Universidade Católica Dom Bosco
Centro de Ciências Exatas e Tecnológicas
Curso de Engenharia de Computação

Filtro de Partículas para Rastreamento de Múltiplos Camundongos

João Bosco O. Monteiro

Prof. Orientador: Hemerson Pistori, Dr.

Relatório Final submetido como um dos requisitos para a obtenção do grau de Engenheiro de Computação.

UCDB - Campo Grande - MS - NOVEMBRO/2005

Resumo

Técnicas de rastreamento baseadas em Visão Computacional têm sido utilizadas em aplicações de controle de qualidade, sistemas de vigilância e sistemas de apoio à análise do comportamento de animais em laboratório. O rastreamento de apenas um único objeto, com pouca oclusão e com um plano de fundo não variante ao longo tempo, é resolvido de forma satisfatória em vários sistemas. No entanto, o rastreamento de múltiplos objetos, de mesmo tipo ou não, com oclusão persistente e plano de fundo complexo e variável, continua sendo um desafio. Uma framework, baseada no filtro de partículas, para o rastreamento de múltiplos objetos é apresentada. O objetivo é desenvolver um módulo para o rastreamento de múltiplos camundongos que será acrescentado ao software que está sendo desenvolvido pelo Projeto Topolino. Esse projeto consiste no desenvolvimento de um sistema computadorizado para a análise do comportamento animal em laboratório, utilizando técnicas de Visão Computacional. Além do rastreamento, o Projeto Topolino concentra estudos de técnicas de segmentação, de construção de interfaces com o usuário e de captura de imagens e vídeos.

Abstract

Computer Vison techniques have been used in quality control, surveillance and analysis of animal behavior. Tracking only one object, without occlusion and clutter background, is a reasonably solved task in some system. However, tracking multiple objects through persistent occlusion and clutter remains a challenge. This work presents a framework, based on particle filter, for tracking multiple objects. Furthermore, a plugin for tracking multiple mice was developed to be added to Topolino software. The Topolino Project consists in developing a computer vison-based system for animal behavior analysis. Tracking, segmentation, interaction human-computer and image and video capture are fields of interest for Topolino Project.

Agradecimentos

Agradeço a Deus, por todas as coisas boas e ruins, que me fizeram crescer como ser humano, durante a graduação. Aos meus pais por terem me proporcinado tantas oportunidades, sem medir esforços. Ao corpo docente do curso de Engenharia de Computação da UCDB. Aos colegas de curso e em especial, aqueles que, por motivos adversos, não puderam concluí-lo: Clineu Sano, Leandro Amaral, Carlos Marcelo e Rogério Maki. A Giovana Bellucci, pela motivação e entusiasmo fornecidos durante a etapa final do curso. Ao Grupo de Pesquisa em Engenharia e Computação (GPEC), professores e acadêmicos. Ao prof. Dr. Albert Schiaveto de Souza e sua equipe do laboratório de Ciências Biológicas e da Saúde. E é claro, ao meu orientador prof. Dr. Hemerson Pistori por compartilhar seus conhecimentos ao longo da realização deste trabalho.

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Visão Computacional é um campo conhecido por seu potencial. Tornar possível que sistemas computacionais possam "ver" e "compreender" o ambiente no qual estão inseridos aparenta ser o último passo para que eles interajam com as pessoas e com o mundo. Enquanto os seres humanos podem compreender as cores e formas a sua volta, mesmo um sistema de Visão Computacional robusto não pode fazê-lo de forma eficaz e barata [6].

Ainda segundo Forsyth *et al* [6] existem diversas áreas que podem ser beneficiadas pela Visão Computacional. A área médica, por exemplo, pode ser auxiliada por sistemas computacionais capazes de identificar importantes fenômenos e eventos. Entretenimento, construção civil e controle de qualidade também são áreas beneficiadas pela Visão Computacional.

A necessidade de automatizar as observações de comportamento de animais em experimentos científicos, como o teste de novos fármacos em camundongos, pode ser suprida utilizando Visão Computacional. Noldus *et al* [24] diz que sistemas de rastreamento por vídeo permitem que pesquisadores estudem o comportamento de animais de forma confiável e consistente durante longos períodos de tempo. Os pesquisadores estão interessados em como os padrões de comportamento são modificados através da exposição a agentes farmacológicos, no contexto dos testes e desenvolvimento de novas drogas e também da geração de novos genótipos e, conseqüentemente, novos fenótipos em programas de mutagênese. Tais atividades são geralmente realizadas em larga escala com a utilização de várias doses de drogas em diversos animais a fim de produzir estatísticas confiáveis [41].

O comportamento dos animais durante esses experimentos pode ser gravado em vídeo de forma manual ou semi-automática. Durante o experimento, o pesquisador observa o animal que, caso demonstre algum padrão de comportamento considerado importante, anota as informações correspondentes àquele comportamento. Ao contrário da observação manual, o rastreamento

por vídeo agrega análise automática de padrões às imagens dos animais observados, a fim de extrair, quantitativamente, medidas a respeito do comportamento do animal.

De forma particular, o rastreamento por vídeo é adequado para mensurar o comportamento locomotor expresso como uma medida espacial (e.g. distância percorrida, velocidade, aceleração), comportamentos raros sucedidos por longos períodos de inatividade e comportamentos que ocorrem durante várias horas ou dias (análise do comportamento diurno, por exemplo) que um observador humano é incapaz de realizar de forma eficiente [38]. Além de não sofrer com a fadiga ou distração do observador, esta abordagem subtrai o componente subjetividade, ou viés, que pode ocorrer quando mais de um observador classifica o mesmo comportamento apresentado pelo animal.

O rastreamento por vídeo não é apenas encontrado em experimentos farmaco-médicos, em ambientes artificiais e controlados, como também em sistemas para o rastreamento de animais em ambientes selvagens. Em Tweed *et al* [40], é descrito o trabalho inicial para o desenvolvimento de um método robusto para o rastreamento de animais em seu *habitat* natural. As peculiaridades dos ambientes selvagens tornam o rastreamento individual de cada animal uma tarefa árdua. De forma geral, as cores e texturas não são características suficientes para distinguir os animais pois a natureza possui artifícios para dificultar a identificação visual, como a camuflagem, embora em alguns casos a textura possa claramente identificar animais como zebras e tigres por conta de suas listras, como mostra o trabalho de Hermes *et al* [10] ilustrado na Figura 1.1.

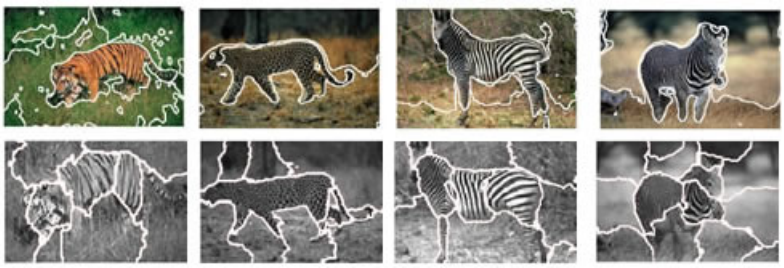

Figura 1.1: Segmentação utilizando aglomeração baseada em informações de textura (*Parametric Distributional Clustering*). Fonte: Hermes *et al* [10].

Mas o rastreamento utilizando técnicas de Visão Computacional não se restringe apenas a animais selvagens e de laboratório. O rastreamento de seres humanos em cenas dinâmicas tem se tornado um importante tópico de pesquisa uma vez que os humanos são os atores principais de diversas

atividades que exigem monitoramento, como em sistemas de vigilância, por exemplo [43].

O rastreamento isolado de apenas um objeto ou de vários pequenos objetos, com oclusão[1] transiente, pode ser realizado de forma confiável e satisfatória em alguns sistemas. No entanto, rastrear múltiplos objetos ou pessoas em um ambiente com muita aglomeração e oclusão persistente permanece sendo um desafio.

Todas as aplicações apresentadas nesta breve introdução têm em comum o rastreamento por vídeo de múltiplas entidades, sejam elas de origem animal ou não. O objetivo deste trabalho é o estudo e a implementação de uma técnica para rastreamento de múltiplos camundongos durante experimentos em laboratório. Os módulos criados serão adicionados no software que está sendo desenvolvido pelo Projeto Topolino, o qual consiste no desenvolvimento de um sistema computadorizado para a segmentação e rastreamento de animais de laboratório utilizando técnicas de Visão Computacional. Este projeto está sendo desenvolvido pelo Grupo de Pesquisa em Engenharia e Computação e pelo Centro de Ciências Biológicas e da Saúde da Universidade Católica Dom Bosco. Mais informações sobre o Projeto Topolino podem ser obtidas no sítio http://www.gpec.ucdb.br/topolino.

Outro beneficiado por este trabalho poderá ser o Projeto SIGUS [28], que visa o desenvolvimento de uma plataforma de apoio ao desenvolvimento de sistemas para inclusão digital de pessoas com necessidades especiais. Possuindo frentes de pesquisa como reconhecimento de gestos baseados na Língua Brasileira de Sinais e na detecção da direção do olhar, o Projeto SIGUS poderá, por exemplo, empregar técnicas de rastreamento de múltiplos objetos para detectar a face e as mãos do usuário na mesma imagem. Mais informações sobre o Projeto SIGUS podem ser obtidas no sítio http://www.gpec.ucdb.br/sigus.

A implementação do módulo para rastreamento de múltiplos camundongos foi realizada utilizando a linguagem Java[2]. Ferramentas de código-fonte aberto como o ImageJ[3], software de processamento e análise de imagens em Java, foram utilizadas para auxiliar no desenvolvimento do módulo.

Este texto está estruturado da seguinte forma: o Capítulo 2 apresenta a fundamentação teórica com conceitos referentes à segmentação, por limiarização e por diferença de imagens, bem como a apresentação do problema de rastreamento além de uma introdução sobre rastreamento baseado em modelos. No Capítulo 3 encontram-se os trabalhos correlatos, já o Capítulo

[1]Oclusão é a situação na qual o objeto ou entidade de interesse é ocultado por outra entidade, que pode ou não ser do mesmo tipo

[2]Mais informações no sítio http://java.sun.com

[3]Pode ser encontrado no sítio http://rsb.info.nih.gov/ij/

4 trata da técnica implementada no módulo de rastreamento que é o filtro de partículas. O Capítulo 5 mostra detalhes da implementação e de que maneira o filtro foi empregado, o Capítulo 6 detalha os experimentos e analisa os resultados. O Capítulo 7 apresenta as considerações finais.

Capítulo 2

Fundamentação Teórica

O rastreamento de objetos em seqüências de imagens é uma tarefa importante em sistemas de vigilância, sistemas de análise de imagens meteorológicas, sistemas de apoio à análise de imagens médicas, detecção de alvos em sistemas balísticos, sistemas que utilizam gestos para a interação homem-máquina e análise do fluxo de multidões, dentre outras.

O primeiro passo para o rastreamento de objetos é detectar as regiões que os contêm. Isso pode implicar a separação do plano de fundo ou a detecção de algum tipo de movimento. A esta separação dá-se o nome de segmentação que será tratada mais adiante. Além disso, uma boa modelagem do problema de rastreamento leva a uma melhor compreensão do mesmo e auxilia a reduzir e a lidar com a sua complexidade.

Para dar suporte ao entendimento das técnicas apresentadas nos próximos capítulos é importante introduzir conceitos a respeito da inferência bayesiana e da estimação estocástica, uma vez que o problema de rastreamento de objetos pode ser interpretado como um problema de estimação de estados.

Neste capítulo serão apresentados alguns fundamentos e algumas técnicas de segmentação e extração do plano de fundo, já que este é o procedimento inicial para a realização do rastreamento, utilização de modelos para o rastreamento de objetos, modelagem e inferência bayesiana, bem como uma visão geral sobre a estimação estocástica de estados.

2.1 Segmentação

O processo de segmentação consiste em subdividir uma imagem em partes ou objetos que a compõe [8]. A segmentação deve prosseguir até que os objetos de interesse estejam completamente isolados, no entanto, como nem sempre isso é possível, às vezes é necessário trabalhar com níveis de segmentação que

variam conforme o problema.

De forma geral, para imagens monocromáticas as propriedades de descontinuidade e similaridade dos níveis de cinza são utilizadas para realizar a segmentação. A descontinuidade permite identificar e separar pontos isolados e bordas da imagem que possuem uma mudança abrupta dos níveis de cinza. Já a similaridade é a propriedade na qual se baseiam as técnicas de crescimento de regiões, limiarização, divisão e fusão de regiões. Em imagens coloridas costuma-se utilizar a similaridade entre os histogramas em cores para efetuar a segmentação [29]. Diversas medidas de similaridades, que constituem uma extensa área de pesquisa, podem ser adotadas como a magnitude e a direção do gradiente, distância Euclidiana, entre outras.

