrar como uma dada seqüência de observações pode ter sido gerada por um MOM específico.

A especificação completa de um MOM requer portanto a especificação de dois parâmetros do modelo ($N$ e $M$), especificação dos símbolos observáveis e a especificação de três distribuições de probabilidades $A$, $B$ e $\pi$. Por conveniência é utilizada a notação compacta

$$\lambda = (A, B, \pi) \tag{3.10}$$

para indicar o conjunto completo de parâmetros do modelo.

### 3.3.3 Os três problemas básicos dos MOMs

Três problemas básicos devem ser resolvidos para que o modelo apresentado na subseção anterior possa ser aplicado, a saber:

**Problema 1:** Dada a seqüência de observações $O = O_0O_1O_2\cdots O_{T-1}$ e o modelo $\lambda$, como $P(O|\lambda)$ pode ser eficientemente computado?

**Problema 2:** Dada a seqüência de observações $O = O_0O_1O_2\cdots O_{T-1}$ e o modelo $\lambda$, como pode ser definida a seqüência de estados $Q = q_0q_1\cdots q_{T-1}$ que melhor representa a seqüência observada, ou seja, que é ótima em algum sentido.

**Problema 3:** Dada a seqüência de observações $O = O_0O_1O_2\cdots O_{T-1}$ e o modelo $\lambda$, como os parâmetros do modelo podem ser ajustados para maximizar $P(O|\lambda)$?

O Problema 1 se resume a calcular a probabilidade de uma seqüência ter sido produzida pelo modelo, dados os parâmetros do modelo e a seqüência de observações. Esse problema também pode ser visto como identificar como determinado modelo casa com uma seqüência de observações. A solução deste problema nos permite selecionar entre diversos modelos possíveis qual melhor representa as observações.

O Problema 2 tenta identificar a parte oculta do modelo, i.e., encontrar a seqüência de estados "correta". Contudo deve ficar claro que, excetuando-se os modelos degenerados,

não há uma seqüência de estados "correta" a ser encontrada. Em geral é utilizado um critério de otimalidade para resolver esse problema da melhor maneira possível. Porém, existem diversos critérios que podem ser adotados e essa escolha é feita de acordo com a aplicação na qual será utilizada a seqüência de estados utilizada.

O Problema 3 envolve a otimização dos parâmetros do modelo para melhor descrever uma seqüência de observações. A seqüência utilizada para ajustar o modelo é denominada seqüência de treinamento, uma vez que é utilizada para treinar o MOM. Esse problema em geral é o mais importante, pois permite a adaptação de modelos para representar fenômenos reais.

A seguir será descrita a solução para cada um desses problemas.

### 3.3.4 Solução para o Problema 1

A maneira direta de calcular a probabilidade da seqüência $O = O_0O_1O_2 \cdots O_{T-1}$ dado o modelo $\lambda = (A, B, \pi)$ seria enumerar todas as seqüências de estados possíveis de tamanho $T$ (número de observações). Considerando uma seqüência de estados fixa

$$Q = q_0q_1 \cdots q_{T-1}\,, \tag{3.11}$$

onde $q_0$ é o estado inicial. A probabilidade da seqüência de observações $O$ para a seqüência de estados da Equação (3.11) é

$$P(O|Q, \lambda) = \prod_{t=0}^{T-1} P(O_t|q_t, \lambda)\,, \tag{3.12}$$

supondo que as observações são estatisticamente independentes. Então, lembrando que $b_j(k)$ é a probabilidade de se observar o símbolo $v_k$ no estado $j$, obtemos

$$P(O|Q, \lambda) = b_{q_0}(O_0) \cdot b_{q_1}(O_1) \cdots b_{q_{T-1}}(O_{T-1}). \tag{3.13}$$

A probabilidade da seqüência de estados $Q$ pode ser escrita como

$$P(Q|\lambda) = \pi_{q_0} \cdot a_{q_0q_1} \cdot a_{q_1q_2} \cdot \cdots \cdot a_{q_{T-2}q_{T-1}}. \tag{3.14}$$

A probabilidade conjunta de $O$ e $Q$, ou seja, a probabilidade de $O$ e $Q$ ocorrerem simultaneamente, é dada pelo produto dos dois termos acima

$$P(O, Q|\lambda) = P(O|Q, \lambda) \cdot P(Q|\lambda) \tag{3.15}$$

A probabilidade de $O$ pode ser finalmente obtida somando essa probabilidade conjunta de todas as possíveis seqüências de estado $Q$, resultando em

$$\begin{aligned} P(O|\lambda) &= \sum_{\forall Q} P(O|Q,\lambda) \cdot P(Q|\lambda) \\ &= \sum_{\forall Q} \left(\pi_{q_0} b_{q_0}(O_0) \cdot a_{q_0 q_1} b_{q_1}(O_1) \cdots a_{q_{T-2} q_{T-1}} b_{q_{T-1}}(O_{T-1})\right). \end{aligned} \tag{3.16}$$

A Equação (3.16) pode ser interpretada da sequinte maneira. No instante inicial ($t = 0$) o sistema encontra-se no estado $q_0$ com probabilidade $\pi_{q_0}$ e gerando o símbolo $O_0$ com probabilidade $b_{q_0}(O_0)$. É então feita a transição do estado $q_0$ para o estado $q_1$ com probabilidade $a_{q_0 q_1}$, gerando o símbolo $O_1$ com probabilidade $b_{q_1}(O_1)$. O processo continua dessa maneira até que seja feita a última transição, no instante $T-1$, do estado $q_{T-2}$ para o estado $q_{T-1}$ com a probabilidade $a_{q_{T-2} q_{T-1}}$ gerando o símbolo $O_{T-1}$ com a probabilidade $b_{q_{T-1}}(O_{T-1})$.

De acordo com a Equação (3.16) a cada instante $t$ existem $N$ estados que podem ser atingidos, ou seja, existem $N^T$ seqüência de estados possíveis. Para cada seqüência são necessárias $(2T-1)$ multiplicações e o somatório delas requer $(N^T-1)$ adições. Portanto o número de operações necessárias para o cálculo de $P(O|\lambda)$ é da ordem de $O(TN^T)$ sendo computacionalmente inviável até mesmo para pequenos valores de $N$ e $T$ (e.g., para $N = 5$ e $T = 100$ são necessárias aproximadamente $10^{72}$ operações).

Felizmente existe um algoritmo mais eficiente para efetuar esse cálculo chamado *forward-backward* [Baum e Egon, 1967, Baum e Sell, 1968]. Considerando a variável *direta* (do inglês, *forward*) $\alpha_t(i)$ definida como

$$\alpha_t(i) = P(O = O_0 O_1 \cdots O_t, q_t = i|\lambda), \tag{3.17}$$

que representa a probabilidade de observação da seqüência parcial $O = O_0 O_1 \cdots O_t$ até o instante $t$ e do estado $i$ no instante $t$, dado o modelo $\lambda$. Podemos calcular $\alpha_t(i)$ recursivamente da seguinte maneira:

1) Inicialização

$$\alpha_0(i) = \pi_i b_i(O_0), \qquad 0 \leq i < N \tag{3.18}$$

2) Recursão

$$\alpha_{t+1}(j) = \left[\sum_{i=0}^{N-1} (\alpha_t(i) a_{ij})\right] b_j(O_{t+1}), \qquad 0 \leq t \leq T-2, \qquad 0 \leq j < N \tag{3.19}$$

3) Terminação

$$P(O|\lambda) = \sum_{i=0}^{N-1} \alpha_{T-1}(i) \tag{3.20}$$

O passo 1 inicializa as probabilidades *diretas* como a probabilidade conjunta do estado $i$ e da observação inicial $O_0$. O passo da recursão, que pode ser considerado o núcleo do algoritmo, é calculado considerando que o estado atual pode ser atingido de qualquer um dos demais $N$ estados, fator que é levado em conta pelo somatório da Equação (3.19). Uma vez que $\alpha_t(i)$ é a probabilidade de que $O = O_0O_1 \cdots O_t$ sejam observados e o estado no instante $t$ seja $i$, o produto $\alpha_t(i)a_{ij}$ é a probalidade de que seja observado $O = O_0O_1 \cdots O_t$ e o estado $j$ seja atingido no instante $t+1$ a partir do estado $i$. A soma desses produtos para todos os possíveis $N$ estados $i$, $0 \leq i < N$, no instante $t$ resulta na probabilidade de $j$ no instante $t+1$ incluindo todas as observações parciais anteriores. Assim que isso é feito e $j$ é conhecido, fica óbvio que $\alpha_{t+1}(j)$ é obtido contabilizando a probabilidade de ser observado o símbolo $O_{t+1}$ no estado $j$, ou seja, multiplicando o somatório por $b_j(O_{t+1})$. A Equação (3.19) é calculada para todos os estados $j$, $0 \leq j < N$, para um dado $t$; sendo então iterada para $t = 0, 1, 2, \cdots, T-2$. Finalmente o passo 3 fornece o valor de $P(O|\lambda)$ como a soma da probabilidades *direta* terminais $\alpha_{T-1}(i)$.

Esse algoritmo requer $N(N+1)(T-1) + N$ multiplicações e $N(N-1)(T-1) + N$ adições, realizando portanto da ordem de $N^2T$ operações (e.g., para $N = 5$ e $T = 100$ são necessárias aproximadamente 3000 operações, uma economia de 69 ordens de magnitude em relação ao método força bruta apresentado anteriormente).

De maneira similar podemos definir uma variável *reversa* (do inglês, *backward*) $\beta_t(i)$, que apesar de não ser utilizada na solução do Problema 1 é importante na solução dos demais problemas, como

$$\beta_t(i) = P(O_{t+1}O_{t+2} \cdots O_{T-1}|q_t = i, \lambda), \tag{3.21}$$

que é a probabilidade de observação da seqüência parcial de $t+1$ a $T-1$ dado o estado $i$ no instante $t$ e o modelo $\lambda$. Utilizando novamente recursão:

1) Inicialização

$$\beta_{T-1}(i) = 1, \qquad 0 \leq i < N \tag{3.22}$$

2) Recursão

$$\beta_t(i) = \sum_{j=0}^{N-1} a_{ij}b_j(O_{t+1})\beta_{t+1}(j), \qquad t = T-2, T-3, \cdots, 0, \qquad 0 \leq i < N \tag{3.23}$$

Na inicialização $\beta_{T-1}(i)$ é arbitrariamente definido como 1 para todo $i$ [Rabiner, 1989]. O passo 2 considera que para que se tenha passado pelo estado $i$ no instante $t$ e para levar em conta as observações do instante $t+1$ em diante, devem ser considerados todos os estados $j$ possíveis no instante $t+1$ levando em conta a probabilidade de transição de $i$ para $j$ (o termo $a_{ij}$) bem como a observação $O_{t+1}$ no estado $j$ (o termo $b_j(O_{t+1})$). Então devem ser levadas em conta as demais observações parciais a partir do estado $j$ (o termo $\beta_{t+1}(j)$).

O cálculo de $\beta_t(i)$, $0 \leq t < T$, $0 \leq i < N$, também requer da ordem de $N^2T$ operações, como pode ser identificado seguindo raciocínio similar ao do cálculo anterior. Eisner [Eisner, 2002] apresentou uma planilha didática para o aprendizado do algoritmo *forward-backward* que mostra todos os seus passos e apresenta os resultados graficamente.

### 3.3.5 Solução para o Problema 2

Há diversas maneiras de resolver o Problema 2 pois, diferentemente do Problema 1, esse problema, que consiste em encontrar a seqüência de estados ótima associada a uma dada seqüência observada, não possui solução exata. Diversos critérios podem ser adotados para identificar o ótimo. Por exemplo, um critério possível é selecionar os estados $q_t$ que são individualmente mais prováveis. Esse critério maximiza o número esperado de estados corretos individualmente. Entretanto, pode haver problemas com a seqüência de estados resultante. Por exemplo, caso o MOM possua transições com probabilidade zero ($a_{ij} = 0$ para algum $i, j$), a seqüência encontrada como ótima pode não ser uma seqüência válida.

Uma solução possível é alterar o critério de otimalidade. Por exemplo, poderia ser encontrada a seqüência de estados que maximiza o número de pares de estado $(q_t, q_{t+1})$ corretos, ou triplas de estados $(q_t, q_{t+1}, q_{t+2})$ corretas, etc. Apesar desses critérios poderem fazer sentido para algumas aplicações, o critério mais utilizado é encontrar a *melhor* seqüência de estados (caminho), ou seja, maximizar $P(Q|O, \lambda)$. Uma técnica formal para encontrar a melhor seqüência de estados existe, baseada no método da programação dinâmica, denominada algoritmo de Viterbi [Viterbi, 1967].

Esse algoritmo, apesar de importante no contexto dos MOMs, não será descrito por não ter sido utilizado na metodologia proposta nesta dissertação. Para deixar claro a importância desse algoritmo podemos citar o trabalho de Vogler e Metaxas [Vogler e Metaxas, 1998], onde o mesmo foi fundamental.

### 3.3.6 Solução para o Problema 3

O terceiro e mais complexo problema consiste em ajustar os parâmetros do modelo ($A$, $B$, $\pi$) de modo a maximizar a probabilidade de determinada seqüência ser observada. Esse problema não possui solução analítica. Dada uma seqüência de observações finita para o treinamento, não há maneira ótima de estimar os parâmetros do modelo [Rabiner, 1989]. Podemos, entretanto, escolher $\lambda = (A, B, \pi)$ tal que $P(O|\lambda)$ seja maximizado localmente utilizando um algotirmo iterativo como o método de Baum-Welch (ou equivalentemente o EM - *Expectation-Maximization*), ou utilizar técnicas baseadas no gradiente. Nessa subseção será apresentado um método iterativo para selecionar os parâmetros do modelo baseado nos trabalhos iniciais de Baum [Baum e Petrie, 1966, Baum e Egon, 1967, Baum e Sell, 1968, Baum et al., 1970, Baum, 1972], compilado por [Rabiner, 1989].

Para descrever o procedimento de reestimação (melhoria e atualização iterativa) dos parâmetros do MOM será definida $\xi_t(i,j)$, a probabilidade de se estar no estado $i$ no instante $t$ e no estado $j$ no instante $t+1$, dado o modelo e a seqüência de observações.

$$\xi_t(i,j) = P(q_t = i, q_{t+1} = j|O, \lambda) \tag{3.24}$$

Partindo da definição das variáveis *direta* e *reversa* podemos escrever $\xi_t(i,j)$ da seguinte maneira

$$\xi_t(i,j) = \frac{\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}{P(O|\lambda)} \tag{3.25}$$

$$= \frac{\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}{\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)} \tag{3.26}$$

onde o numerador é simplesmente $P(q_t = i, q_{t+1} = j, O|\lambda)$ e a divisão por $P(O|\lambda)$ fornece a probabilidade desejada.

