UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
CENTRO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Multi-resolução com Fóvea Móvel para Redução e Abstração de Dados em Tempo Real

Rafael Beserra Gomes

Orientador: Prof. Dr. Luiz Marcos Garcia Gonçalves

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Engenharia Elétrica e de Computação da UFRN (área de concentração: Engenharia de Computação) como parte dos requisitos para obtenção do título de Mestre em Ciências.

Número de ordem PPgEE: M240
Natal, RN, agosto de 2009

Divisão de Serviços Técnicos

Catalogação da publicação na fonte. UFRN / Biblioteca Central Zila Mamede

Gomes, Rafael Beserra.

Multi-resolução com Fóvea Móvel para Redução e Abstração de Dados em Tempo Real / Rafael Beserra Gomes - Natal, RN, 2009

23 p.

Orientador: Luiz Marcos Garcia Gonçalves

Dissertação (mestrado) - Universidade Federal do Rio Grande do Norte. Centro de Tecnologia. Programa de Pós-Graduação em Engenharia Elétrica.

1. Visão computacional - Dissertação. 2. Processamento de imagens - Dissertação. I. Gonçalves, Luiz Marcos Garcia. II. Universidade Federal do Rio Grande do Norte. III. Título.

RN/UF/BCZM CDU 004.932(043.2)

Multi-resolução com Fóvea Móvel para Redução e Abstração de Dados em Tempo Real

Rafael Beserra Gomes

Dissertação de Mestrado aprovada em 07 de agosto de 2009 pela banca examinadora composta pelos seguintes membros:

Prof. Dr. Luiz Marcos Garcia Gonçalves (orientador) . . . . . UFRN

Prof. Dr. Herman Martins Gomes . . . . . UFCG

Prof. Dr. Luiz Eduardo Cunha Leite . . . . . UFRN

Prof. Dr. Bruno Motta de Carvalho . . . . . UFRN

Dedico este trabalho aos meus pais
que sempre me apoiaram e me
forneceram toda a educação
necessária.

Agradecimentos

Primeiramente, agradeço a Deus por toda a força e inteligência concedida, além de ter guiado os meus passos. Segundo, este trabalho não teria sido feito sem a ajuda de diversas pessoas com quem convivo. Agradeço à minha família que sempre me apoiou. Agradeço a todos os professores, em especial àqueles que me ensinaram e orientaram sobre visão computacional e processamento de imagens: Luiz Marcos e Bruno Motta. Há também aqueles que contribuíram de forma indireta. Durante o mestrado, muitos foram os amigos que compartilharam seus conhecimentos, que de certa forma, fizeram parte deste trabalho e merecem meus agradecimentos.

Resumo

Nós propomos uma nova abordagem para reduzir e abstrair informações visuais para aplicações de visão robótica. Basicamente, usamos uma representação em multi-resolução em combinação com uma fóvea móvel para reduzir a quantidade de informações de uma imagem. Apresentamos a formalização matemática do modelo em conjunto com funções de mapeamento que auxiliam na utilização do modelo. Propomos dois índices (resolução e custo) que visam auxiliar na escolha das variáveis do modelo proposto. Com essa nova abordagem teórica, é possível aplicar diversos filtros, calcular disparidade estéreo e obter análise de movimento em tempo real (menos de 33ms para processar um par de imagens em um notebook AMD Turion Dual Core 2GHz). Como principal resultado, na maior parte do tempo, a fóvea móvel permite ao robô não realizar movimentação física de seus dispositivos robóticos para manter uma possível região de interesse visível nas duas imagens. Validamos o modelo proposto com resultados experimentais.

Palavras-chave: Visão Robótica, Fóvea Móvel, Multi-resolução.

Abstract

We propose a new approach to reduction and abstraction of visual information for robotics vision applications. Basically, we propose to use a multi-resolution representation in combination with a moving fovea for reducing the amount of information from an image. We introduce the mathematical formalization of the moving fovea approach and mapping functions that help to use this model. Two indexes (resolution and cost) are proposed that can be useful to choose the proposed model variables. With this new theoretical approach, it is possible to apply several filters, to calculate disparity and to obtain motion analysis in real time (less than 33ms to process an image pair at a notebook AMD Turion Dual Core 2GHz). As the main result, most of time, the moving fovea allows the robot not to perform physical motion of its robotics devices to keep a possible region of interest visible in both images. We validate the proposed model with experimental results.