2.1.1 Segmentação por Limiarização

Umas das técnicas mais importantes empregadas na segmentação de imagens é a limiarização, que consiste em, através da escolha de um limiar L, separar os pixels da imagem em grupos distintos. Isso permite classificar os pixels da imagem como sendo pertencentes ao plano de fundo ou ao plano de interesse, caso haja agrupamentos de pixels que favoreçam uma separação distinta.

Também é possível trabalhar com vários limiares a fim de separar os pixels da imagem em mais grupos, no entanto, esta tarefa fica cada vez mais difícil pois torna-se complicado estabelecer múltiplos limiares que efetivamente isolem as regiões de interesse. Percebe-se então que o desempenho da limiarização está intimamente relacionado com quão bem está particionado o histograma da imagem.

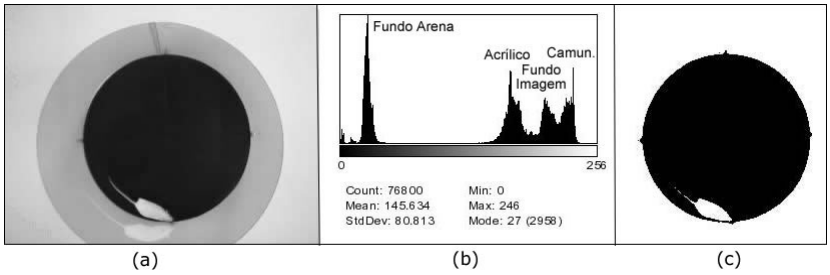

Figura 2.1: (a) Imagem capturada por uma *webcam* em tons de cinza. (b) Histograma da Imagem. (c) Resultado obtido após a aplicação do limiar 120.

Para exemplificar, considere a Figura 2.1a obtida através de uma *webcam* do experimento Campo Aberto. Na região inferior da imagem nota-se um ruído intenso causado pelo reflexo do camundongo no aparato de acrílico e também variações nos tons de cinza do plano de fundo.

O ruído é toda informação indesejada na imagem ou sinal e pode ser causado, por exemplo, pela variação da iluminação, pela conversão analógica para digital realizada pelo dispositivo de captura, por reflexo, pela forma de armazenamento da imagem em formato digital ou também gerado pelo próprio sensor de captura de imagem. Para minimizar o ruído, muitos sistemas empregam filtros de suavização, como o filtro da mediana ou o gaussiano, na etapa de segmentação.

A Figura 2.1b mostra o histograma da imagem e percebe-se que os pixels estão aglomerados em dois grupos: o fundo preto da arena e as partes mais claras que correspondem a parte externa à arena, as paredes de acrílico e ao camundongo. Portanto, uma limiarização é capaz de segmentar o objeto de interesse com relativa precisão. O resultado obtido ao aplicar o limiar 120 (valor do tom de cinza), que foi escolhido variando o seu valor até encontrar um que fosse considerado satisfatório, pode ser visualizado na Figura 2.1c. Este experimento foi realizado utilizando a ferramenta ImageJ.

2.1.2 Segmentação por Diferença entre Imagens

O movimento é um importante atributo para que os seres humanos e os animais possam distinguir um objeto de interesse do plano de fundo. Considerando um dispositivo de captura estacionário e uma seqüência de imagens obtidas através dele, ao comparar duas imagens subseqüentes, pixel a pixel, e realizando a subtração dos seus valores em tons de cinza, é possível determinar se houve movimento ou não uma vez que os componentes estacionários são cancelados na subtração. Aplicando-se um limiar L na imagem resultante da subtração, torna-se possível separar os pixels que representam o movimento realizado por algum componente da imagem [8].

É importante salientar que este movimento pode ser tanto do plano de fundo, se este possuir algum elemento que pode se mover ao longo do tempo, como também do objeto de interesse. A escolha do limiar pode favorecer a segmentação do plano de interesse caso o limiar escolhido seja capaz de distinguir se o movimento foi realizado por um objeto do plano de fundo ou se foi realizado pelo objeto de interesse.

Outra alternativa seria obter uma imagem, em tons de cinza, apenas do plano de fundo para servir de referência. Nas imagens subseqüentes, subtrai-se os valores dos pixels da imagem atual, que contém o objeto a ser rastreado, dos valores dos pixels da imagem de referência. O resultado é uma imagem que contém apenas o objeto de interesse, considerando que o dispositivo de captura seja estacionário e o plano de fundo seja imóvel e que não sofra variações ao longo do tempo. O processo é ilustrado na Figura 2.2.

Caso o plano de fundo não seja totalmente imóvel, como é o caso do exem-

plo, no qual o aparato de acrílico se movimenta sempre que o camundongo o toca, surgem ruídos na imagem, como pode ser observado na Figura 2.2c. Para minimizar o problema, podem ser empregados filtros de suavização ou operadores morfológicos de fechamento.

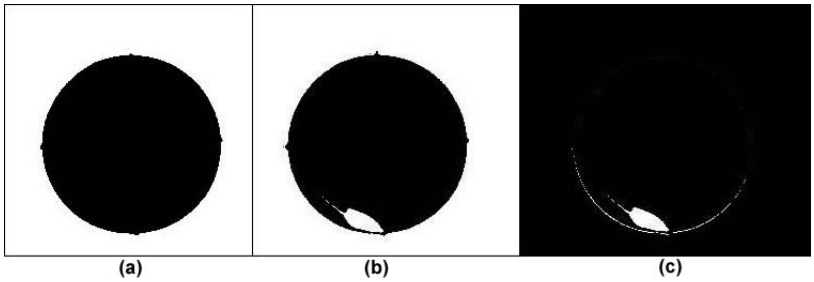

Figura 2.2: (a) Imagem de referência já limiarizada. (b) Imagem da seqüência que contém o objeto de interesse. (c) Resultado obtido após subtração da imagem atual pela imagem de referência.

Em seqüências de imagens do mundo real o plano de fundo geralmente é complexo e freqüentemente apresenta variações na iluminação, causada por exemplo, por variações no tempo (dia ensolarado, nuvens obstruindo a luz solar, sombras) ou calibração do dispositivo de captura de imagens causando ruído. Portanto, é fundamental criar um modelo capaz de descrever o plano de fundo de forma correta e como ele pode se alterar ao longo do tempo, é necessário que o modelo do plano de fundo consiga perceber e se adaptar a essas novas condições. Para isto diversas técnicas têm sido propostas, como por exemplo o filtro de Kalman para a atualização do plano de fundo [35] e a modelagem do fundo utilizando Mistura de Gaussianas. Informações como textura e cor podem ser boas características para a modelagem do plano de fundo, dependendo, é claro, da aplicação em questão.

Para inibir a influência demasiada da variação da iluminação na modelagem do plano de fundo, muitos autores utilizam o espaço de cores HSV [1], uma vez que nele é possível separar a componente V, que é a responsável por determinar a intensidade luminosa presente na imagem (brilho). Desta forma, a matiz é relacionada à cor predominante (faixa do espectro) enquanto a saturação determina quão pura ela é, já o valor caracteriza a quantidade de energia luminosa existente na cor. A quantidade de energia é a responsável por determinar se uma cor é mais clara ou mais escura enquanto a matiz

[1]No sítio http://rsb.info.nih.gov/ij/plugins/color-inspector.html é possível encontrar um módulo para o software ImageJ capaz de mostrar as cores de uma imagem em um espaço de cores 3D.

é o que diferencia, por exemplo, a cor vermelha da cor verde e a saturação permite diferenciar a cor vermelha e a cor rosa.

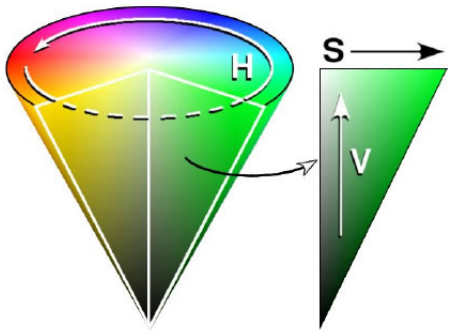

Figura 2.3: O espaço de cores HSV.

A Figura 2.3 ilustra o espaço de cores HSV. A matiz é determinada na base do cone. Quanto mais se afasta do eixo do cone, maior a saturação da cor. Já afastando-se do base em direção ao vértice, a energia diminui.

2.2 Rastreamento

O objetivo principal do rastreamento é seguir objetos em uma seqüência de imagens obtendo informações relevantes para a aplicação como a posição do objeto, velocidade, aceleração, forma aproximada, entre outras. A detecção e rastreamento de objetos pode ser usada diretamente em sistema de detecção de intrusos, contagem de multidões, indexação de vídeos baseado em conteúdo e na interação homem-máquina permitindo que o computadores interajam com mais facilidade e naturalidade com os seres humanos [42].

Algumas das dificuldades são determinar quais são os objetos que devem ser rastreados e localizá-los nas imagens subseqüentes, com o agravante da variação da forma do objeto caso ele não seja rígido, como as mãos por exemplo. Outras dificuldades aparecem quando o objeto se desloca com aceleração não uniforme em um plano perpendicular ao plano da imagem, causando variações no tamanho do objeto [20][32] ou ainda, quando o objeto a ser rastreado está total ou parcialmente oculto por outro elemento da imagem.

Para Goldenstein [7] quando se estuda ou se resolve um problema, primeiramente é necessário uma formulação matemática apropriada para o mesmo. Uma boa representação matemática permite inferir propriedades de uma entidade real e compreendê-la antes de interagir com a mesma. Neste contexto, é

necessário escolher um conjunto de parâmetros que descrevem a configuração de um objeto ao qual dá-se o nome de *vetor de estado*.

Adicionalmente, os parâmetros escolhidos podem ser intrínsecos ao problema mas nem sempre podem ser mensurados diretamente, portanto são necessário outros parâmetros relacionados àqueles desconhecidos a fim de inferir os seus respectivos valores. Este tipo de medida indireta de um parâmetro desconhecido é chamado de *vetor de observação*.

De forma geral, as observações realizadas podem não ser precisas ou podem não explicar claramente qual é o estado do sistema. Conseqüentemente, o modelo matemático empregado não é exato e por isso é fundamental incluir na modelagem a noção de *incerteza*. Por exemplo, o filtro de Kalman, que realiza a predição de estados, utiliza variáveis Gaussianas aleatórias para representar as incertezas no sistema.

Para cada imagem de uma seqüência ou um quadro de um vídeo no qual desejamos realizar algum tipo de rastreamento, é necessário buscar o objeto de interesse com base em um modelo de referência que descreva a sua aparência ou estado. Esta referência pode ser baseada na forma como os pixels estão aglomerados, baseada nos contornos dos objetos, na posição espacial ou ainda em modelos baseados em cores, como os descritos em Pérez *et al* [29].

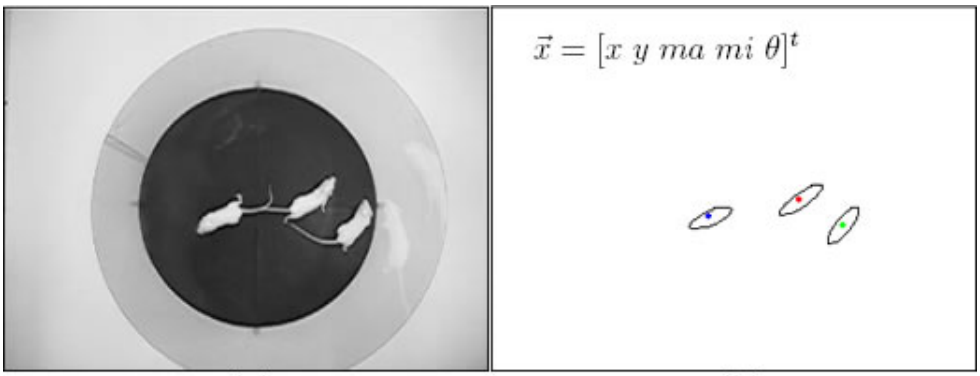

Figura 2.4: Modelo de estado de um camundongo, com as coordenadas x e y do centro de massa, eixo maior (ma) e menor (mi) da elipse e ângulo de inclinação θ.

A Figura 2.4 mostra três camundongos que utilizam o mesmo modelo de estado. A forma dos camundongos é aproximada à uma elipse e o estado é composto pelas coordenadas x e y do centro de massa, eixo maior e menor da elipse além do ângulo de inclinação.

No entanto, não basta apenas modelar o estado do objeto de interesse como também é de fundamental importância modelar o comportamento dinâmico do sistema e para isto deve-se levar em conta os estados passados e também as observações realizadas. Em muitos casos é suficiente considerar que o próximo estado do sistema dependa apenas do estado atual e das observações, reduzindo assim a complexidade.

Em sistemas lineares, o comportamento dinâmico do sistema pode ser reduzido a uma combinação linear dos estados anteriores e das observações realizadas. Estes sistemas podem ser facilmente resolvidos matematicamente utilizando soluções analíticas bem conhecidas e eficientemente implementadas.