Podemos definir a variável

$$\gamma_t(i) = P(q_t = i|O, \lambda) \tag{3.27}$$

que representa a probabilidade de se estar no estado $i$ no instante $t$ dadas a seqüência de observações $O$ e o modelo $\lambda$. A Equação (3.27) pode ser expressada simplesmente em termos das variáveis *direta* e *reversa* da seguinte maneira

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{P(O|\lambda)} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=0}^{N-1}\alpha_t(i)\beta_t(i)} \tag{3.28}$$

uma vez que $\alpha_t(i)$ leva em conta a seqüência parcial de observações $O = O_0O_1 \cdots O_t$ e o estado $i$ em $t$, enquanto $\beta_t(i)$ leva em conta o restante da seqüência de observações $O_{t+1}O_{t+2} \cdots O_{T-1}$, dado o estado $i$ em $t$. O fator de normalização $P(O|\lambda) = \sum_{i=0}^{N-1} \alpha_t(i)\beta_t(i)$ caracteriza $\gamma_t(i)$ como probabilidade de modo que

$$\sum_{i=0}^{N-1} \gamma_t(i) = 1 \tag{3.29}$$

Dada a seqüência de observações e o modelo, podemos encontrar a relação entre $\gamma_t(i)$ e $\xi_t(i,j)$ somando todos os possíveis $j$,

$$\gamma_t(i) = \sum_{j=0}^{N-1} \xi_t(i,j) \tag{3.30}$$

Se somarmos $\gamma_t(i)$ ao longo do tempo $t$, obteremos uma quantidade que pode ser interpretada como o número de vezes que o estado $i$ é visitado ao longo do tempo, ou seja, o número de transições realizadas a partir do estado $i$ (se excluirmos o instante $t = T - 1$ da soma). Similarmente, a soma de $\xi_t(i,j)$ de $t = 0$ a $t = T - 2$ pode ser interpretada como o número esperado de transições do estado $i$ para o estado $j$. Então temos

$$\sum_{t=0}^{T-2} \gamma_t(i) = \text{número de transições esperado de } i \tag{3.31}$$

$$\sum_{t=0}^{T-2} \xi_t(i,j) = \text{número de transições esperado de } i \text{ para } j \tag{3.32}$$

Utilizando as fórmulas acima (e o conceito de contar ocorrências de eventos) podemos obter um método para reestimar os parâmetros de um MOM. Para reestimar $\pi$, $A$ e $B$ temos

$$\begin{aligned}
\bar{\pi}_i &= \text{número esperado de vezes no estado } i \text{ no instante } (t = 0) = \gamma_0(i) \\
\bar{a}_{ij} &= \frac{\text{número de transições esperado de } i \text{ para } j}{\text{número de transições esperado de } i} \\
&= \frac{\sum_{t=0}^{T-2} \xi_t(i,j)}{\sum_{t=0}^{T-2} \gamma_t(i)} \\
\bar{b}_j(k) &= \frac{\text{número esperado de vezes no estado } j \text{ observando o símbolo } v_k}{\text{número esperado de vezes no estado } j}
\end{aligned}$$

$$= \frac{\sum_{t=0, O_t=v_k}^{T-1} \gamma_t(j)}{\sum_{t=0}^{T-1} \gamma_t(j)}$$

Considerando o modelo atual $\lambda = (A, B, \pi)$, e utilizando-o para computar o lado direito das equações acima, encontramos o modelo reestimado $\bar{\lambda} = (\bar{A}, \bar{B}, \bar{\pi})$. Foi provado por Baum e Sell [Baum e Sell, 1968] que ou o modelo inicial $\lambda$ define um ponto crítico na verossimilhança e $\bar{\lambda} = \lambda$ ou o modelo reestimado apresenta verossimilhança maior e $P(O|\bar{\lambda}) > P(O|\lambda)$, ou seja, o modelo reestimado tem maior probabilidade de produzir a seqüência $O$.

Dessa maneira podemos utilizar essas equações iterativamente para aumentar a probabilidade de $O$ ser observado pelo modelo até atingir o limite. O resultado final desse procedimento é denominado estimativa de máxima verossimilhança (*maximum likelihood estimate*) do MOM.

Este capítulo apresentou as principais características das cadeias de Markov e MOMs, que são os sistemas a eventos discretos utilizados na metodologia proposta nesta dissertação. Esta metodologia será apresentada no próximo capítulo.

# Capítulo 4

# Metodologia

A metodologia proposta nesta dissertação se baseia na utilização de algoritmos de visão computacional amplamente conhecidos e utilizados e ferramentas estocásticas de reconhecimento de linguagens sensíveis ao contexto. Cada comando é subdividido em símbolos simples, mais fáceis de serem reconhecidos por algoritmos de visão computacional do que um único símbolo complexo. É utilizado um sistema a eventos discretos para interpretar então a seqüência de símbolos. Com o objetivo de simplificar o sistema de visão e tornar o reconhecimento mais robusto foram considerados apenas gestos simples, como movimentos nas diversas direções.

O sistema considerado é apresentado na Figura 4.1 sendo composto pelo módulo de Aquisição de Imagens, Sistema de Visão Computacional e pelo Sistema a Eventos Discretos. Os módulos do sistema tem entradas e saídas bem definidas, como pode ser observado. Esses módulos serão descritos nas próximas seções.

## 4.1 Sistema de Visão Computacional

O sistema de visão computacional é responsável por processar cada quadro adquirido pela câmera fornecendo o vetor deslocamento correspondente ao movimento do objeto

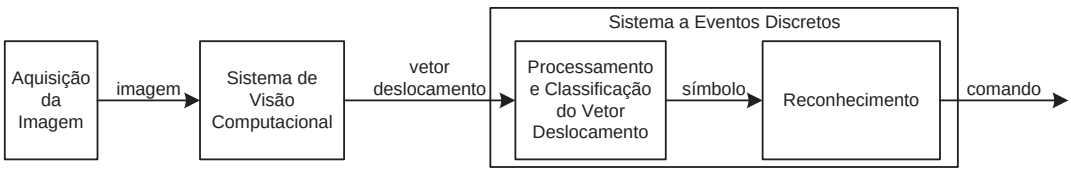


Figura 4.1: Diagrama de blocos do sistema utilizado.

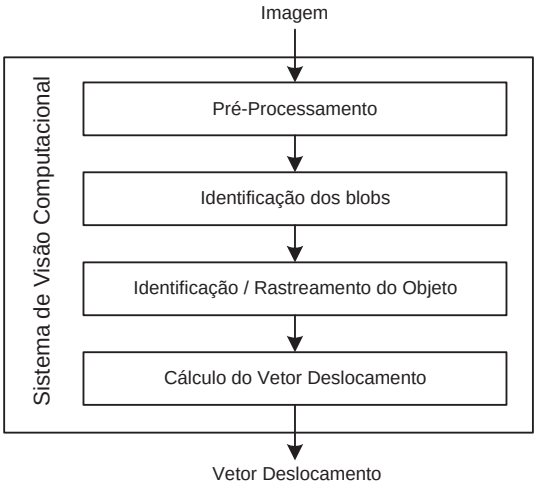


Figura 4.2: Sistema de Visão Computacional - diagrama de blocos.

de interesse na cena, permitindo assim a identificação da trajetória do gesto. O sistema projetado foi o mais simples possível que permitisse a realização desse trabalho, uma vez que o foco do estudo é o processamento das informações e não a aquisição das informações em si.

O sistema de visão é baseado na segmentação por cor, assim como diversos sistemas utilizados no reconhecimento de gestos como, por exemplo, [Hong et al., 2000a, Davis e Shah, 1994a, Siskind e Morris, 1996]. O funcionamento do sistema de visão é apresentado na Figura 4.2 e cada um dos sub-módulos será descrito nas subseções a seguir. A entrada do sistema é a imagem adquirida de tamanho $w \times h$ onde $w$ é a largura e $h$ a altura, ambos em pixels. A saída do sistema é o vetor deslocamento do objeto rastreado, que indica o gesto que está sendo realizado.

O objetivo principal do sistema foi a robustez, embora erros na identificação possam ser corretamente tratados pelo processamento do módulo a eventos discretos.

### 4.1.1 Pré-Processamento

A imagem é pré-processada visando minimizar os ruídos de alta freqüência e também tornar as cores do objeto rastreado um pouco mais uniformes para facilitar a sua identificação. Foram executados dois passos:

1. Conversão do formato RGB para HSV.

2. Borramento Gaussiano (*Gaussian blur*) utilizando um *kernel* de tamanho cinco, tamanho comumente usado na literatura.

A utilização do formato HSV permite uma melhor segmentação por cor, uma vez que a informação de cor é separada da informação de sua intensidade. A conversão do espaço de cores RGB para HSV exige que cálculos de complexidade constante sejam realizados para converter cada pixel, tendo portanto complexidade $O(w \times h)$ [Fairchild, 2005].

O borramento Gaussiano exige que o *kernel* seja convolvido com a imagem. Como o nosso *kernel* tem tamanho constante podemos considerar que a complexidade desta operação também é $O(w \times h)$.

### 4.1.2 Identificação dos blobs

Os *blobs* são identificados pela cor, representada pelas coordenadas H e S do espaço de cores HSV e uma tolerância, que é independente para o valor de H e de S. Caso a diferença entre os valores do pixel da imagem em questão e a cor passada como parâmetro sejam menores que a tolerância o pixel faz parte de um *blob*. Para identificar a cor bastaria o parâmetro H, porém para evitar que regiões muito claras ou muito escuras sejam identificadas, como por exemplo um verde com pouca ou muita saturação, o parâmetros S também foi considerado de modo a eliminar os valores muito baixos e muito altos.

Os *blobs* encontrados são então rotulados, identificando assim os componentes conectados. Para essa tarefa é utilizado o algoritmo de coloração de *blobs* apresentado em [Ballard e Brown, 1982] onde cada pixel da imagem é visitado somente uma vez. O centro de massa e o tamanho de cada *blob*, ou seja, sua área, também são calculados durante a identificação. A complexidade dessa etapa é da ordem de $O(w \times h)$.

### 4.1.3 Identificação/Rastreamento do objeto

Para o primeiro quadro de uma seqüência, o objeto de interesse é considerado o *blob* de maior área. Para os demais quadros, é feito o rastreamento desse *blob* utilizando um algoritmo simples do tipo preditor-corretor linear. O próximo centro de massa é previsto considerando a velocidade de deslocamento constante, ou seja, que o vetor deslocamento entre o quadro atual e o anterior será igual àquele entre o quadro anterior e o anterior a ele. A área identificada como o objeto a ser rastreado é selecionada baseada na área do *blob* e proximidade da localização esperada. A utilização desses dois parâmetros torna o algoritmo de rastreamento robusto. O peso de cada um foi quantificado da seguinte maneira, o peso de cada *blob* é dado pela sua área menos duas vezes a distância da localização esperada (área $- 2 \times dist(\text{centro}, \text{centroPrevisto})$).

Para o cálculo da complexidade desse passo iremos considerar o pior caso, apesar de improvável, no qual são identificados $w \times h$ componentes conectados. Todos eles serão avaliados de acordo com o critério descrito acima, cujos cálculos podem ser realizados em tempo constante, resultando em uma complexidade de $O(w \times h)$ para esse passo no pior caso. Porém o tempo de execução esperado é bem menor.

Caso o objeto não seja identificado em algum quadro é utilizada a parte preditiva do algoritmo de rastreamento para tentar localizá-lo no próximo quadro, a parte corretiva somente é utilizada quando o objeto é localizado. Isso permite robustez ante quadros em que o objeto não é identificado, pois possibilita uma melhor predição nos quadros posteriores independentemente da identificação no quadro atual, tratando assim oclusões.

O algoritmo de rastreamento também considera a aceleração, que é calculada como a diferença dos vetores deslocamento entre o quadro sendo processado e o anterior. Caso o comprimento do vetor aceleração seja maior que o limiar definido, o vetor deslocamento calculado será ignorado, não sendo computado. Essa medida leva em conta as limitações físicas para a aceleração do objeto tornando ainda mais robusto o algoritmo.

### 4.1.4 Cálculo do vetor deslocamento

Caso o centro de massa calculado seja considerado válido de acordo com as condições descritas na subseção anterior o vetor deslocamento é calculado e transmitido para o módulo do sistema a eventos discretos para ser processado. O vetor deslocamento consiste na diferença entre o centro de massa do *blob* selecionado no quadro anterior e o centro de massa do *blob* selecionado no quadro atual. Após a identificação do *blob* no passo anterior é feita a subtração de seu centro de massa do centro de massa do *blob* identificado no quadro anterior (que foi armazenado), operação de ordem de complexidade $O(1)$.

### 4.1.5 Considerações sobre a ordem de complexidade

Somando a complexidade de todos os passos do nosso sistema de visão computacional temos que a ordem de complexidade do processamento realizado por esse sistema é linear no tamanho da imagem $O(w \times h)$.

## 4.2 Máquina de Estados Finitos

A princípio uma máquina de estados finitos (MEF) simples pode ser utilizada para o reconhecimento de gestos. De maneira intuitiva podemos mapear um conjunto de gestos em uma MEF considerando os gestos como eventos, responsáveis pelas transições. Dessa

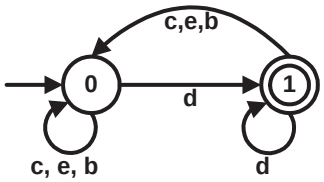


Figura 4.3: MEF que reconhece o gesto *direita* (d).

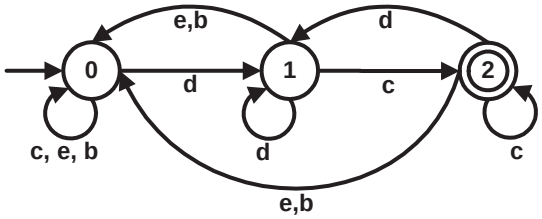


Figura 4.4: MEF que reconhece a seqüência de gestos *direita* (d) e *para cima* (c).

maneira, considerando os gestos *direita* (d), *esquerda* (e), *para cima* (c) e *para baixo* (b) como o alfabeto, uma MEF para reconhecer o gesto *direita* (d) poderia ser construída como na Figura 4.3. A transição do estado 0 para o estado 1 seria ativada pelo evento *direita*, todos os demais eventos iriam manter a MEF no estado 0. No estado 1 apenas o evento *direita* faz com que a MEF permaneça nesse estado, todos os demais fazem com que a MEF retorne para o estado 0.