Keywords: Robotics Vision, Moving Fovea, Multi-resolution.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Propomos uma metodologia para acelerar a visão de baixo nível (*low-level vision*), ou seja a etapa dos algoritmos de visão computacional que geralmente envolve o pré-processamento de imagens para fornecer dados mais adequados às etapas posteriores. O método consiste em abstrair e reduzir a quantidade de dados providos pelas câmeras utilizando um novo modelo de multi-resolução em combinação com uma fóvea móvel, também nossa proposta, de forma a ser possível obter informações visuais em tempo real (por exemplo, menos de 33ms para processar um par de imagens em um notebook AMD Turion Dual Core 2GHz). O modelo mostra-se adequado para aplicações de visão robótica, que, além de exigirem processamento em tempo real, requerem movimentação de recursos robóticos.

Neste trabalho, introduzimos todo o formalismo matemático deste modelo em conjunto com as operações (algoritmos) que permitem a utilização do mesmo. Pensamos num modelo simples de tal forma que ele possa ser implementado facilmente via software, sem a necessidade de hardware dedicado para seu funcionamento em tempo real. Assim, alguns algoritmos clássicos de visão computacional podem ser facilmente adaptados para o novo modelo, uma vez que ele mantém as mesmas características gerais do domínio matricial no qual esses algoritmos operam. As funções de mapeamento que fazem parte do modelo, permitem elaborar algoritmos que atuam no espaço escala, do nível de menor resolução para o de maior resolução ou vice-versa. Como conseqüência da formalização, analisamos a escolha ideal dos parâmetros de acordo com uma métrica definida.

Embora haja redução de informações em regiões periféricas, o método é capaz de obter ganho de informações do ambiente ao permitir a movimentação livre do centro da fóvea. No caso, este novo modelo de fóvea móvel permite minimizar a quantidade desses movimentos, ainda provendo processamento em tempo real. O controle da fóvea pode ser integrado facilmente com modelos de atenção visual, como o mapa de saliências. O movimento físico dos dispositivos pode então ser realizado somente quando a região de interesse estiver prestes a sair do campo de visão das câmeras, ao contrário de outras metodologias que tratam a fóvea como uma posição fixa, geralmente no centro da imagem.

No topo do modelo proposto, tarefas de alto nível, como estratégias de atenção visual e reconhecimento de objetos, podem ser elaboradas. O intuito é fornecer um modelo

que apóie a construção de um sistema robótico autônomo capaz de realizar tarefas em um ambiente dinâmico. Com o objetivo de validar o modelo proposto, implementamos um sistema seguindo este modelo, dando suporte a tarefas de reconhecimento e atenção. Além disso, o modelo pode ser útil em outras aplicações como detecção de objetos e transmissão de vídeo. Para esta última aplicação, testes foram realizados para verificar a utilidade do modelo em transmissão de vídeo onde a banda de transmissão é restrita.

Assim, este trabalho tem como objetivo principal aprimorar o modelo de multi-resolução permitindo que a fóvea possa se mover pela imagem. Além disso, modelos propostos anteriormente [Segundo & Gonçalves 2004, Boyling & Siebert 2000, Boyling 2004] carecem de generalização. O número de níveis e os tamanhos das imagens e dos níveis são constantes e o método de redimensionamento é fixo, embora isso não fosse necessário. Os modelos anteriores carecem também de formalização matemática (os modelos são propostos de forma algorítmica textual). Neste trabalho, formalizamos o modelo de multi-resolução com fóvea móvel e fornecemos operações (funções de mapeamento entre níveis e imagem) que permitem maior utilidade e controle sobre o modelo. A formalização do modelo também permite abrir questões sobre a escolha ideal das variáveis no modelo.

Ao contrário de soluções no espaço variante, o modelo proposto, por ter uma estrutura de armazenamento invariante no espaço, é de fácil indexação, permitindo que algoritmos tradicionais em visão computacional e processamento de imagens sejam facilmente aplicados aos dados, com pouquíssima adaptação. Além disso, o esquema proposto não exige recursos de hardwares dedicados ou mesmo sensores específicos. Convém ressaltar que realizamos testes com *web* câmeras que tornaram possível a realização de tarefas pelo robô a uma taxa moderada, e são bastante acessíveis, no contexto desta metodologia proposta. Outra vantagem do modelo proposto é a sua facilidade de implementação. Claro, se desejarmos um processamento mais rápido, no intuito de atingir taxas de 30 quadros por segundo ou mais, câmeras profissionais devem ser usadas, com hardware de captura específico, mas ainda podendo ser usado um computador convencional, sendo ainda muito mais aquém do que se usarmos computadores e hardware específico para o processamento de imagem.