Para completar a modelagem do sistema, considera-se um modelo para as observações realizadas baseado na combinação dos parâmetro que levarão à inferência do parâmetro oculto desejado. A modelagem da observação visa determinar os estados do sistema de forma precisa e por isso existe a necessidade da predição realizada pelo modelo ser a mais fiel possível. Para a predição ótima dos estados, muitos autores se apóiam na Teoria Bayesiana para a construção do modelo de observação. Vale ressaltar que nem todos os sistemas de rastreamento possuem este tipo de modelagem porém a utilização desta organização pode ajudar a lidar com a complexidade do problema.

Em uma abordagem diferente daquela proposta por Goldenstein, Spengler *et al* [37] propõe quatro níveis de conhecimento que devem ser alcançados e implementados a fim de reduzir a complexidade do rastreamento de múltiplos objetos. O primeiro e mais complexo nível diz respeito aos pixels da imagem propriamente ditos que podem fornecer informações como correspondência dos pixels ao modelo do plano de fundo, movimento realizado através da diferença de imagens ou se os pixels correspondem à cor da pele. Por exemplo, para a detecção de uma face humana, o primeiro passo é encontrar os pixels pertencentes a um determinado modelo de cor da pele. Este nível é caracterizado principalmente pelas informações que os pixels da imagem podem fornecer.

O próximo é o nível de objeto, o qual leva em conta a relação entre os pixels que formam um objeto e que dizem respeito a sua aparência e comportamento. Tipicamente, o nível de objeto é representado por modelos de objetos que podem ser simples ou mais complexos conforme a necessidade da aplicação. A Figura 2.5 ilustra um modelo de objeto mais complexo e sofisticado do que simplesmente um modelo de objeto baseado em histogramas coloridos ou em valores na escala de cinza.

O nível subseqüente, denominado nível composto, é menos complexo e mais representativo. Este nível tem por objetivo modelar a interação entre os objetos, de mesmo tipo ou não, presentes na imagem com o objetivo

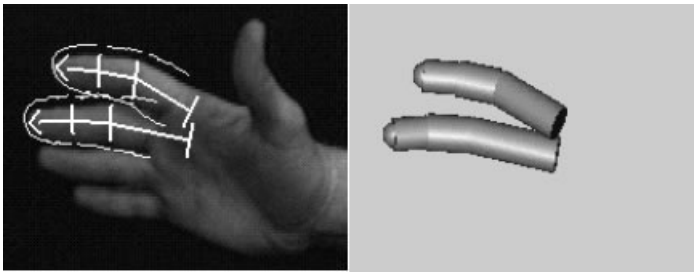

Figura 2.5: Modelo de objeto que representa dedos humanos. Fonte: Rehg *et al* [31].

de prover informações sobre quando novos objetos aparecem na imagem e quando outros a deixam.

Na hierarquia adotada por Spengler, o nível mais alto da pirâmide de conhecimentos é a cena. Considerando, por exemplo, um sistema de vigilância, é razoável modelar as regiões da imagem por onde uma pessoa possa aparecer visto que uma pessoa surgir do teto ou das paredes é improvável. Os conhecimentos referentes ao modelo da cena são geralmente muito específicos e não reutilizáveis, já que delineiam características exclusivas da aplicação.

2.3 Estimação Estocástica e Métodos Monte Carlos Seqüênciais

A modelagem da dinâmica dos sistemas é uma importante ferramenta de análise estatística que tem atraído a atenção de pesquisadores de diversas áreas da ciência. O modelo linear do espaço de estados, que é o modelo mais utilizado, tem sido objeto de estudo já a algum tempo em sistemas de controle que variam ao longo do tempo [9]. Apesar de sua complexidade, os modelos de estado não-lineares/não-gaussianos são importantes para algumas aplicações como o rastreamento de múltiplos objetos. Esta seção é uma breve apresentação da estimação estocástica de estados e dos métodos Monte Carlo Seqüenciais.

Um sistema dinâmico discreto é aquele que possui uma seqüência de distribuições de probabilidades $P_t(x_t)$, indexadas por um tempo discreto t. A variável x_t representa o estado do sistema no tempo t e a função P() determina a probabilidade daquele estado realmente ser observado. O estado do sistema pode evoluir de três maneiras basicamente:

1. Aumento da dimensão, ou seja, x_{t+1} possui uma componente a mais que x_t;

2. Diminuição da dimensão, ou seja, x_{t+1} possui uma componente a menos que x_t;

3. Sem modificação, $x_t = x_{t+1}$;

Na maioria das aplicações, a diferença entre $P(x_t)$ e $P(x_{t+1})$ é provocada pela adição de mais informações no sistema. O que é interessante e útil nestes sistemas é a possibilidade de extrair informações a respeito do estado atual e também da próxima organização dos estados do sistema através da estimação dos estados. Isso é obtido através de três etapas que caracterizam os métodos Monte Carlo Seqüênciais:

1. Predição: $P_t(x_{t+1}|x_t)$;

2. Atualização: $P_{t+1}(x_t)$;

3. Nova Estimação: $P_{t+1}(x_{t+1})$;

Uma das aplicações dos métodos Monte Carlo Seqüênciais é a modelagem do espaço de estados de um sistema dinâmico. Para tal, é necessário utilizar uma equação que descreve a observação, ou seja, o estado atual do sistema e uma equação que define o espaço de estados que pode ser representada por um processo Markoviano [16].

Outra característica dos Métodos Monte Carlo é utilizar variáveis aleatórias para resolver problemas complexos[21]. A idéia central é utilizar parâmentros para representar a distribuição real ou hipotética de algum problema, e realizar a estimação dos valores desses parâmetros amostrando essa distribuição[15].

Neste ponto, é possível notar a relação entre os métodos Monte Carlo Seqüênciais e o rastreamento baseado em modelos. Dessa forma pode-se modelar o problema do rastreamento de múltiplos objetos como um problema de estimação de estados em um sistema dinâmico, utilizando métodos Monte Carlos Seqüênciais e suas especializações, como o filtro de partículas.

Capítulo 3

Trabalhos Correlatos

Ao analisar atentamente as técnicas de rastreamento de múltiplos objetos que existem atualmente, é possível identificar duas abordagens como comenta Spengler *et al.* [37]. A primeira diz respeito à utilização de vários rastreadores de apenas um único objeto, em outras palavras, para cada objeto da cena existe um rastreador que o segue. Muitas vezes essa abordagem não possui a visão completa do espaço de soluções e nem compreende o problema do rastreamento de múltiplos objetos como um todo, gerando soluções subótimas. A outra abordagem é utilizar um rastreador de múltiplos objetos puro, que possui uma visão completa do domínio do problema. No entanto, esta abordagem também tem suas desvantagens como, por exemplo, o alto custo pago pela alta dimensionalidade do espaço de estados.

Diversos sistemas de rastreamento de múltiplos objetos têm sido desenvolvidos e algumas técnicas empregam a combinação de identificação de regiões (*blob identification*) e subtração do plano de fundo. Nesta abordagem o modelo de observação utilizado para interpretar a imagem permite, facilmente, que dois objetos ocupem o mesmo ponto no espaço, pois rastreadores independentes tendem a encontrar os alvos que mais se aproximam do modelo utilizado. Para resolver esse problema, Marccormick [17] propõe um princípio probabilístico de exclusão a fim de impedir que dois objetos se integrem quando suas configurações se tornarem similares, e dessa forma, o modelo de observação continue interpretando os dados da imagem no âmbito de dois objetos.

Perner [27] também utiliza a técnica de subtração de plano de fundo e uma imagem de referência para inferir a localização de suínos em um estábulo. A cada imagem adquirida, os objetos são separados do plano de fundo utilizando a imagem de referência e os limites físicos do estábulo são obtidos aplicando um limiar ao histograma da imagem em tons de cinza, a fim de determinar se as fronteiras do estábulo estão ocultando os suínos. De posse

dessas informações e determinando a posição dos objetos com base no seu centro de massa e um vetor de movimento, é possível inferir a posição subseqüente do objeto, mesmo que ele esteja sendo ocultado por outro objeto ou pelos limites do estábulo.

Em várias técnicas de rastreamento os contornos dos objetos são aproximados por modelos paramétricos como em Isard *et al* [12] que aproxima o contorno de objetos a curvas B-spline e também existem técnicas que utilizam contornos ativos para modelar a forma dos objetos. A utilização de contornos ativos é mais empregada quando se necessita extrair informações sobre o comportamento do objeto de interesse. Twining *et al* [41] descreve a utilização de contornos ativos para o rastreamento robusto de roedores em ambiente de laboratório com objetivos similares aos do Projeto Topolino. No entanto, não trata do rastreamento de múltiplos objetos nem de uma possível interação entre eles.

Nguyen e outros [22] argumenta que utilizar modelos com um número fixo de parâmetros pode prejudicar o rastreamento, uma vez que, se os contornos são aproximados a um polígono com número fixo de vértices por exemplo, não é possível rastrear objetos com formato arbitrário e ainda, esta aproximação impõe uma suavização dos contornos levando à identificação inadequada dos cantos dos objetos. Por isso, Nguyen propõe uma abordagem que não utiliza modelos paramétricos e encara o problema de rastreamento como um problema de segmentação, que consiste em atualizar o contorno do objeto de interesse, na imagem atual, com base nas informações obtidas a partir da imagem anterior [22].

O primeiro passo da técnica apresentada por Nguyen e outros é estimar qual será o próximo contorno do objeto com base no teorema de Bayes. Em seguida, o contorno estimado sofre uma operação morfológica de alargamento (*thickening*) dando origem a duas regiões, delimitadas pelo contorno previsto e pelo contorno criado após a operação de alargamento. Depois disso, emprega-se o algoritmo divisor de águas (*watershed*) a fim de obter o contorno atual do objeto. O algoritmo é ilustrado na Figura 3.1 e um exemplo de resultado pode ser visto na Figura 3.2.

A detecção de bordas desempenha um papel muito importante nesta técnica, por isso foram utilizados três mapas de bordas computados para cada imagem e que dizem respeito às bordas baseadas na intensidade do contorno atual, do contorno estimado e ainda um mapa que mantém informação sobre as bordas baseadas no movimento realizado pelos objetos da cena (*motion edges*). No entanto, esta técnica não lida com rastreamento de múltiplos objetos e também é necessário que o contorno no primeiro quadro seja indicado pelo usuário.

Por outro lado, utilizando modelos paramétricos pode-se modelar o ras-

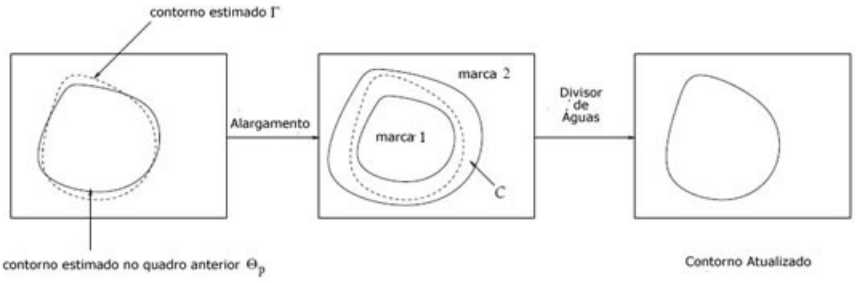

Figura 3.1: Ilustração da execução de um passo do algoritmo de rastreamento de Nguyen *et al.* Fonte: Nguyen [22].

Figura 3.2: Exemplos de resultados obtidos por Nguyen *et al.* Fonte: Nguyen [23].

treamento como um problema de estimação. Técnicas poderosas, como filtro de Kalman [7] e filtros Monte-Carlo [16], podem ser associadas aos modelos a fim de inferir o valor dos parâmetros. Em situações nas quais é necessário rastrear um grande número de objetos do mesmo tipo ou naquelas em que necessita-se lidar com objetos de formato complexo ou articulados, a utilização de modelos para caracterizar o comportamento dinâmico e a aparência dos objetos é predominante. Este tipo de rastreamento consiste em encontrar uma configuração que se aproxime do modelo estipulado, o que resulta em uma estimação robusta da posição do objeto. O rastreamento do movimento das mãos utilizando um modelo 3D e a utilização de grafos para a ordem de visibilidade dos objetos é descrito em [31].

Outra técnica utilizada para rastrear múltiplos objetos é o filtro de partículas, que utiliza múltiplas amostras discretas para representar a distribuição da probabilidade da posição dos objetos rastreados [13]. O filtro de partículas e as suas variações (Condensação[12], Condensação Subordinada) permitem o desenvolvimento de ferramentas robustas de rastreamento uma vez que estas técnicas não são limitadas a sistemas lineares nem requerem que o ruído presente seja Gaussiano como acontece no Filtro de Kalman. Mais detalhes sobre esta técnica são apresentados no Capítulo 4.