Essa máquina pode ser estendida para reconhecer a seqüência de dois gestos, como mostrado na Figura 4.4. Neste caso, a MEF segue os mesmos princípios da descrita anteriormente, porém para a seqüência de gestos *direita* e *para cima*. Adicionando mais um gesto obtemos a máquina apresentada na Figura 4.5 que reconhece a seqüência de gestos *direita*, *para cima* e *esquerda*.

Seguindo essa metodologia de construção teremos uma MEF distinta para cada seqüência a ser reconhecida, sendo possível construir MEFs para reconhecer quaisquer seqüências de gestos. O número de estados de cada MEF é dado por

$$N = (\text{número de gestos}) + 1. \tag{4.1}$$

Analisando esse modelo podemos ver que qualquer erro na identificação de um gesto fará com que o reconhecimento não tenha sucesso. Esse fator nos levou em busca de

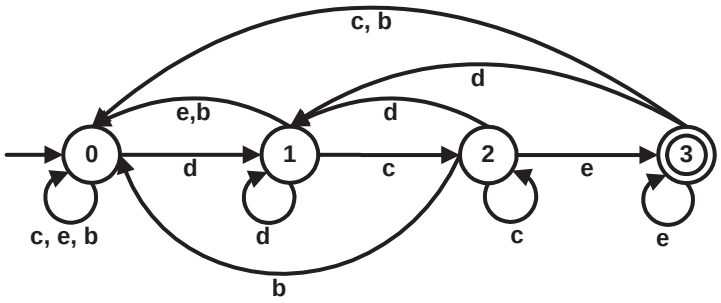


Figura 4.5: MEF que reconhece a seqüência *direita* (d), *para cima* (c) e *esquerda* (e).

alternativas estocásticas, que serão apresentadas nas próximas seções.

## 4.3 Cadeias de Markov

A utilização de cadeias de Markov visa adicionar as vantagens de um sistema estocástico à metodologia apresentada na seção anterior. Uma cadeia de Markov pode ser construída para cada comando a ser reconhecido de maneira similar às MEFs da Seção 4.2. Um exemplo de cadeia de Markov é apresentado na Figura 4.6. A probabilidade das transições a cada instante é determinada dinamicamente de acordo com o ângulo do vetor deslocamento identificado pelo sistema de visão computacional. Os gestos considerados (alfabeto) são *direita* (d), *para cima* (c), *esquerda* (e) e *para baixo* (b). A cada quadro são calculadas probabilidades de ocorrência para cada um dos gestos, de acordo com o vetor deslocamento fornecido pelo sistema de visão computacional (Seção 4.1). Essas probabilidades atualizam a probabilidade de cada estado da cadeia de Markov e, caso a probabilidade do último estado da cadeia atinja um certo limiar o comando é considerado identificado.

Num primeiro instante considerou-se calcular essas probabilidades a partir do seno e cosseno do vetor, levando-se em conta apenas o seu ângulo, desde que seu comprimento fosse maior que um limiar pré-estabelecido. Esta técnica implica em sempre haver duas probabilidades maiores ou iguais a zero e pelo menos duas iguais a zero, para o caso de um alfabeto de quatro gestos. Essas probabilidades iguais a zero são consideradas destrutivas para o sistema, pois levam a probabilidade dos estados da cadeia para zero, o que implica que uma identificação errada pode invalidar toda uma seqüência. Dessa maneira é preferível utilizar inferência bayesiana [Berger, 1993] para resolver esse problema.

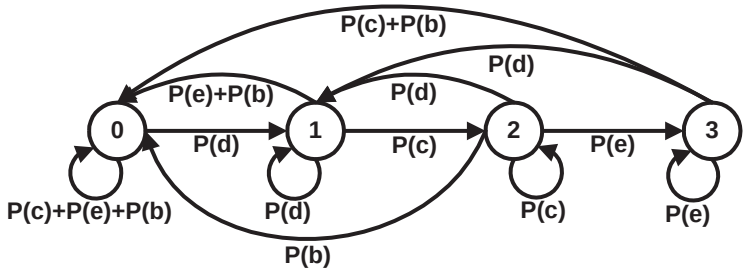


Figura 4.6: Cadeia de Markov que reconhece a seqüência *direita*, *para cima*, *esquerda*. A probabilidade de cada transição $a_{ij}$ é calculada a partir das probabilidades de cada gesto.

As subseções seguintes irão apresentar a formulação matemática e detalhar como é realizado o processamento contínuo.

### 4.3.1 Formulação

O estado atual da cadeia de Markov é representado por um vetor de tamanho $N$, onde $N$ é o número de estados da cadeia. Cada elemento do vetor armazena a probabilidade atual do sistema estar no estado correspondente. Para obter as probabilidades de cada estado no instante seguinte basta multiplicar a matriz de probabilidades de transição $A$ (Seção 3.2) pelo vetor do estado atual, como mostra a Equação (4.2).

$$\begin{bmatrix} P[q_{t+1}=0] \\ \vdots \\ P[q_{t+1}=N-1] \end{bmatrix} = \begin{bmatrix} a_{00}(k) & \cdots & a_{0N-1}(k) \\ \vdots & \ddots & \vdots \\ a_{N-10}(k) & \cdots & a_{N-1N-1}(k) \end{bmatrix} \times \begin{bmatrix} P[q_t=0] \\ \vdots \\ P[q_t=N-1] \end{bmatrix} \quad (4.2)$$

### 4.3.2 Processamento

O algoritmo utilizado processa os quadros seqüencialmente executando os seguintes passos:

1. Validação: valida se o vetor deslocamento deve ser processado de acordo com o seu comprimento. Vetores menores que um limiar definido são descartados.

2. Obtenção das probabilidades de cada gesto: são obtidas probabilidade para cada um dos gestos de acordo com a orientação e comprimento do vetor deslocamento, construindo a matriz de probabilidades de transição $A$.

3. Atualização das probabilidades: Cada estado, de cada cadeia, possui uma probabilidade que é atualizada a cada quadro de acordo com a Equação (4.2).

4. Verificação do comando identificado: caso o último estado de algum dos autômatos atinja probabilidade maior que o limiar definido ($P[q_{t+1} = N - 1] > \text{limiar}$) o comando é considerado identificado e a respectiva ação deve ser tomada. Nesse caso a probabilidade de todos os autômatos é reinicializada.

Cada cadeia de Markov é construída de acordo com o comando a ser reconhecido, sendo que à medida que o estado mais provável se torna o mais à direita o reconhecimento do comando se torna mais próximo.

As cadeias de Markov não modelam a duração de cada gesto. Apesar de permitirem a repetição do mesmo gesto por meio da transição de um estado para ele mesmo, não é exigida uma duração mínima de cada gesto, podendo cada gesto estar presente em apenas um quadro e o comando ser considerado identificado. Isso torna o sistema mais susceptível a erros (falsos positivos), uma vez que uma determinada seqüência de ruídos pode até ser considerada um comando. Esse problema nos motivou a buscar um modelo que considerasse uma duração mínima que pudesse ser aprendida por meio de treinamento e que fosse robusto a variações nessa duração. A próxima seção descreve como MOMs podem ser utilizados para atingir tal objetivo.

## 4.4 Modelos Ocultos de Markov

Antes da utilização para o reconhecimento, os MOMs exigem uma fase de treinamento. A princípio os MOMs poderiam ser treinados de duas maneiras distintas para o reconhecimento de gestos: treinamento do comando completo e treinamento de cada gesto independente. Neste trabalho propõe-se o treinamento de cada gesto independente e as próximas subseções detalham essa escolha. Todas as demais etapas são comuns independentemente da maneira de treinamento. A próxima subseção irá descrever as características comuns das duas abordagens e as subseções seguintes apresentarão as especifidades de cada uma delas.

Como descrito anteriormente o módulo do sistema a eventos discretos têm como entrada o vetor deslocamento do gesto, sendo esse portanto o evento observado dos MOMs.

### 4.4.1 Formulação

Como descrito na Seção 3.3.2 um MOM é definido pelo número de estados $N$, número de símbolos observáveis em cada estado $M$, a matriz de probabilidades de transição $A$, a matriz de probabilidades de observação dos símbolos nos estados $B$ e a distribuição inicial $\pi$. Esses parâmetros são definidos da seguinte maneira nesse trabalho:

$N$ : o número de estados é definido de acordo com o tamanho das seqüências utilizadas para treinamento. O valor de $N$ é o mínimo entre a metade do tamanho médio das seqüências e o tamanho da menor seqüência, dessa maneira não há seqüência com menos elementos que o número de estados e o tempo médio esperado em cada estado é dois;

$M$ : o número de símbolos é definido a partir da classificação do ângulo do vetor deslocamento em intervalos. A opção por utilizar intervalos será explicada a seguir.

$A$ : a matriz de probabilidades de transição é definida pelo treinamento, sendo inicializada de acordo com a topologia selecionada. A inicialização é realizada distribuindo igualmente as probabilidades entre as transições válidas.

$B$ : a matriz de probabilidades de observação dos símbolos nos estados é definida no treinamento, sendo inicializada de forma homogênea $b_j(k) = \frac{1}{M}$, $0 \leq j < N$, $0 \leq k < M$.

$\pi$ : O estado inicial do MOM é sempre o primeiro, de modo que $\pi_0 = 1$ e $\pi_i = 0$, $\forall\, i \neq 0$. Essa característica é fruto da topologia selecionada que será descrita ainda nesta seção.

Para reduzir o número de elementos do alfabeto, os ângulos são agrupados em intervalos. Desse modo a quantidade de eventos possíveis é reduzida e o sistema torna-se mais robusto, uma vez que a utilização direta dos ângulos tornaria o sistema muito mais sensível e exigiria uma quantidade muito maior de dados para treinamento. Caso fosse considerado apenas o ângulo (inteiro) do vetor teríamos 360 símbolos, o que resultaria em uma probabilidade média de aproximadamente $2{,}8 \times 10^{-3}$ para cada símbolo, já com oito símbolos esse valor sobe para $1{,}25 \times 10^{-1}$. Os intervalos considerados são apresentados na Figura 4.7.

A topologia proposta é a esquerda para direita sem omissões (*left right with no skips*), apresentada na Figura 4.8, dado que a mesma já foi utilizada com sucesso tanto no reconhecimento de gestos [Siskind e Morris, 1996, Vogler e Metaxas, 1998] quanto no

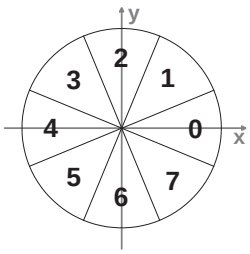


Figura 4.7: Classificação dos ângulos de deslocamento em intervalos. Por exemplo, os ângulos entre $\pi/8$ e $3\pi/8$ correspondem ao intervalo 1.

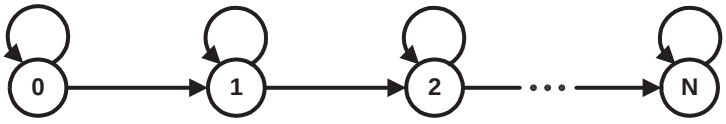


Figura 4.8: Topologia esquerda para direita sem omissões (*left right with no skips*).

de fala [Rabiner e Juang, 1986, Young et al., 2005]. De acordo com Siskind e Morris [Siskind e Morris, 1996] modelos não ergódicos generalizam muito melhor para novas observações. Contudo deve ser considerado que a metodologia descrita é independente da topologia do MOM.

O número de estados é escolhido de forma a maximizar o poder de generalização do modelo. Considerando $T_{min}$ o tamanho da menor seqüência de treinamento e $T_{avg}$ o tamanho médio das seqüências de treinamento, o número de estados do modelo foi calculado da seguinte maneira

$$N = min\left(T_{min}, \frac{T_{avg}}{2}\right). \tag{4.3}$$

Dessa maneira, em média, teríamos duas vezes mais estados que observações. O tamanho mínimo deve ser considerado para que não tenhamos um modelo com mais estados que observações, o que impediria o reconhecimento dessa seqüência devido à metologia utilizada (vide Seção 4.4.3).

### 4.4.2 Treinamento e Preparação

A preparação do sistema para o processamento de quadros e reconhecimento de gestos se inicia com a configuração dos comandos que deverão ser reconhecidos e o treinamento dos MOMs para o reconhecimento de cada um deles. Para cada comando é criado um MOM

distinto. A metodologia de treinamento será descrita detalhadamente nas subseções posteriores para os casos de treinamento independente para cada gesto e treinamento do comando completo. A função do treinamento dos MOMs é ajustar os parâmetros do modelo para o reconhecimento de determinada(s) seqüência(s). Os parâmetros $A$ e $B$ são ajustados de acordo com as seqüências de observações utilizadas para treinamento.

### 4.4.3 Processamento

A cada quadro um novo símbolo é gerado (pode ser que alguns quadros não gerem evento) e este deve ser processado pelos MOMs responsáveis pelo reconhecimento juntamente com símbolos gerados anteriormente que fazem parte desse comando. Porém, identificar o início de um comando é uma tarefa difícil e portanto são determinados tamanhos mínimo, $T_{min}$, e máximo, $T_{max}$, para as seqüências a serem consideradas. Esses tamanhos são definidos na fase de treinamento como os tamanhos mínimo e máximo das seqüências utilizadas para treinamento do modelo.

À medida que cada símbolo é gerado, o mesmo é armazenado em um *buffer* que é gerenciado da seguinte maneira: cada símbolo gerado (por um quadro) é armazenado no *buffer*, caso o tamanho do *buffer* seja maior que o tamanho máximo ($T_{max}$) o primeiro gesto armazenado é descartado. No caso do reconhecimento de algum comando o *buffer* é esvaziado por completo, evitando assim que um único gesto do usuário possa fazer parte de mais de um comando. A cada instante todas as seqüências de tamanho maior ou igual a $T_{min}$ até o tamanho do *buffer*, terminadas no último símbolo, são processadas. A Tabela 4.1 exemplica o funcionamento do *buffer*.

Podemos então descrever os passos realizados para o reconhecimento de gestos pelos MOMs:

1. O símbolo recebido é identificado e armazenado no *buffer*.

2. As seqüências do *buffer* terminadas no último símbolo e maiores que $T_{min}$ são avaliadas por todos os MOMs.

3. Caso um ou mais MOMs indiquem verossimilhança maior que o limiar para qualquer seqüência significa que um comando foi identificado nesse quadro. O comando de maior verossimilhança é considerado o identificado.