Capítulo 2

Trabalhos Relacionados

Prover um sistema visual artificial a um robô é um grande passo para aplicações de robótica autônoma, onde a idéia é justamente construir robôs que possam reagir sem intervenção humana em resposta aos estímulos promovidos pelo ambiente onde se encontram. Porém, o processamento de imagens em tempo real ainda hoje é um dos grandes desafios em visão artificial aplicada à robótica. Os dados visuais adquiridos via dispositivos de captura de vídeo podem ser rapidamente requisitados em tempo real, enquanto o processamento sobre os mesmos é um fator limitante para a constituição de um sistema em tempo real, a menos que se usem arquiteturas específicas, que são geralmente caríssimas. O problema torna-se ainda mais crítico quando se deseja extrair diversas informações visuais para atividades de propósitos gerais de um robô (ou características). A quantidade de informações visuais necessárias pode crescer muito rapidamente à medida que mais tarefas são passíveis de serem executadas. Por exemplo, se a percepção de profundidade do ambiente é necessária, o sistema visual deve efetuar, geralmente, muitas operações de convolução visando a determinação de disparidades sobre duas imagens [Marr 1982, Horn 1986], que é uma das maneiras usuais de se computar um mapa de profundidade.

2.1 Visão ativa

Uma abordagem possível para a construção de um sistema visual artificial é realizar a análise completa e indexação da imagem, a partir da qual um sistema de decisão computacional possa atuar. Entretanto, a grande demanda de processamento computacional para tal fim sugere outras abordagens de menor custo computacional motivadas biologicamente. Há vários aspectos biológicos que podem ajudar na construção de um sistema visual computacional, entre os quais pode-se destacar que um animal não necessita indexar tudo o que estiver no seu campo de visão, mas somente o que é relevante para a tarefa [Churchland & Sejnowski 1992]. Nesse sentido, alguns modelos foram propostos na literatura que transformam uma imagem em uma representação não uniforme baseando-se na fisiologia do olho humano.

De acordo com Tortora [Tortora 2000], a visão é o sentido que responde a estímulos de luz. Para os animais, o sentido da visão permite o reconhecimento do ambiente e o sistema visual é um dos principais sensores para interação do ser com o ambiente. Ainda

de acordo com Tortora [Tortora 2000], a parte de tecido nervoso da retina, onde a imagem se forma, possui três camadas de neurônios. Uma delas possui células foto-receptoras de dois tipos: os bastonetes e os cones. Os primeiros são especializados para a visão em luz fraca, com pouca intensidade. Já os cones são especializados para a visão da cor e para a acuidade visual. Os cones estão mais densamente concentrados na parte central da retina, denominada fóvea central, que é a região de visão mais acurada. A Figura 2.1 ilustra a anatomia do olho humano.

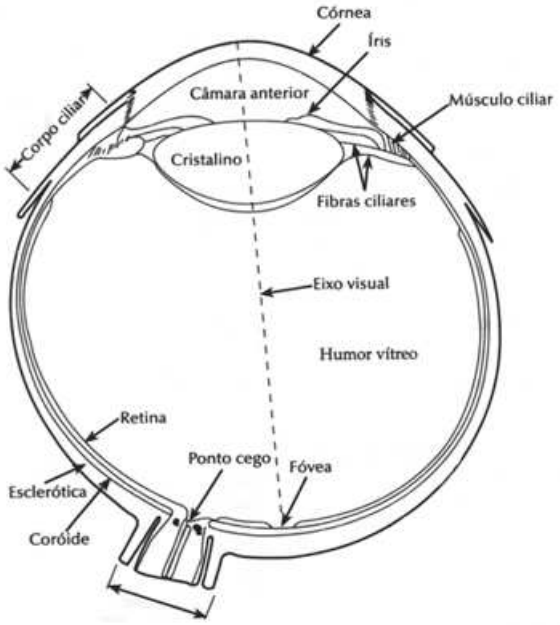

Figura 2.1: Representação esquemática do olho humano.