Ao contrário da utilização de grafos para determinar se um objeto está

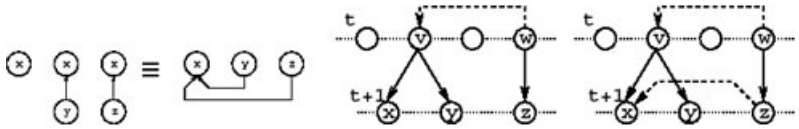

Figura 3.3: Representação de partículas de múltiplos objetos. À esquerda, representação básica e à direita, propagação temporal das conexões de subordinação. Fonte: Tweed *et al* [39].

sendo ocultado por outro, Tweed *et al* [39] extende o algoritmo de Condensação introduzindo a Subordinação e o princípio probabilístico de exclusão para lidar com a oclusão em um conjunto de partículas. Uma partícula é dita subordinada quando outra a oculta total ou parcialmente, conforme o esquema mostrado na Figura 3.3. Kahn *et al*[13] também utiliza filtro de partículas e um modelo de movimento baseado em campos aleatórios de Markov (MRF - *Markov Random Fields*) a fim de rastrear corretamente objetos que interagem entre si, no caso, formigas (Figura 3.4).

Figura 3.4: Exemplo da utilização do Princípio de Exclusão baseado em MRF utilizado por Kahn *et al* [13].

Katja *et al* [25] propõe um filtro de partículas adaptativo baseado em cores. As cores podem fornecer características visuais atrativas para o rastreamentos de objetos não rígidos, no entanto, a cor de um objeto pode variar ao longo do tempo devido a iluminação, mudança do ângulo de visão e dos parâmetros do dispositivo de captura de vídeo.

Para lidar com estes problemas de mudança de aparência foram empregados modelos de cor que se adaptam durante o processamento baseados em uma função que atribui pesos maiores às distribuições que possuem histogramas coloridos mais semelhantes. Por outro lado, Raja *et al*[30] utiliza modelos coloridos baseados no espaço de cores HS (H - matiz, S - saturação) para minimizar a influência da iluminação representada pelo componente V (V - valor, intensidade) e para o rastreamento e segmentação, emprega Mistura de Gaussianas e o Algoritmo EM.

Capítulo 4

Filtro de Partículas

O rastreamento de múltiplos objetos pode ser interpretado como um problema de estimação de estados de um número desconhecido de alvos que se movimentam em um sistema dinâmico[11] e com a utilização de filtros preditivos é possível estimar o estado ótimo do sistema. Para isso, deve-se utilizar um modelo matemático da dinâmica do sistema para propagar os estados e depois, combinar a estimatição com a probabilidade dos estados realmente terem ocorrido [7]. Existem diversos filtros preditivos, como o Filtro de Kalman e o filtro de partículas, que são apropriados para diferentes tipos de modelagens do sistema. Os filtros preditivos são particularmente interessantes quando se trabalha com dados corrompidos e ruído, como por exemplo, durante o rastreamento de objetos que são constantemente ocultados por outros.

O Filtro de Kalman considera que as variáveis aleatórias do sistema respeitam uma distribuição Gaussiana e que os modelos, de dinâmica e de observação, são lineares. Diversas vezes uma variável Gaussiana aleatória não é capaz de descrever ou aproximar o estado de um sistema dinâmico, como por exemplo, quando existem simultaneamente várias modas. Uma possível abordagem para lidar com essa limitação é utilizar modelos paramétricos mais poderosos como a Mistura de Gaussianas [1][30]. Outra alternativa, é empregar uma representação baseada em amostras (ou baseada em partículas) da distribuição.

A representação de uma distribuição baseada em partículas não é descrita por parâmetros, e sim, por um conjunto selecionado de amostras, por isso é chamada de não-paramétrica. Regiões com maior densidade possuem maior concentração de amostras do que regiões com menor densidade de probabilidade. A Figura 4.1 demostra uma possível representação baseada em partículas. Uma vez que o filtro de partículas utiliza esse tipo de representação, é possível empregá-lo em uma grande quantidade de sistemas que

possuem funções de densidade de probabilidade multimodais . Outra característica importante, é que tanto a dinâmica como a observação podem ser não-lineares[14].

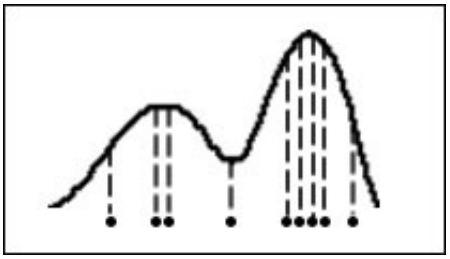

Figura 4.1: Exemplo de representação baseada em partículas.

Algoritmo 4.1 Filtro de Partículas

1: $S^0 = inicializarAmostra()$
2: $k = 1$
3: $TAM = tamanhoDaSequencia()$
4: $M = numeroDeAmostras$
5: **enquanto** $k < TAM$ **faça**
6: **para** $j = 0$ to M **faça**
7: $x_j^{k+1} = f(x_k) + w(k, j)$ {Predição}
8: **fim para**
9: $obs^k = sense()$ {Mede o estado atual do sistema - Observação}
10: **para** $j = 0$ to M **faça**
11: $w_j^{k+1} = atribuaPesos(w_j^k, obs)$ {Atualização}
12: **fim para**
13: $normalizePesos(w)$
14: **se** $ESS(w) < \beta * M$ **então**
15: $S^{k+1} = reamostre(S^k)$ {Correção}
16: **fim se**
17: $k = k + 1$
18: **fim enquanto**

O filtro de partículas, descrito no Algoritmo 4.1, tem natureza iterativa e possui três etapas: *predição*, *atualização de pesos* e *correção*. Formalizando o problema, considere $\vec{x}_k$ como sendo a variável aleatória multidimensional que representa o estado do sistema no tempo k e $S_k = [x_j^k, w_j^k] : j = 1...M$ um conjunto de M partículas que representam a distribuição do sistema. Cada partícula possui um peso associado w_j que indica o quanto ela é significativa para a distribuição.

Antes de iniciar a execução das três etapas citadas, o conjunto de amostras precisa ser inicializado , o que pode ser feito de diversas maneiras. A primeira opção é inicializar as partículas aleatoriamente, sendo possível também, inicializar o conjunto respeitando algum tipo de distribuição, como a normal ou uniforme por exemplo. A desvantagem dessa abordagem é que o filtro leva algumas iterações para determinar quais partículas representam melhor a distribuição do sistema. Por isso, opta-se por realizar a inicialização das partículas baseda em uma observação prévia [11][26]. Inicialmente, todas as partículas são igualmente significativas para o sistema, portanto os pesos são inicializados como

$$w_j^k = \frac{1}{M} \tag{4.1}$$

para $k = 1$ e $j = 1...M$.

A etapa de predição consiste em aplicar a função de dinâmica f sobre o conjunto de partículas e obter os novos possíves estados do sistema:

$$\vec{x}_{k+1} = f(\vec{x}_k) + \vec{w}_k \tag{4.2}$$

$\vec{w}_k$ representa o ruído da predição, geralmente Gaussiano e com média 0. É de fundamental importância que o ruído seja diferente para cada partícula do conjunto como será visto mais adiante.

Assim que é finalizada a predição, deve-se *observar* o real estado do sistema:

$$\vec{y}_k = h(\vec{x}_k) + \vec{v}_k \tag{4.3}$$

$h(\vec{x}_k)$ é a função que extrai as informações sobre o estado atual do sistema (*observação*), e $\vec{v}_k$ é o ruído da observação, geralmente Gaussiano e com média 0.

O próximo passo é combinar a predição com a observação a fim de realizar uma estimação do estado do sistema. Esta etapa de atualização consiste em atribuir novos pesos para as partículas de acordo com a probabilidade condicional

$$w_j^{k+1} = P(\vec{x}_{k+1}|\vec{y}_{k+1}) \tag{4.4}$$

ou seja, as partículas que tem maior probabilidade de ocorrerem, dado a observação $\vec{y}_{k+1}$, recebem um peso maior do que aquelas que ocorrem com menos freqüência. Novamente é importante salientar que tanto as funções f e h podem ser não-lineares e que os ruídos v e w podem não ser Gaussianos. Estas características tornam o filtro de partículas atraente para problemas que não podem ser resolvidos através do Filtro de Kalman, que possui uma solução algébrica fechada e que lida apenas com sistemas lineares e com ruído Gaussiano.

A última etapa realizada pelo filtro é a correção, também chamada de reamostragem (*resampling*). Uma vez que o peso de cada partícula do conjunto já foi atualizado, é necessário decidir quais delas serão propagadas para a próxima iteração. Devido a essa característica, curiosamente é possível encontrar na literatura este filtro com o nome de *Survival of the fittest*. Após algumas iterações, diversas amostras podem ter se distanciado tanto da observação que o seu peso é próximo do zero e por isso contribuem pouco para a distribuição. O intuito é replicar partículas com maiores pesos, e conseqüêntemente mais importantes para a distribuição, e descartar aquelas que possuem pouca probabilidade de ocorrer [34].

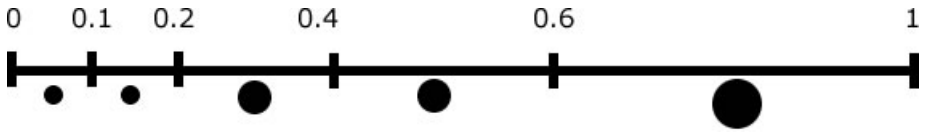

Figura 4.2: Exemplo de mapeamento para a etapa de correção.

A implementação desta etapa consiste em mapear todas as amostras em um intervalo [0,1] e atribuir, de acordo com os seus respectivos pesos, uma probabilidade delas serem propagadas para a próxima iteração. A Figura 4.2 ilustra um exemplo de mapeamento no qual, partículas com pesos maiores possuem porções maiores no intervalo. Finalmente, sorteia-se um número n aleatoriamente, de acordo com uma distribuição uniforme, e a partícula que corresponde à porção n do intervalo é escolhida para ser propagada. Em contrapartida, se a cada iteração for feita uma reamostragem, corre-se o risco das partículas convergirem rapidamente para aquela que possui o maior peso, o que também não é desejável[18][7][34].

A solução discutida por Rekleitis [33] é medir a qualidade de representação das partículas, e com base em um limiar β, que corresponde a uma determinada porcentagem do conjunto de amostras, decidir se a etapa de correção deve ser realizada ou não. Durante a implementação do filtro, o cálculo do coeficiente de variação e do tamanho efetivo da amostra foram os mesmos apresentados por Rekleitis.

Diversos métodos de propagação têm sido propostos e em Rekleitis[34] são discutidos os três mais comuns, enquanto Carpenter[2] apresenta um algoritmo mais eficiente, de ordem $O(n)$, para a realização da reamostragem. Como agora existem amostras repetidas no conjunto, é imprescindível que seja possível obter de $\vec{w}_k$ um valor específico de ruído para cada partícula. Caso contrário, em poucas iterações todas as amostras seriam cópias da partícula com maior peso, fenômeno conhecido como *colapso*.

O filtro de partículas pode ser empregado de diversas maneiras como por

exemplo, determinar qual é a posição de um objeto em uma imagem, ou as regiões que possuem maior aglomeração dos pixels referentes à cor da pele. A utilização do filtro varia conforme o problema. Rekleitis[34] e Cuevas[3] descrevem três formas básicas de extrair informações sobre o problema em questão a cada iteração do filtro:

- Média Ponderada: $x_{est} = \sum_{j=1}^{M} \vec{w}_j \vec{x}_j$
- Melhor Partícula: $x_{est} = maxWeight(S^k)$
- Média Robusta: consiste em aplicar a Média Ponderada levando em consideração apenas as partículas que estão próximas da partícula com maior peso.

Cada método tem suas vantagens e desvantagens que devem ser consideradas de acordo com o problema que se deseja resolver. O filtro de partículas tem sido emprego com bastante sucesso, não só na área de Visão Computacional como também na robótica e economia sendo uma alternativa interessante para sistemas dinâmicos não-lineares e não-Gaussianos. No entanto, quando se aumenta a dimensão do sistema, é necessário aumentar a quantidade de amostras para que elas possam representar melhor a distribuição, o que acaba encarecendo o processo[12]. De forma geral, são empregadas de cem a mil partículas dependendo do problema e também existem casos de sucesso na utilização do filtro de partículas, com algumas otimizações, para rastreamento em tempo real[14].