4. Caso tenha sido identificado algum comando nesse quadro o *buffer* é esvaziado.

Dado o modelo, a probabilidade de uma seqüência de observações ser reconhecida é calculada utilizando procedimento similar ao descrito na Seção 3.3.4, sendo também

| quadro | símbolo | conteúdo do *buffer* | seqüências processadas | houve comando identificado? |
|---|---|---|---|---|
| 1 | 4 | 4 | $\emptyset$ | não |
| 2 | 4 | 44 | $\emptyset$ | não |
| 3 | 4 | 444 | $\emptyset$ | não |
| 4 | 3 | 4443 | {4443} | não |
| 5 | 2 | 44432 | {4432, 44432} | não |
| 6 | 2 | 444322 | {4322, 44322, 444322} | não |
| 7 | 1 | 443221 | {3221, 43221, 443221} | sim |
| 8 | 0 | 0 | $\emptyset$ | não |
| 9 | 0 | 00 | $\emptyset$ | não |

Tabela 4.1: Exemplo de comportamento do *buffer* considerando um tamanho mínimo de seqüência de 4 e máximo de 6. Para cada quadro é apresentado o símbolo identificado, o conteúdo do *buffer*, o conjunto de seqüências processadas pelo MOM e se houve ou não algum comando identificado nesse quadro.

baseado no algoritmo *forward-backward*. A diferença é que apenas a probabilidade de gerar o modelo terminando no último estado é considerada. Assim, em vez de utilizar a Equação (3.20) a probabilidade de reconhecimento considerada é dada por

$$P(O, q_{T-1} = N-1|\lambda) = \alpha_{T-1}(N-1), \tag{4.4}$$

sendo que se o valor de $\alpha_{T-1}(N-1)$ for maior que o limiar considerado o gesto será considerado reconhecido.

Essa alteração torna o reconhecimento mais robusto, exigindo que apenas seqüências inteiramente parecidas com as de treinamento sejam consideradas reconhecidas. Considerando por exemplo que um MOM foi treinado com a seqüência $O_1 = 012345$, caso o procedimento usual fosse utilizado o MOM reconheceria seqüências como por exemplo $O_2 = 000000$ ou $O_3 = 011111$, porém a Equação (4.4) faz com que essas seqüências sejam rejeitadas.

### 4.4.4 Treinamento do MOM com o Comando Completo

O treinamento do MOM com o comando completo é o treinamento direto das observações do comando. A partir de um conjunto de execuções do comando o MOM é treinado. Ele é mais simples, porém possui algumas desvantagens em relação ao treinamento independente de cada gesto, que serão apresentadas na próxima seção.


Figura 4.9: MOM do tipo esquerda para direita sem omissões (*left right with no skips*) de dois estados.

### 4.4.5 Treinamento Independente de Cada Gesto

O treinamento de um MOM para cada gesto independente reduz a quantidade de dados necessários para treinamento e permite a generalização para a construção de comandos utilizando qualquer combinação de gestos treinados. Pode ser reconhecido qualquer comando que possa ser construído concatenando os MOMs dos gestos individuais treinados. O treinamento de um MOM diretamente para cada comando exige que o comando a ser treinado seja repetido várias vezes, o que pode resultar em uma massa significativa de dados, uma vez que cada comando é composto por diversos gestos. Além disso, o treinamento por comando também dificulta a generalização, uma vez que torna-se necessário realizar todo o processo de treinamento para cada novo comando.

A metodologia utilizada será apresentada por meio de um exemplo. Um MOM será construído e treinado para reconhecer uma seqüência de gestos (palavra) simples. O exemplo será realizado utilizando um modelo reduzido para facilitar o entendimento e a apresentação do mesmo, entretanto ele deve ser entendido como sendo diretamente aplicável a modelos complexos, demonstrando o poder dos MOMs.

O primeiro passo é o treinamento individual de um MOM para cada um dos gestos, como será mostrado a seguir. Esses MOMs serão posteriormente concatenados formando um único MOM capaz de reconhecer um comando definido por uma seqüência de gestos. Esse único MOM formado pela concatenação requer uma nova fase de treinamento, ainda assim o treinamento individual é importante pois devemos considerar que o método de reestimação dos parâmetros fornece valores correspondentes a um máximo local, e já foi comprovado que uma boa estimativa inicial para os parâmetros pode ter uma grande influência no resultado final [Rabiner, 1989].

Inicialmente o modelo do tipo esquerda para direita sem omissões (*left right with no skips*) apresentado na Figura 4.9 será treinado para reconhecer o gesto *direita* utilizando as seqüências de treinamento da Tabela 4.2, que contém várias possíveis seqüências observadas do gesto *direita* (cada observação correponde a um intervalo na Figura 4.7). O número de estados foi definido utilizando a Equação (4.3).

| $O_n$ |
|---|
| 0 1 0 0 |
| 0 0 1 1 |
| 7 0 0 7 |
| 7 0 1 0 |
| 1 1 0 1 |
| 1 0 7 7 |
| 7 7 0 0 |
| 0 7 0 1 |
| 0 0 7 7 |

Tabela 4.2: Seqüências de observações utilizadas no treinamento do MOM para o reconhecimento do gesto *direita*.

Consideraremos para os nossos MOMs o seguinte estado inicial (Seção 4.4.1):

$$A = \begin{pmatrix} 0{,}5 & 0{,}5 \\ 0{,}0 & 1{,}0 \end{pmatrix} \tag{4.5}$$

$$B = \begin{pmatrix} 0{,}125 & 0{,}125 & 0{,}125 & 0{,}125 & 0{,}125 & 0{,}125 & 0{,}125 & 0{,}125 \\ 0{,}125 & 0{,}125 & 0{,}125 & 0{,}125 & 0{,}125 & 0{,}125 & 0{,}125 & 0{,}125 \end{pmatrix} \tag{4.6}$$

$$\pi = (1{,}0 \quad 0{,}0) \tag{4.7}$$

Após o treinamento com as seqüências apresentadas na Tabela 4.2 utilizando o algoritmo de treinamento implementado pela biblioteca GHMM [Schliep et al., 2005] obtemos o conjunto de parâmetros apresentado a seguir, sendo que o valor de $\pi$ não é afetado pelo procedimento de treinamento.

$$A = \begin{pmatrix} 0{,}61 & 0{,}39 \\ 0{,}00 & 1{,}00 \end{pmatrix} \tag{4.8}$$

$$B = \begin{pmatrix} 0{,}48 & 0{,}20 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}32 \\ 0{,}46 & 0{,}31 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}23 \end{pmatrix} \tag{4.9}$$

Esse modelo é capaz de gerar as seqüências utilizadas para treinamento com probabilidades em torno de 1,0 a $4{,}0 \times 10^{-2}$ enquanto para o modelo original esse valor era aproximadamente $8{,}0 \times 10^{-4}$.

Procedimento similar foi executado para os gestos *para cima* e *esquerda* considerando as seqüências de treinamento apresentadas, respectivamente, na Tabela 4.3 e na

| $O_n$ |
|---|
| 1 1 2 2 |
| 1 2 3 1 |
| 3 3 2 2 |
| 2 3 2 3 |
| 3 2 1 3 |
| 2 1 2 1 |
| 2 1 2 3 |
| 1 3 1 2 |
| 1 1 2 1 |

Tabela 4.3: Seqüências de observações utilizadas no treinamento do MOM para reconhecimento do gesto *para cima*.

Tabela 4.4. Para o MOM do gesto *para cima* foram obtidos os parâmetros

$$A = \begin{pmatrix} 0{,}58 & 0{,}42 \\ 0{,}00 & 1{,}00 \end{pmatrix} \tag{4.10}$$

$$B = \begin{pmatrix} 0{,}00 & 0{,}43 & 0{,}34 & 0{,}23 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 \\ 0{,}00 & 0{,}29 & 0{,}44 & 0{,}27 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 \end{pmatrix} \tag{4.11}$$

enquanto para o gesto *esquerda* foram obtidos os parâmetros

$$A = \begin{pmatrix} 0{,}84 & 0{,}16 \\ 0{,}00 & 1{,}00 \end{pmatrix} \tag{4.12}$$

$$B = \begin{pmatrix} 0{,}00 & 0{,}00 & 0{,}00 & 0{,}37 & 0{,}26 & 0{,}37 & 0{,}00 & 0{,}00 \\ 0{,}00 & 0{,}00 & 0{,}00 & 0{,}01 & 0{,}73 & 0{,}26 & 0{,}00 & 0{,}00 \end{pmatrix}. \tag{4.13}$$

Supondo agora que temos um comando composto pelos gestos *direita*, *para cima* e *esquerda*, podemos simplesmente concatenar os MOMs dos gestos treinados individualmente para criar um MOM capaz de reconhecer este comando. Para essa união dos MOMs poderia ser utilizando um tipo de estado conhecido como não-emissor [Young et al., 2005] ou silencioso [Schliep et al., 2005] que possui a característica de não gerar evento, ou seja, nesse estado nenhum símbolo é observado. Porém, por simplicidade e devido à falta de suporte da biblioteca utilizada a este tipo de estado, optou-se por apenas concatenar os MOMs unindo o último estado de um ao primeiro estado de outro.

Dessa maneira a topologia do MOM se torna a da Figura 4.10 consistindo de três

| $O_n$ |
|---|
| 3 4 4 4 |
| 3 4 5 4 |
| 5 5 4 4 |
| 3 3 3 3 |
| 4 4 5 3 |
| 5 3 4 5 |
| 4 4 5 5 |
| 5 5 5 3 |

Tabela 4.4: Seqüências de observações utilizadas no treinamento do MOM para reconhecimento do gesto *esquerda*.

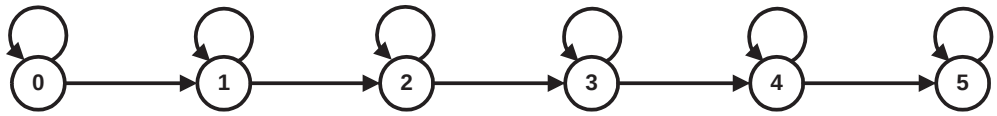


Figura 4.10: MOM com seis estados, resultante da união dos três MOMs de dois estados treinados.

MOMs como o da Figura 4.9 conectados. As matrizes de parâmetros se tornam

$$A = \begin{pmatrix} 0{,}61 & 0{,}39 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 \\ 0{,}00 & 0{,}50 & 0{,}50 & 0{,}00 & 0{,}00 & 0{,}00 \\ 0{,}00 & 0{,}00 & 0{,}58 & 0{,}42 & 0{,}00 & 0{,}00 \\ 0{,}00 & 0{,}00 & 0{,}00 & 0{,}50 & 0{,}50 & 0{,}00 \\ 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}84 & 0{,}16 \\ 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 & 1{,}00 \end{pmatrix} \tag{4.14}$$

$$B = \begin{pmatrix} 0{,}48 & 0{,}20 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}32 \\ 0{,}46 & 0{,}31 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}23 \\ 0{,}00 & 0{,}43 & 0{,}34 & 0{,}23 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 \\ 0{,}00 & 0{,}29 & 0{,}44 & 0{,}27 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 \\ 0{,}00 & 0{,}00 & 0{,}00 & 0{,}37 & 0{,}26 & 0{,}37 & 0{,}00 & 0{,}00 \\ 0{,}00 & 0{,}00 & 0{,}00 & 0{,}01 & 0{,}73 & 0{,}26 & 0{,}00 & 0{,}00 \end{pmatrix}. \tag{4.15}$$

A união dos MOMs treinados para o reconhecimento de gestos requer que sejam atribuídas probabilidades às transições de conexão, que automaticamente definem a probabilidade de permanência no último estado de cada MOM individual. Foram seleciona-

dos empiricamente os valores de 0,50 e 0,50[1], como pode ser visto nos elementos $a_{11}$, $a_{12}$, $a_{33}$ e $a_{34}$ da matriz $A$ (Equação (4.14)). Para ajustar essas probabilidades da matriz é realizado um novo treinamento, dessa vez para o MOM composto. Esse treinamento é realizado com o produto cartesiano das observações de cada gesto, concatenando as seqüências de treinamento individuais, processo que resulta em um conjunto de dados considerável.

O treinamento em conjunto dos três MOMs é utilizado também para evitar um problema do reconhecimento contínuo, que são os efeitos da coarticulação. A coarticulação, identificada inicialmente nos sistemas de reconhecimento de fala, implica que a pronúncia de uma palavra é influenciada pelas predecessoras e seguintes, aplicando-se analogamente para o reconhecimento contínuo de gestos [Vogler e Metaxas, 1998], ou seja, a execução de um gesto é influenciada pelos gestos predecessores e posteriores. Ao alterar os parâmetros dessa conexão os MOMs se tornam dependentes do contexto, uma vez que os valores das transições do seu último estado são alterados.

Após essa segunda etapa de treinamento as matrizes de parâmetros resultantes foram

$$A = \begin{pmatrix} 0{,}51 & 0{,}49 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 \\ 0{,}00 & 0{,}48 & 0{,}52 & 0{,}00 & 0{,}00 & 0{,}00 \\ 0{,}00 & 0{,}00 & 0{,}49 & 0{,}51 & 0{,}00 & 0{,}00 \\ 0{,}00 & 0{,}00 & 0{,}00 & 0{,}46 & 0{,}54 & 0{,}00 \\ 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}82 & 0{,}18 \\ 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 & 1{,}00 \end{pmatrix} \tag{4.16}$$

$$B = \begin{pmatrix} 0{,}50 & 0{,}20 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}30 \\ 0{,}45 & 0{,}29 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}26 \\ 0{,}00 & 0{,}42 & 0{,}30 & 0{,}27 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 \\ 0{,}00 & 0{,}35 & 0{,}52 & 0{,}13 & 0{,}00 & 0{,}00 & 0{,}00 & 0{,}00 \\ 0{,}00 & 0{,}00 & 0{,}00 & 0{,}44 & 0{,}23 & 0{,}33 & 0{,}00 & 0{,}00 \\ 0{,}00 & 0{,}00 & 0{,}00 & 0{,}01 & 0{,}68 & 0{,}31 & 0{,}00 & 0{,}00 \end{pmatrix}. \tag{4.17}$$

A partir da matriz $B$ podemos inferir os símbolos que cada estado é capaz de gerar, apresentados na Tabela 4.5. Essa informação nos permite ter uma idéia das seqüências que podem ser geradas (reconhecidas) por esse modelo.