Existem várias abordagens para novas representações de imagens na literatura. Uma das abordagens mais tradicionais é usar a representação de imagens em multi-resolução [Burt & Adelson 1983]. Isto é, em um espaço de escala usando estruturas piramidais ou wavelets [Chang & Yap 1997] ou em um espaço variante [Bernardino & Santos-Victor 2002]. O crédito pela idéia de se utilizar o modelo clássico em multi-resolução pode ser dado à Leonard Uhr [Uhr 1972]. O modelo de espaço de escalas é formalizado por Witkin [Witkin 1983] e a idéia da pirâmide Laplaciana foi introduzida por Burt e Adelson [Burt & Adelson 1983]. Outras abordagens são inspiradas na visão com fóvea. As metodologias de espaço variante [Schwartz et al. 1995] utilizam um sistema onde os sensores visuais possuem distribuição não uniforme, geralmente distribuídos de forma a concentrar mais sensores na região da fóvea. O resultado de uma amostragem em um conjunto de sensores em um espaço variante pode ser visto na Figura 2.2. A transformação log-polar reproduz a imagem em um novo espaço que tenta reproduzir o mapeamento da retina para o cortex visual através de uma representação log-polar, o que resulta em uma representação mais compacta da imagem [Gomes & Fisher 2001, **?**].

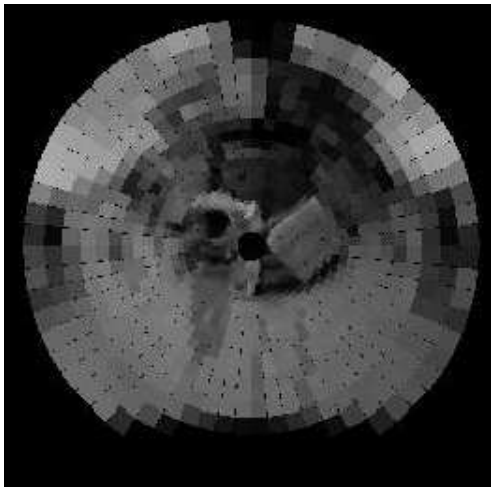

Figura 2.2: Conjunto de sensores em espaço variante.

Alguns desses modelos de redução de dados permitem uma livre movimentação da fóvea e não possuem aplicação somente em visão robótica, mas também em transmissão em tempo real e codificação de vídeos [Chang & Yap 1997, Wu & Rao 2005, Kortum & Geisler 1996]. Nesse âmbito de aplicação, deseja-se que clientes requisitem imagens de um servidor, podendo essas imagens ter altas resoluções. A comunicação entre cliente e servidor pode ser limitada, por exemplo, por uma conexão de internet [Chang & Yap 1997]. A posição da fóvea nas imagens também pode ser sincronizada com o olho humano [Kortum & Geisler 1996]. Neste caso, um sistema com câmera mantém informações sobre a posição atual dos olhos do usuário e altera a posição da fóvea da imagem para o ponto para o qual o usuário está olhando. Com isso, o usuário pode perceber a imagem como se não tivesse ocorrido perda de resolução na periferia. Quando se trata de transmissão de vídeo, é factível esperar até alguns segundos de atraso pelo primeiro dado, desde que os dados seguintes, a partir de então, cheguem à taxa supostamente agendada, digamos a 30 quadros por segundo num sistema de TV Digital. Já em outras aplicações, isso pode gerar um problema se o sistema tiver que esperar um determinado tempo antes de poder tomar alguma decisão, o que pode ser crítico, por exemplo, se trabalhamos com sistemas de tempo real com restrição de atraso (aplicações de robótica são exemplos de tais sistemas).

2.2 Modelos de multi-resolução

Vários modelos de multi-resolução são propostos na literatura. Aqui descrevemos com maiores detalhes alguns desses modelos e principalmente o modelo proposto por Gonçalves e colaboradores [Gonçalves et al. 2000, Segundo & Gonçalves 2004] o qual serviu como base para o modelo proposto aqui.

2.2.1 Pirâmide de multi-resolução

No modelo de pirâmide de multi-resolução tradicional, uma imagem é re-amostrada em níveis diferentes de resolução, a partir da aplicação de algum filtro, que pode ser, por exemplo, o filtro média ou o filtro Gaussiano, formando uma estrutura similar a uma pirâmide (de imagens). A base da pirâmide é uma imagem com a mesma resolução e tamanho que a imagem original e as imagens seguintes sofrem uma redução de tamanho, mas mantendo a mesma área de abrangência da imagem original (toda a imagem é re-amostrada). Consequentemente, as imagens menores possuem resoluções menores, pois cobrem a mesma área que a base da pirâmide, mas com uma quantidade menor de pixels. Não abordaremos a sua construção aqui por estar bem definida na literatura a respeito [Uhr 1972, Witkin 1983, Burt & Adelson 1983, Schwartz et al. 1995]. A imagem da Lena aplicada no modelo de pirâmide de multi-resolução pode ser vista na Figura 2.3.