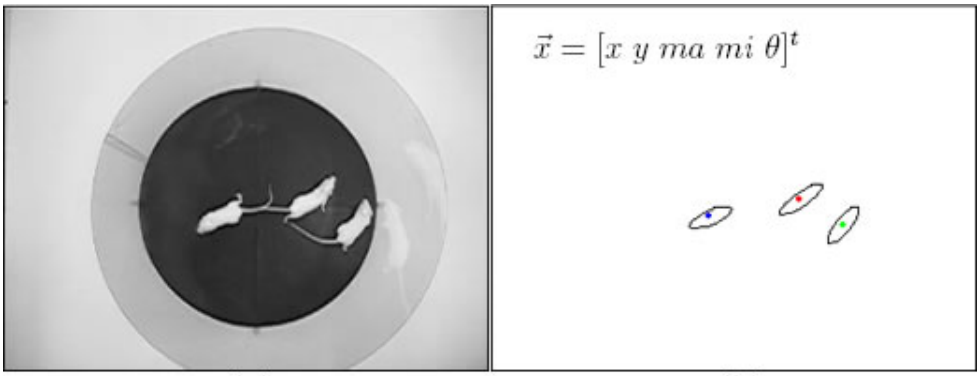

Figura 4.3: Modelo de estado de um camundongo, com as coordenadas x e y do centro de massa, eixo maior (ma) e menor (mi) da elipse e ângulo de inclinação θ.

Um exemplo prático da aplicação do filtro de partículas no rastreamento de três camundongos será apresentado a seguir. Considere o vetor de estados

e o modelo de observação discutidos na seção 2.2.1 e ilustrado novamente na Figura 4.3. Foram empregados três filtros de partículas, uma para cada camundongo, para efetuar o rastreamento durante o experimento Campo Aberto. A Figura 4.4 demonstra as etapas da primeira iteração dos filtros de partículas utilizados.

Inicialmente, os filtros devem receber uma imagem pré-processada na qual se encontram apenas os objetos de interesse, como ilustra a Figura 4.4(a). Logo em seguida, as partículas de cada filtro são inicializadas, como pode ser visto na Figura 4.4(b). Neste exemplo, as partículas foram inicalizadas com base em uma observação prévia com a adição de um ruído Gaussiano aleatório, com média 0 e desvio padrão 1. A próxima etapa consiste em realizar a predição do estado do sistema com base em um modelo de dinâmica, que foi implementado com base no movimento Browniano. A Figura 4.4(c) mostra esta etapa. Nota-se que as partículas sofreram um deslocamento aleatório, característico do movimento Browniano.

Na próxima etapa, o modelo de observação extrai informações relacionadas com o estado do sistema, como pode ser visto na Figura 4.4(d). Em seguida, a etapa de atualização de pesos combina as informações da observação com a predição realizada, atribuindo pesos maiores para as partículas que estão mais próximas da observação, como mostra a Figura 4.4(e). Na última etapa, chamada de correção ou reamostragem, as partículas são mapeadas em um intervalo entre 0 e 1 de acordo com o seu peso. Então, sorteia-se um número aleatório e a partícula que corresponde ao intervalo sorteado é selecionada para continuar no conjunto de amostras. Ao final desta etapa, todas as partículas voltam a possuir o mesmo peso, ou seja, todas são igualmente importantes para a distribuição, como pode ser visualizado na Figura 4.4(f).

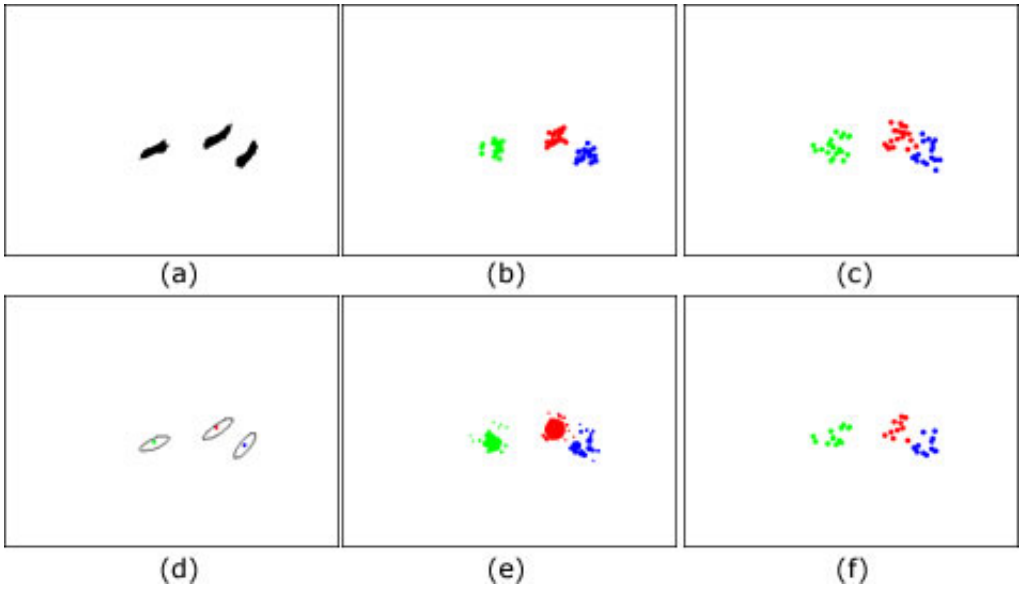

Figura 4.4: (a) Imagem segmentada restanto apenas os objetos de interesse. (b) Inicialização das Partículas. (c) Predição. (d) Observação, medida atual do estado do sistema. (e) Atualização de pesos. (f) Correção

Capítulo 5

Implementação

Ao longo deste trabalho foram implementadas duas técnicas de rastreamento de múltiplos objetos, a primeira baseada apenas em segmentação por limiarização e diferença entre imagens que foi implementada através de um plugin, o Topolino Tracker, para o ImageJ. Já a segunda técnica, uma framework baseada no filtro de partículas, foi implementada para resolver os problemas de rastreamento detectados pelo módulo Topolino Tracker e também foi escrita para ser facilmente integrada à plataforma SIGUS[28]. O módulo implementado ilustra a dificuldade e problemas existentes no rastreamento de múltiplos objetos. Apesar de utilizar técnicas simples, serve como base para a comparação com técnicas mais sofisticadas como o filtro de partículas.

A implementação do módulo para rastreamento de múltiplos camundongos foi realizada utilizando a linguagem Java a fim de utilizar diversos recursos proporcionados pela ferramenta ImageJ, também escrita em Java. O ImageJ é um software de domínio público, com código-fonte aberto, para o processamento e análise de imagens. O software permite, de maneira simples, o acréscimo de novos módulos, também escritos em Java, para expandir ainda mais suas funcionalidades.

Diversos módulos adicionais estão disponíveis no sítio[1] do software e em particular os módulos ParticleAnalyzer (neste caso, o substantivo *"Particle"* não está relacionado com o filtro de partículas) e MultiTracker que foram utilizados durante o desenvolvimento do módulo. O módulo ParticleAnalyzer é um identificador de regiões, bastante simples, que percorre uma imagem limiarizada até encontrar um pixel que corresponde ao plano de fundo (o ImageJ considera que pixels com valor 255 correspondem ao plano de fundo). Em seguida, o plugin busca, a partir do ponto encontrado, um pixel que esteja sob a borda do objeto. Depois, a borda é seguida até o ponto inicial, resultando

[1] http://rsb.info.nih.gov/ij/

em uma região. Já o MultiTracker utiliza os centros de massa das regiões identificadas em uma imagem, para buscá-los nas imagens subseqüentes da seqüência, de acordo com um deslocamento máximo v. As implementações são discutidas nas seções a seguir e os resultados obtidos por cada um deles serão apresentados no próximo capítulo.

5.1 Módulo Topolino Tracker

A implementação do módulo *plugin* Topolino Tracker para o ImageJ, baseado em técnicas simples, teve um papel importante na compreensão do problema de rastreamento de múltiplos objetos, além de trazer à tona problemas, como oclusão e ruído, que deveriam ser solucionados para que o rastreamento fosse realizado de forma eficiente.

Na etapa de segmentação, o módulo Topolino Tracker, subtrai cada imagem da seqüência de uma imagem de referência, obtendo o plano de interesse com os três camundongos que devem ser rastreados durante a seqüência de imagens. Após a subtração de imagens, aplica-se o ParticleAnalyzer e as regiões referentes aos camundongos são obtidas. No entanto, durante este processo, podem ocorrer problemas de identificação como ilustra a Figura 5.1.

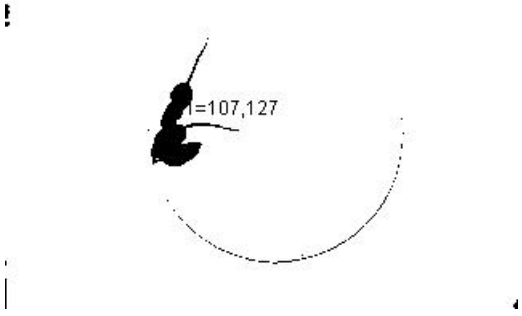

Figura 5.1: Identificação errada dos camundongos causada por oclusão.

Neste caso, o módulo ParticleAnalyzer responsável por identificar as regiões da imagem, acaba identificando apenas uma região ao invés de três, uma vez que os camundongos estão muito próximos. Uma alternativa seria aplicar a operação morfológica de erosão até que se obtivesse novamente três regiões, respectivas a cada camundongo. Em contrapartida, isso causaria sérios danos na forma dos camundongos o que influenciaria no cálculo do

seu centro de massa além de ser um procedimento que pode consumir algum tempo a mais de processamento.

Outro problema detectado durante a utilização do Topolino Tracker, foi a identificação errada do camundongo quando existe um ruído intenso na imagem, como mostra a Figura 5.2. A arena, onde é realizado o experimento Campo Aberto, possui paredes feitas de acrílico e que não são totalmente fixas. Portanto, sempre que os camundongos tocam as paredes, elas se deformam ou se movem mesmo que com pouca intensidade. Isso causa ruído pois, quando é realizada a subtração de imagens, as paredes da arena já não estão mais no local original. Ou seja, o plano de fundo sofreu uma pequena alteração e essa alteração cria compenentes a mais na imagem que não são os camundongos e sim, parte do plano de fundo.

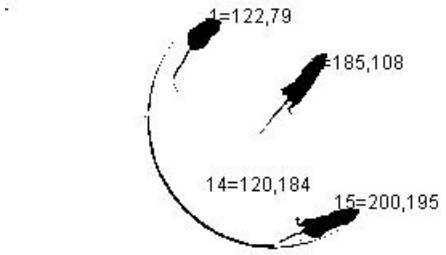

Figura 5.2: Identificação errada dos camundongos causada por ruído.

No entanto, quando os camundongos se encontram afastados é possível indentificá-los corretamente, como se pode ver na Figura 5.3. Logo após a identificação das regiões que contém os camundongos, o seu centro de massa (coordenadas X e Y) é obtido através de uma rotina do software ImageJ que o calcula utilizando momentos da imagem [36]. Para cada região encontrada na imagem atual, o algoritmo busca recursivamente nas imagens posteriores um centro de massa que esteja dentro de um limiar v de pixels. Este limiar serve para indicar qual é o deslocamento máximo que o camundongo pode realizar. Quando o novo centro de massa é encontrado, ele é acrescentado a uma lista de coordenadas que correspondem ao rastro daquele camundongo.

5.2 Particle Filter Tracker

Os resultados obtidos pelo módulo Topolino Tracker foram pouco precisos, portanto houve a necessidade da implementação de uma técnica que supe-

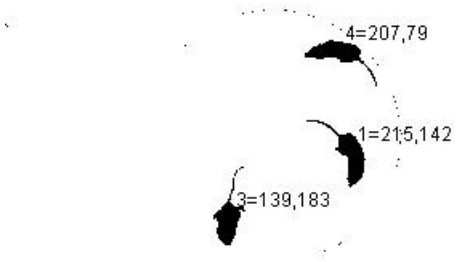

Figura 5.3: Identificação correta dos camundongos.

rasse os problemas de rastreamento que foram encontrados. Por isso, uma framework para rastreamento baseada no filtro de partículas foi desenvolvida que pode ser facilmente integrada à plataforma SIGUS. A framework é bastante genérica e permite que o filtro seja especializado para cada tipo de problema e a sua utilização não é restrita ao rastreamento de múltiplos objetos. Sendo possível criar, através da framework, rastreadores de um único objeto ou de vários objetos de tipos diferentes. A discussão teórica do filtro já foi realizada no Capítulo 4 e agora serão discutidos os detalhes de implementação.

A framework implementada visa facilitar a reutilização de código e a especialização de classes para cada problema. Para exemplicar uma possível utilização das classes desenvolvidas, considere um imagem na qual se deseja rastrear a face e as mãos de uma pessoa. Seriam necessários dois modelos, de dinâmica e de observação, e dois filtros de partícula, um para cada objeto para efetuar o rastreamento. O modelo de observação poderia aproximar a face a uma elipse e utilizar um modelo linear de movimento para identificar a face da pessoa no decorrer das imagens. As mãos poderiam ser modeladas a partir de modelos 3D ou ainda, modelos baseados em momentos da imagens com um modelo dinâmico mais sofisticado.