A Tabela 4.6 apresenta as probabilidades de reconhecimento para 18 seqüências de observações utilizadas para validação. Na tabela são apresentadas tanto a probabilidade

[1] Foram realizados experimentos utilizando outros valores como, por exemplo, 0,35 e 0,65, e foi identificado que o valor inicial dessas transições não afeta significativamente o resultado após o treinamento.

| Estado | Símbolos |
|---|---|
| 0 | { 0, 1, 7 } |
| 1 | { 0, 1, 7 } |
| 2 | { 1, 2, 3 } |
| 3 | { 1, 2, 3 } |
| 4 | { 3, 4, 5 } |
| 5 | { 3, 4, 5 } |

Tabela 4.5: Símbolos que podem ser gerados por cada estado, de acordo com a matriz $B$ após o treinamento.

utilizando a metologia apresentada (Equação (4.4)) quanto o resultado que seria obtido utilizando a metodologia tradicional [Rabiner, 1989].

A metodologia proposta reconheceu corretamente todas as seqüências que representam o comando *direita*, *para cima*, *esquerda* (apresentadas nas linhas 1, 3, 4, 5, 6, 7 e 9), considerando um limiar de 1e-10, exceto as que tinham menos de N elementos, como a seqüência apresentada na linha 10 da tabela. Esse pode ser considerado um problema da metodologia, uma vez que quando o número de observações é menor que o número de estados do modelo o último estado não pode ser atingido. Selecionando corretamente o número de estados do modelo esse problema pode ser eliminado.

Podemos observar na coluna $P(O|\lambda)$ que as únicas linhas que têm probabilidade zero são as que não podem ser geradas pelo modelo, ou seja, as que contêm alguma subseqüência de símbolos que não pode ser observada devido às probabilidades de ocorrência zero na matriz $B$. Já na coluna $P(O, q_{T-1} = N - 1|\lambda)$ todas as seqüências que não terminam no último gesto do comando possuem probabilidade zero, o que demonstra sua maior robustez e indica um menor índice de falsos positivos para a aplicação em questão. Pode ser facilmente observado que para qualquer seqüência de gestos diferente do comando treinado a probabilidade se aproximará de zero.

A probabilidade baixa de se observar o símbolo 3 no último estado ($b_5(3) = 0{,}01$) faz com que a probabilidade fique pelo menos 100 vezes menor para as seqüências terminadas em 3 quando se considera a metodologia proposta, como pode ser observado na Tabela 4.6. Essa é uma característica que não prejudica o reconhecimento, desde que o limiar seja escolhido adequadamente.

Resultados práticos da utilização da metodologia proposta são apresentados no próximo capítulo.

| $n$ | $O_n$ | $P(O, q_{T-1} = N-1 \mid \lambda)$ | $P(O \mid \lambda)$ |
|---|---|---|---|
| 1 | **0 0 0 0 2 2 2 2 4 4 4 4** | 8,7e-007 | 9,5e-007 |
| 2 | 1 1 1 1 1 1 1 1 1 1 1 1 | 0 | 0,2e-007 |
| 3 | **1 1 1 1 3 3 3 3 5 5 5 5** | 0,5e-007 | 3,0e-007 |
| 4 | **7 7 7 7 1 1 1 1 3 3 3 3** | 2,1e-010 | 3,2e-007 |
| 5 | **7 0 1 7 1 2 3 1 3 4 5 3** | 0,9e-010 | 5,4e-008 |
| 6 | **0 0 0 0 0 2 2 2 2 2 4 4 4 4 4** | 5,2e-008 | 5,3e-008 |
| 7 | **0 0 0 2 2 2 4 4 4** | 0,1e-005 | 1,5e-005 |
| 8 | 0 1 0 2 1 2 | 0 | 5,3e-004 |
| 9 | **0 1 2 3 4 5** | 5,4e-006 | 3,2e-005 |
| 10 | **0 1 2 3 4** | 0 | 9,7e-005 |
| 11 | 3 3 3 3 3 3 | 0 | 0 |
| 12 | 0 0 0 0 0 0 | 0 | 2,3e-003 |
| 13 | 1 1 1 1 1 1 | 0 | 5,2e-004 |
| 14 | 2 2 2 0 0 0 | 0 | 0 |
| 15 | 1 1 1 0 0 0 | 0 | 1,9e-004 |
| 16 | 5 6 6 6 | 0 | 0 |
| 17 | 7 7 7 7 7 7 | 0 | 9,7e-005 |
| 18 | 7 7 7 7 7 | 0 | 5,8e-004 |

Tabela 4.6: Seqüências de observações utilizadas na validação do MOM construído para reconhecimento do comando *direita*, *para cima*, *esquerda* e probabilidades de cada uma ser reconhecida utilizando tanto a metodologia proposta (penúltima coluna) quanto a tradicional (última coluna). As seqüências que representam esse comando, e portanto deveriam ser reconhecidas, são exibidas em negrito.

# Capítulo 5

# Resultados

Esse capítulo apresenta os resultados obtidos nos experimentos realizados, bem como a metodologia de teste e avaliação utilizada. Serão apresentados primeiramente o ambiente e alguns detalhes da implementação realizada, a seguir os conjuntos de treinamento e a metodologia de avaliação e então os resultados obtidos utilizando tanto cadeias de Markov quanto Modelos Ocultos de Markov (MOMs). Por último, será feita uma análise crítica dos resultados e apresentada uma tabela comparativa das duas metodologias.

## 5.1 Ambiente e Implementação

Todos os resultados apresentados foram obtidos a partir de imagens adquiridas de uma *webcam* genérica com resolução de 320 x 240 *pixels* a uma taxa de 20 quadros por segundo. Os testes foram executados em um microcomputador portátil com processador Pentium M de 1,6 GHz e 512MB de RAM com o sistema operacional Microsoft Windows XP (Service Pack 2).

Todo o sistema foi implementado na linguagem C++ seguindo os princípios da orientação para objetos de modo a obter o código mais intuitivo e reusável possível, sempre programando por intenção. No intuito de acelerar o desenvolvimento foram utilizadas bibliotecas disponíveis que já implementavam os algoritmos necessários sempre que possível.

Toda a implementação relativa aos MOMs baseou-se na biblioteca GHMM (*General Hidden Markov Model*) [Schliep et al., 2005], em sua versão 0.7.0a. GHMM é uma biblioteca multi-plataforma de código aberto na liguagem C que implementa de maneira eficiente as estruturas de dados e algoritmos para lidar com MOMs.

A aquisição e o processamento das imagens foram realizados utilizando a biblioteca OpenCV (*Open Source Computer Vision Library*) [OpenCV, 2004]. A OpenCV é uma biblioteca voltada para visão computacional em tempo real, ou seja, seu principal objetivo é desempenho. A OpenCV é baseada na Intel IPP (*Integrated Performance Primitives*) que é uma camada de baixo nível multi-plataforma que possibilita a realização de operações como o processamento de imagens utilizando rotinas de alto desempenho em processadores Intel. Todas as operações com matrizes (e.g., Seção 4.3.1) também foram realizadas utilizando a biblioteca OpenCV.

A utilização dessas bibliotecas, que constituem módulos reusáveis de código, visa tornar o código mais claro e mais fácil de manter, se beneficiando das melhorias nas bibliotecas, uma vez que as mesmas estão em constante evolução independentemente deste trabalho.

Visando manter todo o código orientado para objetos, foram criados *wrappers* para todas as estruturas de dados utilizadas dessas bibliotecas, uma vez que ambas são implementadas em C. As funções que operam sobre essas estruturas se tornaram métodos desses *wrappers*. Dessa maneira, eliminou-se a alocação e desalocação de memória manual, sendo as rotinas necessárias encapsuladas nas respectivas classes.

Devido ao alto acoplamento do algoritmo *forward* presente na biblioteca GHMM o mesmo foi implementado novamente para a fase de avaliação (Seção 4.4.3). As limitações se devem ao fato do algoritmo fazer validações para as demais operações presentes na biblioteca, o que não seria necessário na avaliação de cada seqüência de observações, uma vez que apenas o algoritmo *forward* é utilizado nessa etapa.

## 5.2 Avaliação

Os conjuntos de dados utilizados nos experimentos são apresentados na Tabela 5.1 com uma breve descrição de cada um deles. O conjunto TRVAL contém seqüências de gestos em que o executor está próximo da câmera e se mantém a uma distância constante, além dos gestos executados possuírem aproximadamente a mesma extensão. Já no conjunto NUMREP a extensão dos gestos varia consideravelmente, o executor dos gestos se move e o *background* é complexo, sendo portanto um conjunto de testes considerado difícil. Por sua vez, o conjunto TWDIR contém sequências de gestos diferentes realizadas por um usuário leigo que se mantém a uma distância constante da câmera. Um exemplo de seqüência de gestos é apresentado na Figura 5.1, onde pode ser vista a seqüencia *para cima*, *direita*, *para baixo*. Conforme pode ser visto na Figura 5.1, foi utilizado um objeto colorido para identificar a trajetória do movimento realizado pelo usuário.

(a) Quadro 38 (b) Quadro 46 (c) Quadro 51 (d) Quadro 55 (e) Quadro 62 (f) Quadro 71 (g) Quadro 73

Figura 5.1: Seqüência de gestos *para cima*, *direita*, *para baixo*. A trajetória percorrida pelo objeto rastreado é apresentada na cor branca.

| Código | Número de quadros | Descrição |
|---|---|---|
| NUMREP | 2104 | Repetições da seqüência de gestos *direita*, *para cima*, *esquerda*, *para baixo*. Contém 63 repetições dessa seqüência executadas de diferentes maneiras com grande variação no comprimento dos gestos e deslocamento do executor (tanto horizontal quanto em profundidade) em *background* complexo. |
| TRVAL | 1240 | Repetições da seqüência de gestos *direita*, *para cima*, *esquerda*, *para baixo*. Contém 31 execuções desse comando com um intervalo no meio no qual são executados diversos outros gestos. Este intervalo também contém gestos na diagonal. |
| TWDIR | 894 | Sequência contendo 19 repetições da seqüência de gestos *esquerda*, *para cima*, *direita*, *para baixo* e 25 repetições da seqüência *para cima*, *esquerda*, *para baixo*, *direita*. |

Tabela 5.1: Conjuntos de dados utilizados nos experimentos.

A avaliação de sistemas de reconhecimento é feita por meio da contagem de erros de deleção (D), inserção (I) e substituição (S). Essa metodologia de avaliação é baseada na modelagem da transmissão de informação por um canal ruidoso [Bahl e Jelinek, 1975] e foi utilizada, por exemplo, em sistemas de reconhecimento de fala [Ynoguti, 1999] e em sistemas de reconhecimento de gestos [Vogler e Metaxas, 1998]. O erro de deleção consiste na não identificação do comando nos quadros esperados. O erro de inserção consiste na identificação de um comando em quadros onde nenhum comando era esperado. O erro de substituição consiste na identificação de um comando nos quadros em que outro comando deveria ser reconhecido. Portanto, por definição, os erros de inserção são os *falsos positivos* e os erros de substituição representam erros de reconhecimento (classificação) do sistema.

Para calcular o desempenho geral do sistema será utilizado o critério *word accuracy*,

que é um critério largamente utilizado para avaliação de sistemas de reconhecimento. Uma explicação detalhada sobre esse critério pode ser encontrada em [Boros et al., 1996]. Esse critério calcula a taxa de reconhecimento (TR) atribuindo aos erros de deleção, inserção e substituição pesos equivalentes. A taxa de reconhecimento pode ser calculada da seguinte maneira:

$$TR = 1 - \frac{D + I + S}{N_{real}} \quad , \tag{5.1}$$

onde:
D é o número de erros de deleção;
I é o número de erros de inserção;
S é o número de erros de substituição;
$N_{real}$ é o número de comandos que deveriam ter sido identificados (*ground truth*).

Definindo H como o número de comandos identificados corretamente pelo sistema, podemos redefinir o número de comandos que deveriam ser identificados como o total de comandos identificados corretamente somado à quantidade de erros de deleção e substituição, ou seja, $N_{real} = H + D + S$. Assim, podemos reescrever a Equação (5.1) como

$$TR = \frac{H - I}{H + D + S} \quad , \tag{5.2}$$

que fornecerá o índice de desempenho a ser considerado neste trabalho. Podemos observar que, de acordo com o número de erros de inserção, a TR pode ser negativa, significando que houveram mais falsos positivos que acertos.

No reconhecimento contínuo a classificação dos erros identificados nessas três categorias exige que sejam definidas as fronteiras entre cada um desses conjuntos. Nesse tipo de reconhecimento a identificação dos erros de inserção e substituição é complexa pois o reconhecimento não tem um quadro exato para ocorrer e sim uma faixa de quadros. Dessa maneira, a identificação de outro comando que não o esperado na faixa de quadros na qual o reconhecimento seria considerado um acerto será considerado um erro de substituição. Analogamente, qualquer reconhecimento fora dessa faixa será considerado um erro de inserção e o não reconhecimento de gesto algum nessa faixa será considerado um erro de deleção. Na próxima seção serão apresentados os resultados obtidos utilizando cadeias de Markov e na seção subseqüente os resultados obtidos utilizando MOMs.

## 5.3 Cadeias de Markov

Essa seção apresenta os resultados dos experimentos utilizando cadeias de Markov. Resultados preliminares desse trabalho foram apresentados em [Resende et al., 2006]. A primeira etapa é a inferência bayesiana (ou estatística) da probabilidade de cada gesto estar sendo executado dado um intervalo. Para cada conjunto de treinamento esse procedimento foi realizado utilizando os 30% iniciais de cada conjunto e os 30% finais, sendo os demais 40% utilizados para validação. A Tabela 5.2 apresenta os resultados para o conjunto NUMREP, tabelas similares foram obtidas para os demais conjuntos. Os ângulos de deslocamento são classificados em 16 setores, apresentados na Figura 5.2. Esses valores são utilizados nos demais experimentos com cadeias de Markov para obter a matriz de probabilidades de transição $A$ e atualizar o estado do modelo (Seção 4.3.1). As probabilidades são obtidas por meio da contagem dos intervalos (setores do círculo) identificados quando cada gesto (direção) está sendo realizado e normalizando essas quantidades por intervalo, obtendo assim os valores utilizados, exemplificados na Tabela 5.2.