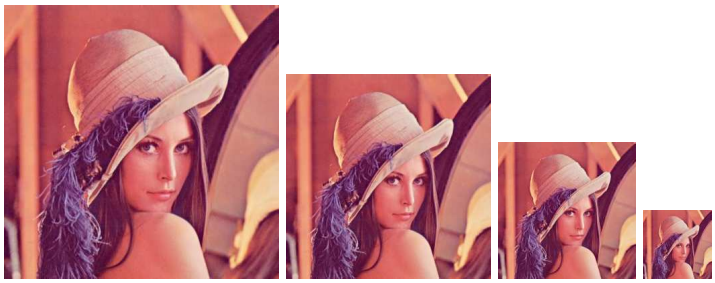

Figura 2.3: Pirâmide de multi-resolução com 4 níveis.

2.2.2 Wavelets

Wavelets são funções que permitem transformar um sinal de forma a ser possível analisar seu domínio da frequência assim como o seu domínio espacial em escalas diferentes. Essa transformação se dá pela transformada wavelet. Em um primeiro passo, um filtro passa alta e um passa baixa são convoluídos com as linhas da imagem e as colunas ímpares são eliminadas. Em seguida, um filtro passa alta e um passa baixa são convoluídos com cada imagem resultante (duas imagens com metade do comprimento) e as linhas ímpares são eliminadas, resultando em 4 imagens com metade do tamanho original. Esse processo de decomposição utilizando a transformada wavelet pode ser visualizado na Figura 2.4. A imagem resultante das duas filtragens passa baixa contém uma versão reduzida da imagem original na qual também pode ser aplicada uma nova transformada e continuamos com esse procedimento recursivamente até um determinado nível desejado. Um exemplo na qual uma transformada wavelet foi aplicada utilizando somente 1 passo pode ser visto na Figura 2.5. Utilizando a transformada wavelet inversa é possível reconstruir perfeitamente a imagem original integrando os 4 componentes que compõem um nível. Entretanto, essa perfeição na prática não é atingida dada a limitação numérica finita de um computador.

A transformada wavelet pode ser útil não só pela análise do espaço escala, mas também por extrair características da imagem, dependendo da função wavelet utilizada, como,

por exemplo, gradiente ou laplaciano. Entretanto, utilizar wavelets no contexto de visão robótica em tempo real é desnecessariamente custoso, pois a decimação de um nível é resolvida através de 4 convoluções, enquanto no modelo proposto, como será visto, a mesma análise do espaço escala pode ser realizada apenas através de redimensionamento de imagens.

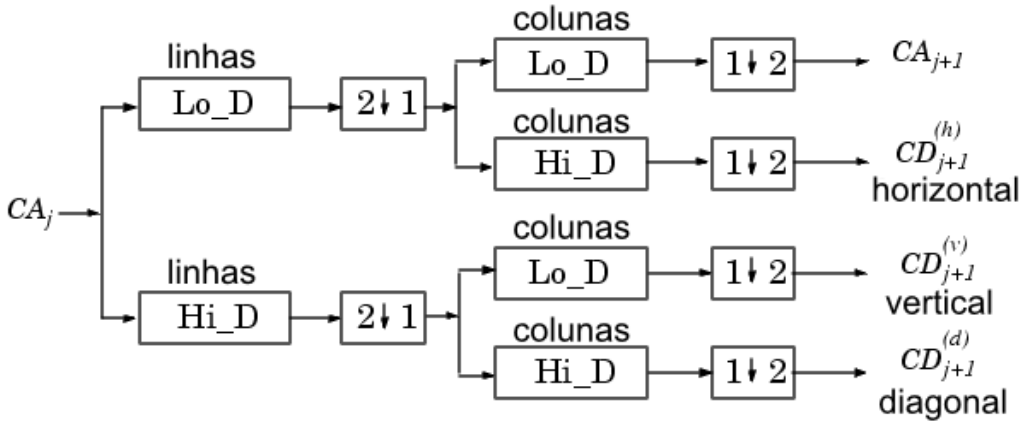

Figura 2.4: Transformada wavelet discreta bidimensional.