Apesar do filtro implementado ser genérico, aqui será detalhada a utilização da framework para o rastreamento de três camundongos durante o experimento Campo Aberto. Devido a dificuldade inerente a modelagem de um sistema complexo com três objetos que se movem e interajem entre si, optou-se por utilizar um filtro de partículas para cada camundongo, empregando o mesmo modelo de observação e de dinâmica pois os objetos são do mesmo tipo.

O modelo de observação é especificado na classe *ObservationModel* que

deve ser estendida e especializada de acordo com o problema em questão. O modelo aproxima a forma do camundongo a uma elipse utilizando a classe Measurements disponível no ImageJ e calcula o centro de massa do camundongo novamente através da ParticleAnalyzer, conforme ilustra a Figura 5.4. Esta classe não armazena valores, apenas fornece o estado do sistema na imagem i através do método *sense()*, que retorna um objeto do tipo *Particle*. No nível mais baixo na hierarquia da framework, a classe *Particle* corres-

Figura 5.4: Modelo de Obervação. A forma dos camundongos é aproximado à uma elipse e os pontos em vermelho, verde e azul correspondem ao centro de massa.

ponde a uma amostra (partícula) da distribuição do sistema que armazena as informações referentes aos possíveis estados do sistema como:

- Coordenada X do Centro de Massa do Camundongo;
- Coordenada Y do Centro de Massa do Camundongo;
- Circularidade da Elipse;
- Ângulo de inclinação da Elipse;

O ângulo de inclinação é o ângulo formado entre o eixo maior da elipse e o eixo x. A circularidade é a propriedade da Elipse que indica o quanto ela é alongada, assumindo valores entre 0 e 1. No caso da ocorrência de um círculo perfeito, o valor da circularidade é igual a 1. Esta medida é calculada da seguinte forma

$$c = 4\pi(area/perimetro^2) \tag{5.1}$$

A dinâmica do sistema é definida pela classe *MotionModel* da qual devem surgir especializações como *LinearMotionModel*, *AngularMotionModel*,

BrownianMotionModel e assim por diante. Inicialmente o modelo implementado adicionava uma velocidade v a cada componente do centro de massa do camundongo, acrescentando também um ruído Gaussiano com média zero e desvio padrão v. Ou seja, o modelo assumia que o camundongo se movia sempre na diagonal com alguma dose de ruído. Em seqüências com poucas imagens este modelo não apresentou problemas, no entanto, em seqüências de testes que são consideravelmente maiores, o modelo não representava o movimento real dos camundongos e dessa forma, as partículas se degeneravam rapidamente. A Figura 5.5 exibe iterações intermediárias do filtro, sendo possível verificar que grande parte das partículas utilizadas se tornam pouco significativas, mesmo após a reamostragem, pelo fato do modelo de dinâmica não ser adequado.

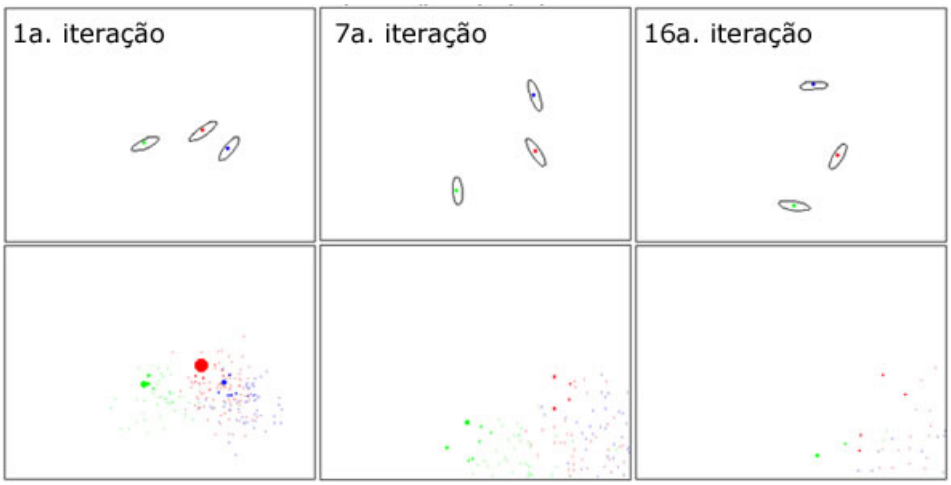

Figura 5.5: Degeneração das partículas após algumas iterações do filtro.

Por este motivo, houve a necessidade de implementar um novo modelo de dinâmica para o sistema. A classe *BrownianMotionModel* foi criada e representa o Movimento Browniano, também utilizado por Maybank [19] e Cuevas [3]. Este tipo de movimento é caracterizado por ser aleatório e está relacionado com o movimento de um objeto em suspensão (um grão de pólen na água, por exemplo) causado pelo choque das suas moléculas com aquelas que compõem o meio [4]. A Figura 5.6 mostra as mesmas iterações da imagem anterior mas dessa vez as partículas se comportam de acordo com o Movimento Browniano.

No topo da hierarquia, a classe *ParticleFilterTracker* aplica, de fato, o filtro de partículas portanto implementa os métodos para predição, atualização e reamostragem. Esta classe armazena referências para os modelos de observação e de dinâmica, e a coleção de partículas que representa a dis-

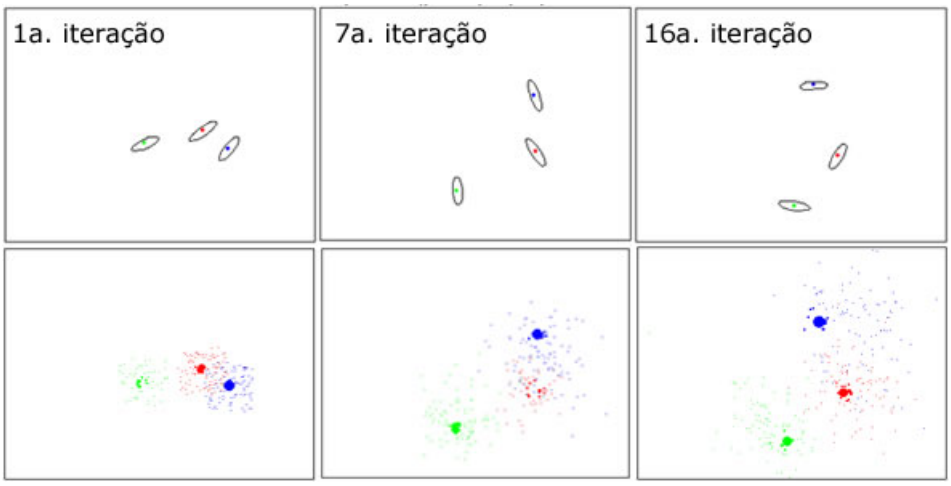

Figura 5.6: Conjunto de partículas com menor degeneração ao utilizar a dinâmica do Movimento Browniano.

tribuição do sistema. Métodos utilitários são fornecidos para inicializar as partículas aleatoriamente ou seguindo uma distribuição Gaussiana. A Figura 5.7 mostra a implementação realizada em linguagem Java do algoritmo apresentado no Capítulo 4, enquanto a Figura 5.8 representa o diagrama de classes proposto para a framework.

```
public Particle[] track(ImagePlus imp, ImageProcessor ip){
        predict();
        sense(imp,ip, index);
        update();
        normalize();
        if(ESS() < (betha * numParticles)){
            resample();
        }
        return samples;
}
```

Figura 5.7: Implementação do filtro de partículas

As classes descritas até o momento compõe o núcleo da framework. No entanto, resta definir como utilizar o filtro pois ele não faz nada além de estimar os estados do sistema. Agora é necessário realmente aproveitar as informações provenientes do filtro para realizar o rastreamento. Para isso, uma nova classe foi implementada, a *ParticleFilterController* que tem por objetivo controlar e utilizar as informações geradas pelos três filtros de partículas

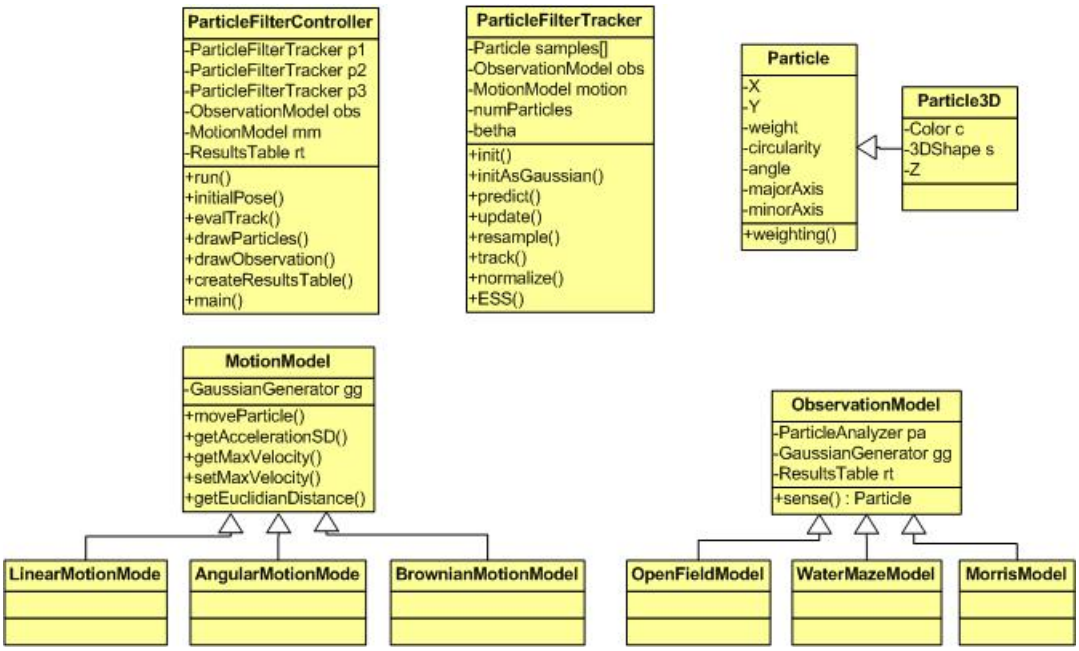

Figura 5.8: Diagrama de Classes UML para a Framework proposta.

empregados no problema em questão.

Conforme descrito na seção anterior, um dos maiores problemas a serem combatidos é a oclusão causada quando os camundongos estão muito próximos. É nesta situação que o filtro terá um papel importante. O filtro de partículas foi utilizado para ajudar o modelo de observação a identificar corretamente o centro de massa dos camundongos. Quando a observação assinala que foi encontrado um número menor de regiões do que o previsto, a partícula mais significativa de cada filtro será utilizada para tomar lugar das observações incorretas.

Capítulo 6

Experimentos e Resultados

O problema a ser resolvido é rastrear simultaneamente três camundongos durante um experimento denominado Campo Aberto. Este experimento, inicialmente descrito por Calvin S. Hall em 1930, avalia o comportamento locomotor e o estado emocional do animal [5]. O laboratório do Centro de Ciências Biológicas e da Saúde da Instituição, que realiza este experimento, utiliza como parâmetros para avaliação a distância percorrida (freqüência de exploração horizontal) e a quantidade de vezes que o camundongo eleva as patas dianteiras e o tronco (freqüência de exploração vertical). O escopo das implementações realizadas neste trabalho é extrair informações sobre a exploração horizontal de cada camundongo, enquanto outro trabalho de pesquisa, que está sendo realizado pelo Projeto Topolino, emprega técnicas de aprendizagem de máquina para avaliar a exploração vertical.

O experimento Campo Aberto é realizado em uma arena cilíndrica, de 40cm de diâmetro com paredes de acrílico translúcido, com 30cm de altura colocada sobre uma base de madeira recoberta com fórmica, a qual é subdividida em doze quadrantes de 104,7 cm^2 cada. A vista superior da arena descrita é mostrada na Figura 6.1a com a presença de dois camundongos e a vista lateral do experimento é ilustrada na Figura 6.1b.

Para a realização dos experimentos iniciais foram utilizadas três seqüências com 294 (A), 476 (B) e 549 (C) imagens respectivamente, capturadas de três vídeos distintos, do experimento Campo Aberto. Os vídeos foram gravados com resolução de 320 x 240 pixels, por uma câmera digital Canon Powershot A80. As imagens foram convertidas para tons de cinza e posteriormente foram organizadas em pilhas [1] representando cada seqüência, através do ImageJ. Todos os testes foram realizados em um computador pessoal, com processador AMD Athlon 64 3000+ (Clock de 1.8GHz), 1GB de memória RAM

[1]Recurso existente no ImageJ que permite agrupar várias imagens a fim de facilitar a manipulação das mesmas.

Figura 6.1: (a) Vista Superior. (b) Vista Lateral.

e sistema operacional Conectiva Linux 10.