Para validar a metodologia, foi realizado um experimento considerando apenas o

| Setor | Probabilidade de cada direção | | | |
|---|---|---|---|---|
| | *para baixo* | *esquerda* | *direita* | *para cima* |
| 0 | 00,0% | 96,6% | 00,0% | 03,4% |
| 1 | 00,0% | 36,4% | 00,0% | 63,6% |
| 2 | 02,9% | 08,8% | 00,0% | 88,2% |
| 3 | 00,0% | 01,6% | 00,0% | 98,4% |
| 4 | 00,0% | 00,0% | 00,0% | 100,0% |
| 5 | 00,0% | 00,0% | 00,0% | 100,0% |
| 6 | 00,0% | 00,0% | 60,0% | 40,0% |
| 7 | 00,0% | 00,0% | 95,0% | 05,0% |
| 8 | 01,6% | 00,8% | 96,0% | 01,6% |
| 9 | 21,7% | 04,4% | 69,6% | 04,4% |
| 10 | 93,6% | 00,0% | 04,3% | 02,1% |
| 11 | 98,2% | 00,9% | 00,9% | 00,0% |
| 12 | 100,0% | 00,0% | 00,0% | 00,0% |
| 13 | 88,2% | 11,8% | 00,0% | 00,0% |
| 14 | 13,0% | 87,0% | 00,0% | 00,0% |
| 15 | 00,0% | 99,5% | 00,0% | 00,5% |

Tabela 5.2: Exemplo de probabilidades de cada gesto a partir do setor no qual o vetor deslocamento foi classificado. Probabilidades obtidas para o conjunto NUMREP por meio de inferência bayesiana.

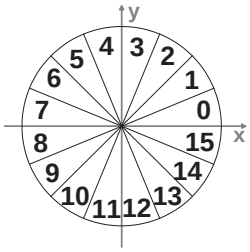


Figura 5.2: Intervalos nos quais os ângulos foram classificados para a inferência bayesiana. Por exemplo, os ângulos entre $\pi/8$ e $2\pi/8$ correspondem ao intervalo 1.

| Limiar | TRVAL | | | NUMREP | | | TWDIR | | |
|---|---|---|---|---|---|---|---|---|---|
| | H | D / I / S | TR | H | D / I / S | TR | H | D / I / S | TR |
| 0,10 | 10 | 4 / 5 / 0 | 0,357 | 7 | 17 / 19 / 0 | -0,500 | 9 | 2 / 4 / 0 | 0,455 |
| 0,15 | 10 | 4 / 5 / 0 | 0,357 | 8 | 16 / 16 / 0 | -0,333 | 8 | 3 / 2 / 0 | 0,545 |
| 0,20 | 10 | 4 / 5 / 0 | 0,357 | 22 | 2 / 1 / 0 | 0,875 | 7 | 4 / 2 / 0 | 0,455 |
| 0,25 | 10 | 4 / 5 / 0 | 0,357 | 21 | 3 / 1 / 0 | 0,833 | 6 | 5 / 1 / 0 | 0,455 |
| 0,30 | 10 | 4 / 5 / 0 | 0,357 | 21 | 3 / 1 / 0 | 0,833 | 6 | 5 / 1 / 0 | 0,455 |
| 0,35 | 10 | 4 / 5 / 0 | 0,357 | 21 | 3 / 1 / 0 | 0,833 | 6 | 5 / 1 / 0 | 0,455 |
| 0,40 | 14 | 0 / 0 / 0 | 1,000 | 20 | 4 / 1 / 0 | 0,792 | 5 | 6 / 1 / 0 | 0,364 |
| 0,45 | 14 | 0 / 0 / 0 | 1,000 | 18 | 6 / 0 / 0 | 0,750 | 3 | 8 / 1 / 0 | 0,182 |
| 0,50 | 14 | 0 / 0 / 0 | 1,000 | 18 | 6 / 0 / 0 | 0,750 | 3 | 8 / 1 / 0 | 0,182 |
| 0,55 | 14 | 0 / 0 / 0 | 1,000 | 18 | 6 / 0 / 0 | 0,750 | 4 | 7 / 0 / 0 | 0,364 |
| 0,60 | 14 | 0 / 0 / 0 | 1,000 | 18 | 6 / 0 / 0 | 0,750 | 4 | 7 / 0 / 0 | 0,364 |
| 0,65 | 13 | 1 / 0 / 0 | 0,929 | 18 | 6 / 0 / 0 | 0,750 | 4 | 7 / 0 / 0 | 0,364 |
| 0,70 | 13 | 1 / 0 / 0 | 0,929 | 18 | 6 / 0 / 0 | 0,750 | 4 | 7 / 0 / 0 | 0,364 |
| 0,75 | 13 | 1 / 0 / 0 | 0,929 | 18 | 6 / 0 / 0 | 0,750 | 4 | 7 / 0 / 0 | 0,364 |
| 0,80 | 10 | 4 / 0 / 0 | 0,714 | 12 | 12 / 0 / 0 | 0,500 | 4 | 7 / 0 / 0 | 0,364 |
| 0,85 | 10 | 4 / 0 / 0 | 0,714 | 12 | 12 / 0 / 0 | 0,500 | 4 | 7 / 0 / 0 | 0,364 |
| 0,90 | 10 | 4 / 0 / 0 | 0,714 | 12 | 12 / 0 / 0 | 0,500 | 4 | 7 / 0 / 0 | 0,364 |

Tabela 5.3: Taxas de reconhecimento das cadeias de Markov para cada um dos conjuntos de testes considerando diversos limiares para a seqüência de gestos *direita*, *para cima* e *esquerda*. Foram utilizados diversos limiares de reconhecimento para avaliar sua influência (Seção 4.3.2).

comando *direita*, *para cima*, *esquerda*. O resultado desse experimento é apresentado na Tabela 5.3, na qual podem ser observadas as taxas de reconhecimento (TR) obtidas para cada conjunto de acordo com o limiar utilizado. Também pode ser verificado que o melhor resultado médio foi para o limiar 0,4. Este limiar pode ser considerado adequado, uma vez que se aproxima de 0,5, que indicaria que nenhum outro estado é mais provável.

| Código | Seqüência de Gestos |
|---|---|
| BDC | *para baixo*, *direita*, *para cima* |
| EDB | *esquerda*, *direita*, *para baixo* |
| DBE | *direita*, *para baixo*, *esquerda* |
| DCE | *direita*, *para cima*, *esquerda* |
| CBE | *para cima*, *para baixo*, *esquerda* |
| CBD | *para cima*, *para baixo*, *direita* |
| CDB | *para cima*, *direita*, *para baixo* |
| CDE | *para cima*, *direita*, *esquerda* |

Tabela 5.4: Comandos utilizados na validação do reconhecimento contínuo. O código se refere às iniciais dos gestos.

A Tabela 5.4 apresenta os comandos utilizados nos experimentos considerando múltiplos comandos. Os resultados do reconhecimento contínuo são apresentados na Tabela 5.5, onde para cada conjunto de comandos são apresentados a quantidade de acertos, bem como a quantidade de erros de deleção, inserção e substituição e a taxa de reconhecimento para cada um dos três conjuntos de treinamento. Nesses experimentos foi utilizado o limiar de 0,40, selecionado por ter apresentado, na média, o melhor resultado para os três conjuntos.

| n | Comandos considerados | TRVAL | | | | | NUMREP | | | | | TWDIR | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | H | D | I | S | TR | H | D | I | S | TR | H | D | I | S | TR |
| 1 | DCE | 14 | 0 | 0 | 0 | 1,00 | 20 | 4 | 1 | 0 | 0,79 | 5 | 6 | 1 | 0 | 0,36 |
| 2 | DCE,ECD | 14 | 0 | 0 | 0 | 1,00 | 20 | 4 | 1 | 0 | 0,79 | 8 | 8 | 2 | 0 | 0,38 |
| 3 | BDC,DCE | 11 | 3 | 3 | 0 | 0,57 | 16 | 9 | 12 | 1 | 0,15 | 7 | 4 | 1 | 0 | 0,55 |
| 4 | BDC,BEC,DCE,ECD,EDB | 11 | 3 | 3 | 0 | 0,57 | 16 | 9 | 13 | 1 | 0,12 | 10 | 6 | 2 | 1 | 0,47 |
| 5 | BDC,BEC,DCE,ECD,EDB,CBE | 11 | 3 | 3 | 0 | 0,57 | 16 | 9 | 14 | 1 | 0,08 | 10 | 6 | 2 | 1 | 0,47 |
| 6 | BDC,BEC,DCE,ECD,EDB,CBE,CBD | 11 | 3 | 3 | 0 | 0,57 | 15 | 10 | 16 | 1 | -0,04 | 10 | 7 | 6 | 1 | 0,22 |
| 7 | BDC,BEC,DCE,ECD,EDB,CBE,CBD, CDB,CDE,CEB,DBE,EBD | 6 | 14 | 21 | 2 | -0,68 | 18 | 14 | 30 | 2 | -0,35 | 12 | 10 | 16 | 1 | -0,17 |
| 8 | CDBE,DCEB | 8 | 6 | 6 | 0 | 0,14 | 13 | 11 | 3 | 0 | 0,42 | 7 | 8 | 0 | 0 | 0,47 |
| 9 | CDBE,DCEB,ECDB,BDCE | 13 | 1 | 1 | 0 | 0,86 | 8 | 16 | 9 | 0 | -0,04 | 6 | 10 | 3 | 0 | 0,19 |

Tabela 5.5: Resultados do reconhecimento para múltiplos comandos utilizando cadeias de Markov. Para cada conjunto de comandos utilizado são apresentados os resultados para cada um dos conjuntos de treinamento.

## 5.4 MOMs

Essa seção apresenta os resultados dos experimentos utilizando MOMs, introduzindo antes algumas características importantes que nortearam as decisões tomadas. Tanto o reconhecimento isolado quanto o contínuo são realizados. O reconhecimento isolado consiste na apresentação de seqüências com início e fim determinado ao modelo para serem avaliadas, enquanto no reconhecimento contínuo o modelo está associado a um fluxo (*stream*) de informações onde o início e o fim de cada comando não são indicados.

O reconhecimento isolado não possui aplicação prática no contexto desse trabalho, uma vez que requer que o início do comando seja indicado. Todavia o reconhecimento isolado é importante para mensurar algumas características do modelo que podem ser mais facilmente avaliadas no modo isolado que no contínuo.

A próxima subseção demonstra como a quantidade de seqüências utilizadas para treinamento afeta o resultado e a subseção seguinte apresenta os resultados do reconhecimento contínuo com múltiplos comandos.

### 5.4.1 Treinamento

Visando identificar a quantidade de dados que o modelo considerado necessita para treinamento foi realizado um experimento envolvendo a repetição exaustiva de um mesmo comando de modo a avaliar a influência do tamanho do conjunto de treinamento nos resultados. Para cada tamanho do conjunto de treinamento, começando em uma execução do comando, o procedimento de treinamento e validação foi executado 20 e 30 vezes, escolhendo aleatoriamente os elementos desse conjunto. O número de repetições do procedimento também foi variado para verificar se esse parâmetro estava influindo nos resultados obtidos. Os resultados desse experimento são apresentados na Tabela 5.6. Podemos verificar que a partir de um certo tamanho do conjunto de treinamento, aproximadamente 17 execuções do comando, as variações na taxa de reconhecimento são bem pequenas, podendo algumas distorções nos valores serem atribuídas à redução do tamanho do conjunto de validação à medida que aumenta o tamanho do conjunto de treinamento.

O resultado desse experimento não pode ser tomado como verdade absoluta uma vez que foi utilizado apenas um conjunto de dados, entretanto, ele pode ser utilizado como referência para os demais experimentos, indicando a quantidade mínima de dados que deve ser utilizada para treinamento de modo a obter um resultado satisfatório com o modelo utilizado.

| Tamanho Conjunto Treinamento | Tamanho Conjunto Validação | 20 Execuções | | 30 Execuções | |
|---|---|---|---|---|---|
| | | TR Médio | Aumento TR Médio (%) | TR Médio | Aumento TR Médio (%) |
| 1 | 62 | 0,014 | | 0,037 | |
| 2 | 61 | 0,172 | 1155,5% | 0,252 | 590,5% |
| 3 | 60 | 0,340 | 97,5% | 0,342 | 35,6% |
| 4 | 59 | 0,493 | 45,1% | 0,550 | 60,6% |
| 5 | 58 | 0,544 | 10,3% | 0,503 | -8,4% |
| 6 | 57 | 0,600 | 10,3% | 0,584 | 16,0% |
| 7 | 56 | 0,632 | 5,4% | 0,658 | 12,6% |
| 8 | 55 | 0,728 | 15,2% | 0,693 | 5,4% |
| 9 | 54 | 0,733 | 0,7% | 0,729 | 5,1% |
| 10 | 53 | 0,746 | 1,8% | 0,745 | 2,2% |
| 11 | 52 | 0,772 | 3,5% | 0,799 | 7,2% |
| 12 | 51 | 0,803 | 4,0% | 0,758 | -5,2% |
| 13 | 50 | 0,869 | 8,2% | 0,807 | 6,5% |
| 14 | 49 | 0,748 | -13,9% | 0,819 | 1,5% |
| 15 | 48 | 0,827 | 10,6% | 0,837 | 2,2% |
| 16 | 47 | 0,855 | 3,4% | 0,834 | -0,3% |
| 17 | 46 | 0,851 | -0,5% | 0,843 | 1,1% |
| 18 | 45 | 0,847 | -0,5% | 0,850 | 0,8% |
| 19 | 44 | 0,858 | 1,3% | 0,848 | -0,3% |
| 20 | 43 | N/D | | 0,886 | 4,5% |
| 21 | 42 | N/D | | 0,881 | -0,6% |
| 22 | 41 | N/D | | 0,884 | 0,3% |
| 23 | 40 | N/D | | 0,882 | -0,2% |
| 24 | 39 | N/D | | 0,882 | 0,0% |
| 25 | 38 | N/D | | 0,879 | -0,4% |
| 26 | 37 | N/D | | 0,883 | 0,4% |
| 27 | 36 | N/D | | 0,908 | 2,9% |
| 28 | 35 | N/D | | 0,889 | -2,2% |

Tabela 5.6: Taxa de reconhecimento de acordo com a quantidade de elementos utilizados para treinamento utilizando o conjunto NUMREP. Para cada tamanho do conjunto de treinamento são apresentados o tamanho do conjunto de validação, a taxa de reconhecimento TR (Equação (5.2)) e o aumento percentual relativo à linha anterior. O procedimento de treinamento e validação foi executado 20 e 30 vezes para cada tamanho de conjunto de treinamento, sendo que as seqüências utilizadas para treinamento foram escolhidas aleatoriamente sendo as demais utilizadas para validação.

### 5.4.2 Reconhecimento contínuo

Para o reconhecimento contínuo o treinamento foi realizado utilizando os 30% iniciais de cada conjunto e os 30% finais, sendo os demais 40% utilizados para validação. A Tabela 5.7 apresenta os resultados do reconhecimento contínuo considerando diversos conjuntos de comando para os três conjuntos de dados utilizados, são apresentados a quantidade de acertos, bem como as quantidades de erros de deleção, inserção e substituição e a taxa de reconhecimento. Os quadros foram processados seqüencialmente, como descrito na Seção 4.4.3. A Tabela 5.7 apresenta os resultados para o reconhecimento de um único comando, para o reconhecimento de conjuntos simples de comandos e para conjuntos contendo comandos cujos gestos de início são também gestos finais de outros comandos, onde podem ser observados os piores resultados, uma vez que essa interseção é crítica para a metologia proposta.

A metodologia de avaliação contínua utilizada busca maximizar de certa maneira a identificação das seqüências com a utilização do *buffer* (Seção 4.4.3). Pode-se considerar que esse *buffer* torna o processo de avaliação um pouco menos severo, uma vez que permite que as sub-seqüências dentro de um comando possam ser identificadas, considerando que o início do comando pode estar em qualquer um dos quadros. Dessa maneira é tratado o principal problema do reconhecimento contínuo, que é o desconhecimento do início e fim de cada comando.

A metodologia utilizada para treinamento permite que sejam criados modelos para reconhecer quaisquer comandos formados pelos gestos do conjunto de treinamento, mesmo que os comandos não tenham sido executados por completo nenhuma vez. Essa gama de possibilidades foi explorada na criação de diversos modelos para reconhecer comandos que não foram executados em nenhum dos conjuntos de dados. Entretanto, essa possibilidade deve ser utilizada com critério, uma vez que se o gesto estiver inserido sempre em uma mesma seqüência no conjunto de treinamento pode ser considerado pelo modelo que o intervalo de transição para o próximo gesto faz parte obrigatoriamente do gesto, sendo este trecho, em geral, na direção diagonal entre os dois gestos.

| n | Comandos considerados | TRVAL | | | | | NUMREP | | | | | TWDIR | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | H | D | I | S | TR | H | D | I | S | TR | H | D | I | S | TR |
| 1 | DCE | 15 | 0 | 0 | 0 | 1,00 | 24 | 0 | 0 | 0 | 1,00 | 11 | 0 | 0 | 0 | 1,00 |
| 2 | BDC,BEC,DCE,ECD,EDB | 15 | 0 | 0 | 0 | 1,00 | 22 | 3 | 2 | 0 | 0,80 | 16 | 0 | 0 | 0 | 1,00 |
| 3 | BDC,BEC,DCE,ECD,EDB,CBE | 15 | 0 | 0 | 0 | 1,00 | 22 | 3 | 3 | 0 | 0,76 | 16 | 0 | 0 | 0 | 1,00 |
| 4 | BDC,BEC,DCE,ECD,EDB,CBE,CBD | 15 | 0 | 0 | 0 | 1,00 | 15 | 10 | 16 | 0 | -0,04 | 16 | 0 | 0 | 0 | 1,00 |
| 5 | BDC,BEC,DCE,ECD,EDB,CBE,CBD, CDB,CDE,CEB,DBE,EBD | 9 | 10 | 17 | 0 | -0,42 | 16 | 16 | 26 | 1 | -0,30 | 12 | 10 | 17 | 0 | -0,23 |
| 6 | CDBE,DCEB | 14 | 0 | 0 | 0 | 1,00 | 24 | 0 | 0 | 0 | 1,00 | 15 | 0 | 0 | 0 | 1,00 |
| 7 | CDBE,DCEB,ECDB,BDCE | 1 | 13 | 14 | 0 | -0,93 | 14 | 10 | 10 | 0 | 0,17 | 12 | 4 | 5 | 0 | 0,44 |

Tabela 5.7: Resultados do reconhecimento para múltiplos comandos utilizando MOMs. Para cada conjunto de comandos utilizado são apresentados os resultados para cada um dos conjuntos de treinamento.

## 5.5 Análise crítica dos resultados

Considerando o reconhecimento de apenas um comando os MOMs foram superiores às cadeias de Markov, uma vez que foram perfeitos nesse caso versus uma taxa de reconhecimento média de 0,72 para as cadeias de Markov.

No reconhecimento de múltiplos comandos por sua vez, podemos observar alguns casos em que os MOMs são inferiores. O resultado se degrada significamente quando o gesto final de um comando também é início de outro comando, pois nesse caso pode ser que esse gesto seja identificado somente como final ou como final e inicial. Devido à grande quantidade de comandos e ao fato de não haver intervalos entre a execução de cada comando, os erros de inserção que porventura ocorram acabam resultando em um ou mais erros de deleção e em novos erros de inserção posteriormente. Esses erros em cadeia ocorrem porque assim que cada gesto é identificado o *buffer* é esvaziado, impedindo que a sequência correta de gestos seja retomada, e isso implica que um gesto identificado erroneamente (inserção) irá esvaziar o *buffer* e fará com que o próximo comando não seja reconhecido, resultando em erros em cadeia até o fim da seqüência. Isso pode ser observado principalmente nas linhas 5 e 7 da Tabela 5.7. Caso houvesse uma pausa entre cada execução do comando esse fenômeno não seria observado.

Nos resultados para os MOMs também pôde ser observada a influência do tamanho do conjunto de treinamento (Tabela 5.6). À medida que mais dados são utilizados para treinamento a capacidade de generalização do modelo é aumentada reduzindo a quantidade de erros de inserção. Também pode ser observado que a tendência é a de que o modelo se estabilize a partir de determinado tamanho desse conjunto.

A taxa de reconhecimento cai à medida que o número de comandos aumenta, tanto utilizando cadeias de Markov quanto utilizando MOMs. No caso de cadeias de Markov a queda é muito mais significativa por se tratar de um sistema mais vulnerável aos erros de inserção (falsos positivos), que pode ser destrutivo para o reconhecimento contínuo em ambas as metologias. Como em uma identificação, correta ou não, o sistema impede que seja feita nova identificação em seguida, limpando o *buffer* no caso dos MOMs ou reinicializando as probabilidade no caso das cadeias de Markov, os erros se tornam atrelados de certa maneira caso comandos estejam sendo executados em seqüência, pois um falso positivo pode prejudicar o reconhecimento de vários comandos posteriores.

Podemos observar no reconhecimento contínuo considerando múltiplos comandos que à medida que a quantidade de comandos aumenta a taxa de reconhecimento cai e que essa queda depende da interseção do comando adicionado com os demais existentes. Isso mostra que a escolha dos comandos tem vital importância na obtenção de um bom

resultado. A escolha de comandos que não possuam nenhuma interseção (ortogonais) pode levar a uma taxa de reconhecimento próxima da taxa para comandos individuais.

Devido ao produto cartesiano realizado com os dados de treinamento (Seção 4.4.5) a quantidade de registros utilizados para treinamento aumenta à ordem do número de gestos à medida que o conjunto de treinamento cresce (e.g., ao cubo para um comando de três gestos), tornando inviável o treinamento para grandes conjuntos de dados. O treinamento de um modelo para o reconhecimento de um comando de quatro gestos utilizando um conjunto de 30 execuções do comando leva mais de sete horas no computador utilizado para a realização dos experimentos, já para um comando de três gestos leva em torno de quatro minutos.

Os melhores resultados com MOMs podem se dever ao fato destes exigirem um comprimento mínimo para cada gesto e para os comandos como um todo, tornando-os mais robustos a ruídos, o que não ocorre com as cadeias de Markov. Podemos citar como exemplo o problema de se reconhecer um comando no final do penúltimo gesto, pois, no período de transição, quando um usuário faz um gesto diagonal pode parecer que ele já esteja fazendo o último gesto. Ocorrências como essa foram contabilizadas como erros de inserção e prejudicaram significamente os resultados com cadeias de Markov, pois erros de inserção podem causar diversos erros de identificação em seqüência quando os gestos são executados sem pausa, uma vez que o sistema reconhecedor é reiniciado.

A Tabela 5.8 compara as características das duas metodologias. Essa tabela visa apresentar de forma clara as vantagens e desvantagens de cada uma delas.

| Item | Cadeias de Markov | MOMs |
|---|---|---|
| Complexidade | Baixa | Alta |
| Generaliza para qualquer comando | Sim | Sim |
| Exige duração mínima de cada gesto | Não | Sim |
| Complexidade do Treinamento | Baixa | Alta |
| Sensibilidade ao limiar de reconhecimento | Alta | Baixa |
| Taxa de reconhecimento média para um comando | 0,72 | 1,00 |
| Tempo de execução médio por quadro por comando (apenas Sistema a Eventos Discretos) | 0,04 ms | 0,8 ms |

Tabela 5.8: Comparação das duas metologias utilizadas, baseadas em cadeias de Markov e MOMs.

# Capítulo 6

# Conclusões e Trabalhos Futuros

Este trabalho apresentou uma metodologia para construção de uma interface visual humano-robô baseada em uma linguagem composta por pequenos gestos, mais fáceis de serem identificados e modelados. Dois módulos desacoplados compõem o sistema, o de visão computacional e o de processamento estocástico dos gestos. Foram utilizados dois tipos de Sistemas a Eventos Discretos (SEDs) para o processamento estocástico dos gestos: cadeias de Markov e MOMs, sendo os resultados dos dois comparados. Apesar de ter sido validada apenas com gestos simples e comandos compostos por um pequeno número de gestos a metodologia é genérica e suporta tanto gestos quanto comandos mais complexos.

As principais características do sistema são o reconhecimento contínuo dos comandos sem a necessidade da segmentação dos gestos, sendo o início e o fim de cada comando identificados implicitamente pelo sistema, e a construção do modelo de maneira automática a partir da especificação dos comandos e dos dados de treinamento, desde que estes estejam segmentados. O número de estados do modelo, que geralmente é definido de maneira experimental, neste trabalho é definido a partir dos dados de treinamento, utilizando a Equação (4.3).

Em comparação com trabalhos publicados, a abordagem proposta difere da de Siskind e Morris [Siskind e Morris, 1996] por propor o reconhecimento contínuo on-line, sem a necessidade da indicação de início e fim de cada evento, além de propor um método de rastreamento que privilegia a robustez. Difere da de Vogler e Metaxas [Vogler e Metaxas, 1998] por utilizar um sistema de visão simples e de baixo custo computacional, além de não necessitar da informação da posição 3D, cuja determinação é complexa. Além disso, não exige segmentação para identificação do início e fim de cada gesto, sendo esta realizada implicitamente pelo modelo.

Os experimentos realizados comprovaram a robustez do método e a baixa incidência de falsos positivos, principalmente para os MOMs. A utilização de MOMs no contexto proposto apresentou desempenho superior ao das cadeias de Markov. Vale ressaltar também que os resultados foram obtidos utilizando uma porção relativamente pequena dos dados para treinamento, uma vez que foram utilizados aproximadamente 60% dos dados para treinamento, enquanto trabalhos similares publicados utilizam em torno de 80% dos dados [Vogler e Metaxas, 1998, Yang e Ahuja, 1999]. Também foi identificada a importância da escolha correta dos comandos da gramática, para evitar uma deterioração significativa da taxa de reconhecimento à medida que a quantidade de comandos aumenta.

Foram obtidas taxas de reconhecimento de até 1,00 no reconhecimento contínuo de múltiplos comandos utilizando MOMs. Comparando com trabalhos existentes a metodologia apresentada pode ser considerada uma alternativa interessante. Vogler e Metaxas [Vogler e Metaxas, 1998] obtiveram taxa de reconhecimento de até 0,8771 no reconhecimento contínuo da ASL (*American Sign Language*) utilizando um sistema com três câmeras que fornece a posição 3D e MOMs. Hong [Hong et al., 2000a] obteve taxas de reconhecimento em torno de 0,9 para gestos como acenar com a mão esquerda, desenhar um círculo e desenhar um oito utilizando máquinas de estados finitos construídas a partir de dados de treinamento, um pequeno conjunto de dados foi utilizado para validação.

A complexidade computacional é baixa devido ao armazenamento do contexto nos sistemas a eventos discretos. No caso das cadeias de Markov apenas os novos dados são processados a cada quadro. Isso é possível devido à propriedade das cadeias de Markov serem "sem memória" (*memoryless*) e garante bom desempenho ao sistema, pois apenas os dados do quadro atual são considerados na atualização do estado. No caso dos MOMs a avaliação do sistema também é eficiente, uma vez que sua complexidade é $O(N^2T)$ e para os modelos utilizados temos $7 < N < 16$ e $20 < T < 40$.

Neste trabalho foram utilizados poucos conjuntos de dados para teste, não sendo nenhum dos conjuntos muito grande devido à dificuldade de geração de dados para teste e à necessidade de segmentação manual dos mesmos para fornecer os dados para treinamento e avaliação do desempenho do sistema. Também deve ser considerado que o alto custo computacional do treinamento dos MOMs pode inviabilizar a utilização do método baseado em MOMs para conjuntos de treinamento ou comandos muito grandes.

Como produção científica desse trabalho, há um artigo sendo preparado para ser enviado para um periódico indexado e os resultados apresentados na Seção 5.3 foram publicados no artigo:

Resende, R. M.; Pereira, G. A. S.; Maia, C. A. e Carceroni, R. L. (2006). A gestural language recognition methodology for human-robot interaction. In *Proceedings of the International Conference on Robotics and Automation*, pp. 4333–4335.

Para continuidade desse trabalho podemos citar as seguintes iniciativas, sendo algumas simples e outras mais complexas de serem desenvolvidas:

- Como foi identificado que a escolha dos comandos afeta diretamente o resultado, um trabalho futuro seria o estudo dessa gramática, buscando a ortogonalidade entre os gestos.
- Utilização de métodos de rastreamento mais elaborados, como por exemplo o Condensation [Isard e Blake, 1998], que possibilitem que os gestos sejam realizados com a própria mão do usuário, sem a necessidade de outro objeto.
- Verificação do desempenho dos MOMs com diferentes critérios para definição do número de estados, podendo ser obtida uma melhora em relação aos resultados apresentados.
- Modificação do procedimento de treinamento de modo a considerar a maximização da probabilidade que é avaliada, ou seja, maximizar $P(O, q_{T-1} = N - 1|\lambda)$ e não $P(O|\lambda)$.
- De acordo com Cassandras [Cassandras e Lafortune, 1999] pode-se trabalhar com SEDs em diferentes níveis de abstração: linguagens, linguagens temporizadas e linguagens temporizadas estocásticas. Esse trabalho utilizou a primeira delas, mais simples, não temporizada. A utilização de uma linguagem temporizada estocástica, na qual a duração de cada evento é modelada por uma função de distribuição de probabilidades, poderia possibilitar melhores resultados por meio da modelagem adequada da duração de cada gesto, por exemplo.
- Alteração do critério de avaliação dos MOMs somando a probabilidade de todos os estados que representam o último gesto, ou seja, que faziam parte do MOM individual do último gesto, em vez de considerar apenas o último estado (Equação (4.4)). Essa abordagem pode melhorar os resultados ao considerar gestos com extensão menor e amplificar as probabilidades de gestos contínuos na mesma direção.
- Avaliação da atribuição de valores diferentes de zero a todos os elementos das matrizes A e B de modo a permitir maior flexibilidade no treinamento dos MOMs individuais concatenados, após o treinamento independente de cada gesto. Da maneira

atual as transições e símbolos que não são permitidas nos MOMs individuais não poderão se tornar válidas no MOM composto.