Inicialmente, o plugin Topolino Tracker foi testado utilizando as seqüências de imagens descritas anteriormente, e os resultados preliminares ajudaram na compreensão das técnicas básicas de segmentação e também na identificação dos principais problemas de rastreamento, como oclusão e ruído. Assim que um camundongo é identificado pela primeira vez na imagem, um novo rastro é iniciado. O rastro representa a identificação, de um objeto através de seu centro de massa, e continua existindo enquanto aquele objeto continuar sendo identificado corretamente. Nas imagens seguintes quando um camundongos é encontrado, o rastreador verifica se ele é o mesmo que foi identificado no rastro da imagem anterior tendo como base uma velocidade máxima v. Caso o camundongo tenha se movido com uma velocidade superior a determinada, o módulo considera que existe um novo objeto, e inicia um novo rastro.

Utilizando a primeira seqüência com 294 imagens, o módulo detectou 40 rastros distintos. Já utilizando as 476 imagens da segunda seqüência, foram identificados 31 rastros diferentes e com a última e maior seqüência, com 549 imagens, foram encontrados 21. O resultado ideal seria que apenas 3 rastros fossem encontrados, que é a quantidade de camundongos presentes nas imagens.

Essas discrepâncias ocorrem devido à dificuldade em determinar a continuidade do movimento realizado pelo camundongo utilizando técnicas simples. Quando o animal faz mudanças abruptas na direção ou quando ele interage com outro camundongo, o módulo acaba criando erroneamente um novo rastro, acreditando ser um novo animal.

A Tabela 6.1 apresenta a distância percorrida por cada objeto detectado e a quantidade de quadros nos quais ele foi identificado na seqüência C. Nos casos em que os camundongos encontram-se separados, é possível identificá-los e rastreá-los corretamente como mostra a Figura 6.2.

Alguns rastreamentos realizados de forma incorreta são ilustrados na Fi-

Rastro	Distância Percorrida	Total de Quadros
1	44.809505	475
2	39.336815	15
3	3.669784	5
4	5.78424	3
5	9.065568	133
6	69.717445	8
7	1.4967653	10
8	25.716734	16
9	55.113926	59
10	21.926954	8
11	40.773617	55
12	57.768803	30
13	120.29482	268
14	22.200516	3
15	3.7178595	5
16	62.261097	68
17	19.607477	8
18	16.4123	36
19	12.875804	7
20	1.6525261	2
21	78.96609	61

Tabela 6.1: Melhor resultado obtido pelo Topolino Tracker utilizando a seqüência de imagens C. Distância Percorrida em pixels

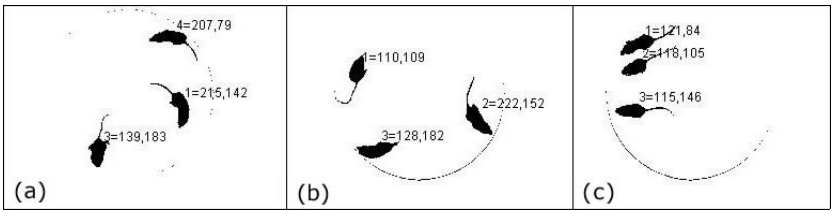

Figura 6.2: Exemplos de rastreamentos corretos.

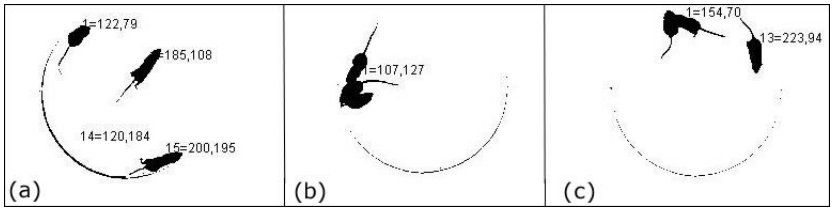

Figura 6.3: Exemplos de rastreamentos incorretos.

gura 6.3 na qual percebe-se a influência de ruído causando a identificação errada de um objeto (a), a detecção incorreta de apenas 1 objeto quando os três camundongos encontra-se muito próximos (b) e a não identificação de um camundongo (c).

Após estes resultados preliminares ficou evidente que o módulo Topolino Tracker apresenta sérios problemas que prejudicam o rastreamento. Ficou claro também que seria necessário eliminar, de alguma forma, o ruído que aparece após a subtração de imagens. A fim de resolver esse problema, foi feita na fase de segmentação a operação morfológica de fechamento. A fase de pré-processamento é ilustrada na Figura 6.4.

A primeira etapa da segmentação consiste em, para cada imagem da seqüência (Figura 6.4(a)), aplicar a limiarização (Figura 6.4(b)). Já com as imagens binarizadas, faz-se a subtração do plano de fundo com base em uma imagem de referência (Figura 6.4(d)) e inverte-se o valor dos pixels da imagem resultante (Figura 6.4(e)). Como comentado anteriormente, essas operações não eliminam o ruído nas imagens que pode causar identificação incorreta dos camundongos. Para eliminar ruídos e separar os camundongos quando eles não se encontram tão próximos, aplica-se a operação morfológica de fechamento (Figura 6.4(f)).

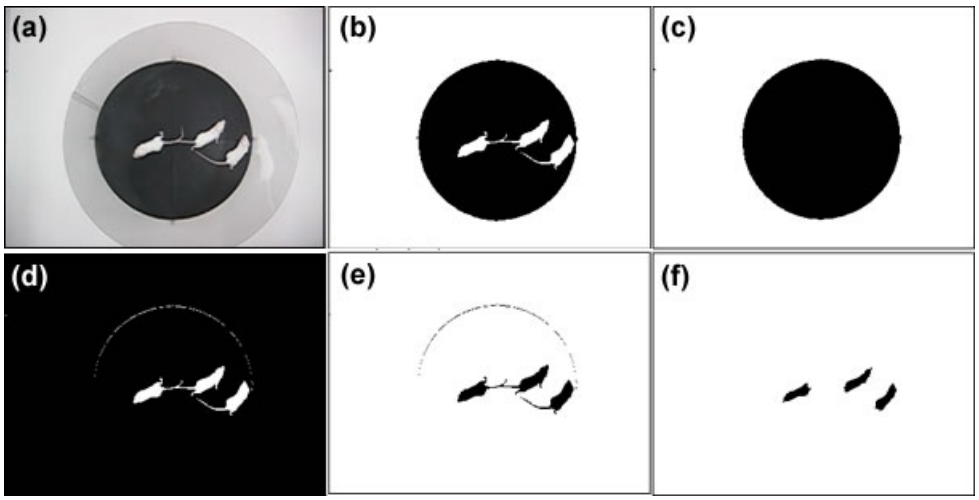

Figura 6.4: (a) Imagem capturada pela câmera. (b) Imagem Limiarizada. (c) Imagem de Referência Limiarizada. (d) Resultado da subtração da imagens. (e) Inversão dos pixels. (f) Resultado da operação de fechamento.

Após esta alteração na etapa de segmentação, um novo conjunto de seqüências de imagens foi preparado. Foram criadas 9 pilhas, cada uma com 50 imagens, numeradas de 1 a 9, contendo apenas situações de interesse

para o rastreamento. Ou seja, as imagens escolhidas para formar as pilhas são de situações nas quais aparecem ruído ou aquelas que mostram algum tipo de interação entre os camundongos.

Com as novas imagens de teste, o módulo Topolino Tracker obteve resultados melhores em relação às seqüências de imagens anteriores. O melhor resultado havia encontrado 21 "camundongos" (rastros) diferentes e na seqüência 5 foram encontrados 6 rastros como mostra a Tabela 6.2. Já o pior resultado foi de 17 rastros encontrados na seqüência 3.

Rastro	Distância Percorrida	Total de Quadros
1	56.75358	9
2	21.406218	6
3	47.399788	28
4	38.709373	42
5	30.702118	41
6	57.928196	20

Tabela 6.2: Melhor Resultado obtido pelo Topolino Tracker utilizando a seqüência de imagens 5. Distância Percorrida em pixels.

Como grande parte do ruído foi eliminado, o módulo Topolino Tracker apresentou melhoras em relação aos resultados anteriores. No entanto, ainda existe identificação incorreta dos camundongos quando eles permanecem muito próximos por algum tempo. Já utilizando o filtro de partículas, é possível identificar os três camundongos em todas as nove seqüências de teste, utilizando 100, 200, 500 ou 1000 partículas. Para facilitar a visualização dos resultados, serão apresentados apenas aqueles realizados com um conjunto de 100 partículas. A Tabela 6.3 mostra o rastreamento correto dos camundongos na seqüência 5.

Rastro	Distância Percorrida	Total de Quadros
1	2033.754	50
2	1727.746	50
3	1655.741	50

Tabela 6.3: Resultado obtido pelo filtro de partículas utilizando a seqüência de imagens 5. Distância Percorrida em pixels.

A Figura 6.5 mostra situações nas quais o filtro de partículas obteve sucesso na identificação e rastreamento dos camungondos, inclusive quando eles estão muito próximo. Na primeira coluna da imagem, à esquerda, tem-se as imagens provenientas da seqüência, já segmentadas. Na segunda coluna

está a visualização gráfica dos dados obtidos pelo modelo de observação e na última coluna, o conjunto de partículas de cada filtro após a atualização de pesos.

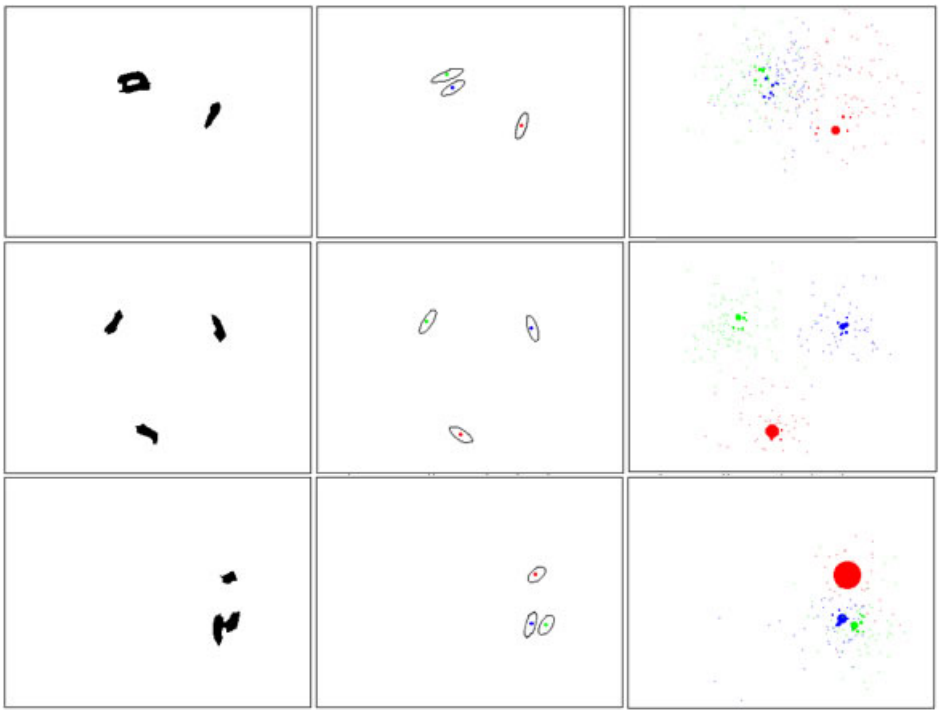

Figura 6.5: À esquerda, imagens segmentadas. Ao centro, visualização gráfica da etapa observação. À direita, conjunto de partículas de cada filtro após a etapa de atualização de pesos.

Comparando apenas a quantidade de camundongos identificados corretamente, pode-se dizer que o filtro de partículas obteve um resultado superior ao módulo Topolino Tracker, o que já era esperado. No entanto, é necessário verificar quão certo está o rastreamento realizado pelo filtro de partículos. Para isso, um banco de imagens precisa ser preparado de modo que, os camundongos sejam marcados manualmente para que seja possível calcular o seu centro de massa. Este sem dúvida é um trabalho que demanda muito tempo, desde a escolha das imagens até a marcação manual dos pixels que correspondem aos camundongos, mas é de grande importância para a avaliação da técnica implementada.

Apesar disso, não foi possível criar tal banco de imagens. Contudo, três seqüências de imagens (1, 5 e 9) foram escolhidas e a partir delas, um observador humano marcou manualmente, de forma empírica, um ponto que correspondesse, aproximadamente, ao centro de massa daquele camundongo.

Esta etapa foi realizada utilizando o software ImageJ. O intuito dessa solução paliativa para os testes é prover uma análise preliminar dos resultados obtidos pelo filtro de partículas. A Tabela 6.4 mostra a diferença média entre o centro de massa marcado e o obtido pelo filtro, assim como o desvio padrão. Vale ressaltar que área aproximada de um camundongo é de 100 pixels quando ele está em pé e 250 pixels quando não está.

	Camundongo 1	Camundongo 2	Camundongo 3
Média	36,13668716	30,5824887	24,87906147
Desvio Padrão	28,84885971	23,28319519	17,22420969

Tabela 6.4: Diferenças entre a marcação manual do centro de massa dos camundongos e o resultado obtido pelo filtro de partículas, utilizando um conjunto de 100 amostras, para a seqüência 5. Média e Desvio Padrão em pixels.

Apesar do filtro implementado sempre encontrar os três camundongos, algumas vezes o modelo de observação associa erradamente o centro de massa a um determinado camundongo. Em outras palavras, em determinadas situações o centro de massa do camundongo 1, é associado ao camundongo 2, por exemplo. Isto ocorre, geralmente, quando os camundongos permanecem juntos por muito tempo e trocam de posições no final da interação, fazendo com que o modelo de observação avalie de maneira errada a correspondência entre os camundongos e seus respectivos centros de massa, como pode ser visualizado na Figura 6.6.

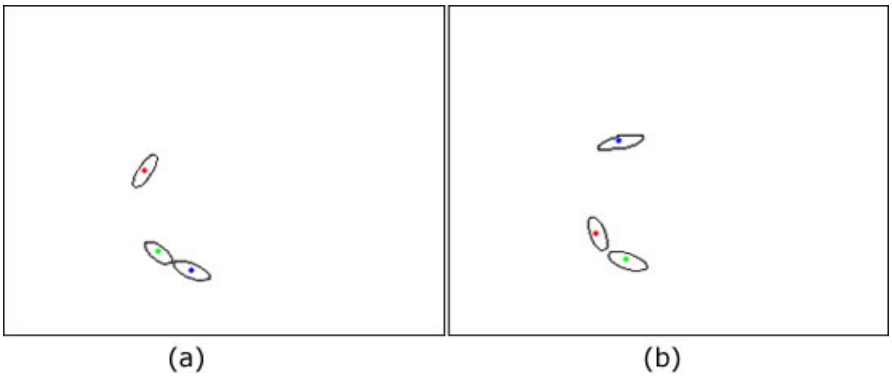

Figura 6.6: a) Observação realizada no quadro 10. b) Observação realizada no quadro 11.

Capítulo 7

Considerações Finais

A utilização de técnicas de Visão Computacional é uma alternativa interessante para o rastreamento de animais em experimentos de laboratório. Estes experimentos são etapas importantes que devem ser realizadas durante o desenvolvimento e testes de novos fármacos. De forma geral, um observador humano é o responsável por anotar, manualmente ou de forma semi-automática, o comportamento apresentado pelo animal. Esta abordagem pode sofrer com a fadiga do observador e ainda, podem haver discrepâncias entre a avaliação de um mesmo comportamento por um grupo distinto de observadores.

Neste contexto, o projeto Topolino propõe o desenvolvimento de um sistema automático de análise do comportamento animal em experimentos de laboratório, sendo uma das frentes de pesquisa a utilização de técnicas de Visão Computacional para o rastreamento de múltiplos objetos. Neste trabalho, foi realizado um levantamento bibliográfico sobre as principais técnicas utilizadas para esse tipo de rastreamento, assim como foi desenvolvida uma framework para rastreamento baseada no filtro de partículas. Durante a análise preliminar, foi também implementado um módulo para o ImageJ para o rastreamento de múltiplos objetos utilizando técnicas simples de segmentação que não apresentou bons resultados.

O rastreamento de múltiplos objetos é um problema complexo que exige técnicas mais sofisticadas para que possa ser resolvido de forma eficiente. Na literatura, muitos autores têm utilizado o filtro de partículas com sucesso para lidar com esse problema. Ao utilizar amostras para representar a distribuição de um sistema e assumir que os modelos de dinâmica e de observação podem ser não-lineares, o filtro de partículas se torna bastante poderoso e flexível, sendo empregado para estimar o estado ótimo do sistema.

Os resultados obtidos pelo filtro de partículas durante o rastreamento de múltiplos objetos foram superiores aos obtidos utilizando técnicas simples. No entanto, a eficiência do filtro de partículas está diretamente relacionada

com a eficiência dos modelos utilizados e ainda, o custo computacional aumenta a medida que mais partículas são exigidas para representar a distribuição do sistema.

Apesar dos resultados animadores, a framework desenvolvida ainda possui alguns pontos que precisam ser melhorados. A modelagem matemática mais aprofundada do rastreamento de três camundongos durante o experimento Campo Aberto, pode proporcionar melhores resultados e resolver o problema com o modelo de observação citado no capítulo anterior. Adicionalmente, o modelo de dinâmica pode ser adaptativo, ou seja, pode ter seus parâmetros ajustados a medida que o rastreamento é realizado. Por exemplo, a cada três imagens a direção do movimento e a velocidade são atualizadas com base nas informações obtidas sobre a posição do camundongo naquelas imagens.

A criação, com metodologias adequadas, de um banco de imagens com marcações manuais dos camundongos é de grande importância para que, no futuro, possam ser feitas comparações mais criteriosas entre técnicas de rastreamento. A utilização de outros filtros preditivos, como o filtro de Kalman, devem ser considerados para efeito de comparação.

Espera-se que a framework de rastreamento seja integrada à plataforma SIGUS e que suas classes sejam estendidas para utilização de modelos, de dinâmica e observação, específicos para cada problema. Todo o material produzido ao longo deste trabalho, como códigos-fonte e imagens, podem ser obtidos no sítio do projeto Topolino (http://www.gpec.ucdb.br/topolino).

Referências Bibliográficas

[1] J. Bilmes. A gentle tutorial on the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models, 1997.

[2] J. Carpenter, P. Clifford, e P. Fernhead. An improved particle filter for non-linear problems, 1997.

[3] E. Cuevas, D. Zaldivar, e R. Rojas. Particle filter in vision tracking. Relatório técnico, Freie Universität Berlin - Department of Mathematics and Computer Science, 2005.

[4] Laboratório de Sistemas Neurais. Descrição microscópica da difusão. Relatório técnico, Universidade de São Paulo - Ribeirão Preto.

[5] D. Eilam. Open-field behavior withstands drastic changes in arena size. *Behavioural Brain Research*, 142:53–62, 2003.

[6] D. A. Forsyth e J. Ponce. *Computer Vision: a modern approach*. Prentice Hall, 2003.

[7] S. K. Goldenstein. A gentle introduction to predictive filters. *Revista de Informatica Teórica e Aplicada (RITA)*, 11:61–89, 2004.

[8] R. C. Gonzalez e R. E. Woods. *Processamento de Imagens Digitais*. Edgard Blücher, 1992.

[9] A. Harvey. *Forecasting, Structure Time Series Models and the Kalman Filter*. Cambridge University Press, 1989.

[10] L. Hermes, T. Zöller, e J. M. Buhmann. Parametric distributional clustering for image segmentation. *European Conference on Computer Vision*, 3:577–591, 2002.

[11] C. Hue, J. P. Le Cadre, e P. Pérez. Tracking multiple objects with particle filtering. *IEEE Transactions on Aerospace and Eletronic Systems*, 38:791–812, 2000.

[12] M. Isard e A. Blake. CONDENSATION - conditional density propagation for visual tracking. *Int. J. Computer Vision*, 1:5–28, 1998.

[13] Z. Kahn, T. Balch, e F. Dellaert. Efficient particle filter-based tracking of multiple interacting targets using an MRF-based motion model. *Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1:254– 259, 2002.

[14] C. Kwok, D. Fox, e M. Meil. Real-time particle filters. *verificar*, 2:627–630, 2002.

[15] M. Lewerenz. Monte carlo methods: Overview and basics. *Quantum Simulations of Complex Many-Body Systems: From Theory to Algorithms, Lecture Notes.*, 10:1–24, 2002.

[16] J. S. Liu e R. Chen. Sequential Monte Carlo methods for dynamic systems. *Journal of the American Statistical Association*, 93(443):1032–1044, 1998.

[17] J. Maccormick e A. Blake. A probabilistic exclusion principle for tracking multiple objects. *International Journal of Computer Vision*, 39:57–71, 2000.

[18] S. Maskell, M. Rollason, D. Salmond, e N. Gordon. Efficient particle filtering for multiple target tracking with application to tracking in structured images, 2002.

[19] Stephen J. Maybank, Anthony D. Worall, e Geoffrey D. Sullivan. A filter for visual tracking based on a stochastic model for driver behaviour. In *ECCV (2)*, páginas 540–549. 1996.

[20] R. A. F. Mini e M. F. M. Campos. Visual tracking of objects using multiresolution. *XII Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens*, páginas 153–160, 1999.

[21] K. P. N. Murthy. Monte carlo: Basics. Relatório técnico, Theoretical Studies Section, Materials Science Division, Indira Gandhi Centre for Atomic Research, Kalpakkam 603 102, Tamil Nadu, INDIA, 2001.

[22] H. T. Nguyen e M. Worring. Multifeature object tracking using a model-free approach. *IEEE Conference on Computer Vision and Pattern Recognition*, 1:145–150, 2000.

[23] H. T. Nguyen, M. Worring, R. V. D. Boomgaard, e A. W. M. Smeulders. Tracking nonparameterized object contours in video. *IEEE Transaction on Image Processing*, 11:1081–1091, 2002.

[24] L. P. J. J Noldus, A. J. Spink, e R. A. J. Tegelenbosch. Ethovision: A versatile video tracking system for automation of behavioral experiments. *Behavior Research Methods, Instruments and Computers*, páginas 398–414, 2001.

[25] K. Nummiaro, E. Koller-Meier, e L. J. Van Gool. Object tracking with an adaptive color-based particle filter. *Symposium for Pattern Recognition of the DAGM*, páginas 353–360, 2002.

[26] A. Pavlov e M. G. S. Bueno. Filtro de partículas Ótimo para rastreamento de alvos balísticos. *XX Simpósio Brasileiro de Telecomunicações*, 2003.

[27] P. Perner. Motion tracking of animals for behavior analysis. *International Workshop on Visual Form*, páginas 779–786, 2001.

[28] H. Pistor, J. J. Neto, A. A. C. Junior, M. C. Pereira, e T. R. V. Santos. Sigus - plataforma de apoio ao desenvolvimento de sistemas para inclusão digital de pessoas com necessidades especiais. 2004.

[29] P. Pérez, C. Hue, J. Vermaak, e M. Gangnet. Color-based probalistic tracking. *European Conference on Computer Vision*, páginas 661–675, 2002.

[30] Y. Raja, J. McKenna, e S. Gong. Segmentation and tracking using colour mixture models. *Asian Conference on Computer Vision*, 1:607–614, 1998.

[31] J. M. Rehg e T. Kanade. Model-based tracking of self-occluding articulated objetcs. *Proccedings of the 5th International Conference on Computer Vision*, páginas 612–617, 1995.

[32] D. B. Reid. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, 6, 1979.

[33] I. M. Rekleitis. *Cooperative Localization and Multi-Robot Exploration*. Tese de Doutoramento, School of Computer Science, McGill University, Montreal, Quebec, Canada, February 2003. Http://www.cim.mcgill.ca/~yiannis/Publications/thesis.pdf.

[34] I. M. Rekleitis. A particle filter tutorial for mobile robot localization. Relatório Técnico TR-CIM-04-02, Centre for Intelligent Machines, McGill University, 3480 University St., Montreal, Québec, CANADA H3A 2A7, 2004.

[35] C. Ridder, O. Munkelt, e H. Kirchner. Adaptative background estimation and foreground detection using kalman-filtering. *International Conference on recent Advances in Mechatronics*, páginas 193–199, 1995.

[36] K. P. Souza e H. Pistori. Implementação de um extrator de características baseado em momentos da imagem. *XVIII Brazilian Symposium on Computer Graphics and Image Processing - SIBGRAPI, III Workshop de Trabalhos de Iniciação Científica em Computação Gráfica e Processamento de Imagens*, 2005.

[37] M. Spengler e B. Schiele. Multi-object tracking: Explicit knowledge representation and implementation for complexity reduction. *Cognitive Vision Workshop*, 2002.

[38] A.J. Spink, R.A.J. Tegelenbosch, M.O.S. Buma, e L.P.J.J. Noldus. The ethovision video tracking system: A tool for behavioral phenotyping of transgenic mice. *Physiology and Behavior*, 73:731–744, 2001.

[39] D. Tweed e A. Calway. Tracking many objects using subordinated condensation. *British Machine Vision Association 2002*, páginas 283–292, 2002.

[40] D. Tweed e A. Calway. Tracking multiple animals in wildlife footage. *16th International Conference on Pattern Recognition*, 2:24–27, 2002.

[41] C. J. Twining, C. J. Taylor, e P. Courtney. Robust tracking and posture description for laboratory rodents using active shape models. *Behavior Research Methods, Instruments & Computers*, 33:381–391, 2001.

[42] T. Zhao. *Model-based Segmentation and Tracking of Multiple Humans in Complex Situations*. Tese de Doutoramento, University of Southern California, 2003.

[43] T. Zhao e R. Nevatia. Tracking multiple humans in crowded environment. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 9:406–413, 2004.