- Utilização de MOMs contínuos no lugar dos discretos, compostos por misturas de gaussianas. Esse tipo de MOM não exige a discretização das informações, que pode resultar na degradação da informação.

- Modificação do procedimento de reconhecimento para permitir erros e pausas na execução da seqüência de gestos.

- Por último, para facilitar o treinamento e permitir o uso prático do sistema, é importante o desenvolvimento de uma interface para segmentação automática de gestos. Essa interface pode fornecer os dados para o treinamento, tornando desnecessária a segmentação manual dos dados.

# Referências Bibliográficas


[Aloimonos, 1990] Aloimonos, J. (1990). Purposive and qualitative active vision. In *Proceedings of the International Conference on Pattern Recognition*, volume 1, pp. 346–360.

[Bahl e Jelinek, 1975] Bahl, L. R. e Jelinek, F. (1975). Decoding for channels with insertions, deletions, and substitutions with applications to speech recognition. *IEEE Transactions on Information Theory*, 21(4):404–411.

[Baker, 1975] Baker, J. K. (1975). The dragon system – an overview. *IEEE Transactions on Acoustic, Speech and Signal Processing*, 23(1):24–29.

[Baldi e Brunak, 1998] Baldi, P. e Brunak, S. (1998). *Bioinformatics: The Machine Learning Approach.* MIT Press.

[Ballard e Brown, 1982] Ballard, D. H. e Brown, C. M. (1982). *Computer Vision.* Prentice Hall.

[Baum, 1972] Baum, L. E. (1972). An inequality and associated maximization technique in the statistical estimation for probabilistic functions of Markov processes. *Inequalities*, 3:1–8.

[Baum e Egon, 1967] Baum, L. E. e Egon, J. A. (1967). An inequality with applications to statistical estimation for probabilistic functions of a Markov process and to a model for ecology. *Bull. Amer. Meteorol. Soc.*, 73:360–363.

[Baum e Petrie, 1966] Baum, L. E. e Petrie, T. (1966). Statistical inference for probabilistic functions of finite state Markov chains. *Annals of Mathematical Statistics*, 37:1554–1563.

[Baum et al., 1970] Baum, L. E.; Petrie, T.; Soules, G. e Weiss, N. (1970). A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *Annals of Mathematical Statistics*, 41(1):164–171.

[Baum e Sell, 1968] Baum, L. E. e Sell, G. R. (1968). Growth functions for transformations on manifolds. *Pacific Journal of Mathematics*, 27(2):211–227.

[Berger, 1993] Berger, J. O. (1993). *Statistical Decision Theory and Bayesian Analysis*. Springer, 2a edição.

[Bishop, 2006] Bishop, T. (2006). Microsoft developing robotics software. URL http://www.technewsworld.com/story/KckBhJJDyjKtwi/Microsoft-Developing-Robotics-Software.xhtml. Acessado em 21 de junho de 2006.

[Bobick e Wilson, 1997] Bobick, A. F. e Wilson, A. D. (1997). A state-based approach to the representation and recognition of gesture. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(12):1325–1337.

[Boros et al., 1996] Boros, M.; Eckert, W.; Gallwitz, F.; Görz, G.; Hanrieder, G. e Niemann, H. (1996). Towards understanding spontaneous speech: Word accuracy vs. concept accuracy. In *Proceedings of the International Conference on Spoken Language Processing*, volume 2, pp. 1009–1012.

[Cassandras e Lafortune, 1999] Cassandras, C. e Lafortune, S. (1999). *Introduction to Discrete Event Systems*. The Kluwer International Series in Discrete Event Dynamic Systems. Kluwer Academic Publishers.

[Cipolla et al., 1993] Cipolla, R.; Okamoto, Y. e Kuno, Y. (1993). Robust structure from motion using motion parallax. In *Proceedings of the International Conference on Computer Vision*, pp. 374–382.

[Cui e Weng, 1999] Cui, Y. e Weng, J. (1999). A learning-based prediction and verification segmentation scheme for hand sign image sequence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 798–804.

[Davis e Shah, 1994a] Davis, J. e Shah, M. (1994a). Determining 3-D hand motion. In *Proceedings of the IEEE Asilomar Conference on Signals, Systems and Computers*, pp. 1262–1266.

[Davis e Shah, 1994b] Davis, J. e Shah, M. (1994b). Recognizing hand gestures. In Eklundh, J.-O., editor, *Proceedings of the European Conference on Computer Vision*, volume 1, pp. 331–340. Springer-Verlag.

[Davis e Shah, 1994c] Davis, J. e Shah, M. (1994c). Visual gesture recognition. *Vision, Image and Signal Processing*, 141(2):101–106.

[Durbin et al., 1998] Durbin, R.; Eddy, S. R.; Krogh, A. e Mitchison, G. (1998). *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press.

[Eisner, 2002] Eisner, J. (2002). An interactive spreadsheet for teaching the forward-backward algorithm. In Radev, D. e Brew, C., editores, *Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching NLP and CL*, pp. 10–18.

[Fairchild, 2005] Fairchild, M. D. (2005). *Color Appearance Models*. John Wiley & Sons, 2a edição.

[Freeman et al., 1996] Freeman, W. T.; Tanaka, K.; Ohta, J. e Kyuma, K. (1996). Computer vision for computer games. In *Proceedings of the International Conference on Automatic Face and Gesture Recognition*, pp. 100–105.

[Fukumoto et al., 1994] Fukumoto, M.; Suenaga, Y. e Mase, K. (1994). Finger-pointer: pointing interface by image processing. *Computers and Graphics*, 18(5):633–642.

[Hastie e Stuetzle, 1989] Hastie, T. e Stuetzle, W. (1989). Principal curves. *Journal of American Statistical Association*, 84(406):502–516.

[Hogg, 1983] Hogg, D. (1983). Model-based vision: a program to see a walking person. *Image and vision computing*, 1(1):5–20.

[Hong et al., 2000a] Hong, P.; Turk, M. e Huang, T. S. (2000a). Constructing finite state machines for fast gesture recognition. In *Proceedings of the International Conference on Pattern Recognition*, pp. 3695–3698.

[Hong et al., 2000b] Hong, P.; Turk, M. e Huang, T. S. (2000b). Gesture modeling and recognition using finite state machines. In *Proceedings of the International Conference on Face and Gesture Recognition*, pp. 410–415. IEEE Computer Society.

[Horn, 1986] Horn, B. K. P. (1986). *Robot Vision*. MIT electrical engineering and computer science series. MIT Press.

[Horowitz e Pentland, 1991] Horowitz, B. e Pentland, A. (1991). Recovery of non-rigid motion and structure. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 325–330.

[Isard e Blake, 1998] Isard, M. e Blake, A. (1998). Condensation – conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28.

[Kölsch et al., 2004] Kölsch, M.; Turk, M. e Höllerer, T. (2004). Vision-based interfaces for mobility. In *Proceedings of the International Conference on Mobile and Ubiquitous Systems: Networking and Services*, pp. 86–94. IEEE Computer Society.

[Kulp et al., 1996] Kulp, D.; Haussler, D.; Reese, M. G. e Eeckman, F. H. (1996). A generalized hidden Markov model for the recognition of human genes in DNA. In *Proceedings of the Fourth International Conference on Intelligent Systems for Molecular Biology*, pp. 134–142. AAAI Press.

[Murphy e Casper, 2002] Murphy, R. e Casper, J. (2002). Human-robot interactions in robot-assisted urban search and rescue. In *Proceedings from the 2002 NRL workshop on multi-robot systems*, p. 221.

[Nilubol et al., 1998] Nilubol, C.; Pham, Q. H.; Mersereau, R. M.; Smith, M. J. T. e Clements, M. A. (1998). Translational and rotational invariant hidden Markov models for automatic target recognition. *Signal Processing, Sensor Fusion, and Target Recognition VII*, 3374(1):179–185. Proceedings of SPIE.

[Ong e Bowden, 2004] Ong, E. J. e Bowden, R. (2004). A boosted classifier tree for hand shape detection. In *Proceedings of the International Conference on Automatic Face and Gesture Recognition*, pp. 889–894.

[OpenCV, 2004] OpenCV (2004). Open source computer vision library (beta 4). Available at http://sourceforge.net/projects/opencvlibrary.

[O'Rourke e Badler, 1980] O'Rourke, J. e Badler, N. (1980). Model-based image analysis of human motion using constraint propagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(6):522–536.

[Pedersen et al., 1996] Pedersen, A. G.; Baldi, P.; Brunak, S. e Chauvan, Y. (1996). Characterization of prokaryotic and eukaryotic promoters using hidden markov models. In *Proceedings of the Fourth International Conference on Intelligent Systems for Molecular Biology*, pp. 182–191. AAAI Press.

[Pransky, 1996] Pransky, J. (1996). Service robots - how should we define them? *Service Robot: An International Journal*, 2(1):4–5.

[Quek et al., 1995] Quek, F. K. H.; Mysliwiec, T. e Zhao, M. (1995). FingerMouse: A freehand pointing interface. In *Proceedings of the International Workshop on Automatic Face and Gesture Recognition*, pp. 372–377.

[Rabiner, 1989] Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

[Rabiner e Juang, 1986] Rabiner, L. R. e Juang, B. H. (1986). An introduction to hidden Markov models. *IEEE ASSP Magazine*, 3(1):4–16.

[Rangarajan e Shah, 1991] Rangarajan, K. e Shah, M. (1991). Establishing motion correspondence. In *CVGIP: Image Understanding*, volume 54, pp. 56–73.

[Resende et al., 2006] Resende, R. M.; Pereira, G. A. S.; Maia, C. A. e Carceroni, R. L. (2006). A gestural language recognition methodology for human-robot interaction. In *Proceedings of the International Conference on Robotics and Automation*, pp. 4333–4335.

[Rosales et al., 2001] Rosales, R.; Athitsos, V.; Sigal, L. e Sclaroff, S. (2001). 3D hand pose reconstruction using specialized mappings. In *Proceedings of the International Conference on Computer Vision*, volume 1, pp. 378–385.

[Russell e Norvig, 2002] Russell, S. J. e Norvig, P. (2002). *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2a edição.

[Schliep et al., 2005] Schliep, A.; Georgi, B.; Wasinee; Rungsarityotin; Costa, I. G.; Grunau, J. e Heinig, M. (2005). GHMM library 0.7.0a. http://sourceforge.net/projects/ghmm.

[Scholtz, 2002] Scholtz, J. C. (2002). Human-robot interactions: creating synergistisc cyber forces. In *Proceedings from the 2002 NRL workshop on multi-robot systems*, pp. 177–184.

[Schrodt, 2000] Schrodt, P. A. (2000). Pattern recognition of international crises using hidden Markov models. In Richards, D. E.-A., editor, *Political Complexity: Nonlinear models of politics*, pp. 296–328. University of Michigan Press, Ann Arbor, MI, USA. Presented at the Annual Meeting of the International Studies Association, Toronto.

[Segen e Kumar, 1998] Segen, J. e Kumar, S. (1998). GestureVR: vision-based 3D hand hand interface for spatial interaction. In *Proceedings of the ACM International Multimedia Conference*.

[Sheridan e Ferrell, 1963] Sheridan, T. e Ferrell, W. (1963). Remote manipulative control with transmission delay. *IEEE Transactions on Human Factors in Electronics*, 4:25–29.

[Siskind e Morris, 1996] Siskind, J. M. e Morris, Q. (1996). A maximum-likelihood approach to visual event classification. In Buxton, B. e Cipolla, R., editores, *Proceedings of the European Conference on Computer Vision*, volume 2, pp. 347–360. Springer-Verlag.

[Triesch e von der Malsburg, 1996] Triesch, J. e von der Malsburg, C. (1996). Robust classification of hand postures against complex backgrounds. In *Proceedings of the International Conference on Automatic Face and Gesture Recognition*, pp. 170–175.

[Viterbi, 1967] Viterbi, A. J. (1967). Error bounds for convolutional codes and an asymtotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–267.

[Vogler e Metaxas, 1998] Vogler, C. e Metaxas, D. (1998). ASL recognition based on a coupling between HMMs and 3D motion analysis. In *Proceedings of the International Conference on Computer Vision*, pp. 363–369.

[Waldherr et al., 2000] Waldherr, S.; Romero, R. e Thrun, S. (2000). A gesture based interface for human-robot interaction. *Autonomous Robots*, 9(2):151–173.

[Wu e Huang, 2000] Wu, Y. e Huang, T. S. (2000). View-independent recognition of hand postures. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, volume 2, pp. 84–94.

[Yada e Hirosawa, 1996] Yada, T. e Hirosawa, M. (1996). Gene recognition in cyanobacterium genomic sequence data using the hidden markov model. In *Proceedings of the Fourth International Conference on Intelligent Systems for Molecular Biology*, pp. 252–260. AAAI Press.

[Yamamoto, 1991] Yamamoto, M. (1991). Human motion analysis based on a robot arm model. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 664–665.

[Yamato et al., 1992] Yamato, J. L.; Ohya, J. e Ishii, K. (1992). Recognizing human action in time-sequential images using hidden markov model. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 379–385.

[Yang e Ahuja, 1999] Yang, M.-H. e Ahuja, N. (1999). Recognizing hand gesture using motion trajectories. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 466–472.

[Ynoguti, 1999] Ynoguti, C. A. (1999). *Reconhecimento de fala contínua usando modelos ocultos de Markov*. PhD thesis, Universidade Estadual de Campinas, Campinas, SP, Brasil.

[Young et al., 2005] Young, S.; Evermann, G.; Gales, M.; Hain, T.; Kershaw, D.; Moore, G.; Odell, J.; Ollason, D.; Povey, D.; Valtchev, V. e Woodland, P. (2005). The HTK book (for HTK version 3.3).