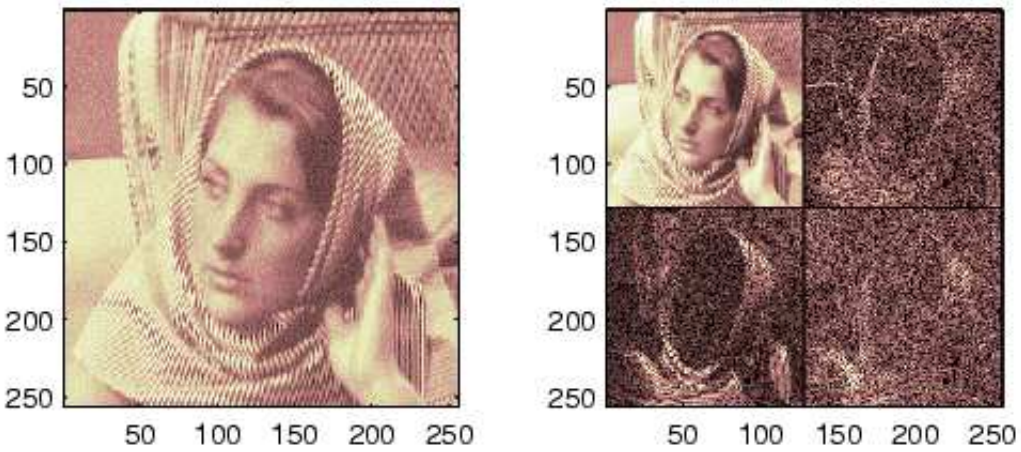

Figura 2.5: Exemplo de aplicação da transformada wavelet em uma imagem por 1 nível.

Conforme discutido anteriormente, alguns modelos de redução de dados permitem a livre movimentação da fóvea e possuem aplicação na transmissão e codificação de vídeos. Nesse sentido, a transformada wavelet pode ser aplicada para codificar uma imagem fazendo com que a resolução dependa da distância para uma ou mais posições de fóvea, reduzindo a quantidade de dados que precisam ser transmitidas [Chang & Yap 1997]. Essa redução pode ser realizada através do descarte de coeficientes, uma vez que para

determinada região tão poucos coeficientes são necessários transmitir, quanto o inverso da distância dessa região para a posição da fóvea. Em outras palavras, se uma região está distante da fóvea, esta região pode ser representada por menos coeficientes do que uma região que está mais perto. A função escala pode ser descrita por: $w(t) = \alpha \mid t - \gamma \mid + \beta$, onde α é a taxa de resolução, γ é o centro da fóvea e β é a resolução máxima [Chang & Yap 1997]. Um exemplo de aplicação desse método em uma imagem pode ser vista na Figura 2.6.

Figura 2.6: Exemplo de imagem integrando wavelets com fóvea.

2.2.3 Multi-resolução com fóvea centralizada

Em outro modelo de multi-resolução, sem denominação, abordado por Gonçalves e colaboradores [Gonçalves et al. 2000, Segundo & Gonçalves 2004] e Boyling [Boyling & Siebert 2000, Boyling 2004], assim como na pirâmide de multi-resolução, uma imagem é re-amostrada em níveis de diferentes resoluções. Entretanto, todos os níveis possuem o mesmo tamanho e cada qual é o redimensionamento de uma região centralizada na imagem. As regiões que mapeiam os níveis formam uma pirâmide (vide Figura 2.7), de forma que o primeiro nível é resultado do redimensionamento da imagem inteira (região da base da pirâmide) e o último nível é resultado do redimensionamento de uma região de tamanho equivalente ao tamanho do nível (topo da pirâmide), isto é, simplesmente uma cópia pixel a pixel. Este modelo simula uma visão com fóvea, no sentido que o modelo abstrai diferentes regiões da imagem com diferentes resoluções.

Um exemplo da estrutura pode ser visto na Figura 2.8 [Segundo & Gonçalves 2004]. As imagens multi-resolução são as imagens inferiores, que foram re-amostradas e re-escaladas a partir da imagem superior (em tamanho original). No exemplo mostrado na Figura 2.8, são usadas 5 imagens de tamanho 32×32 pixels cada uma, totalizando 5.120 pixels para serem manipulados por outros processos de mais alto nível. A imagem original possui tamanho 512×512 totalizando 262.144 pixels, provendo então uma redução drástica na quantidade de dados a serem processados. Note que imagens com maior resolução incluem mais detalhes dos objetos da cena, porém cobrindo uma região menor da cena. No limite, a resolução é a mesma da imagem original, mas apenas uma pequena porção da

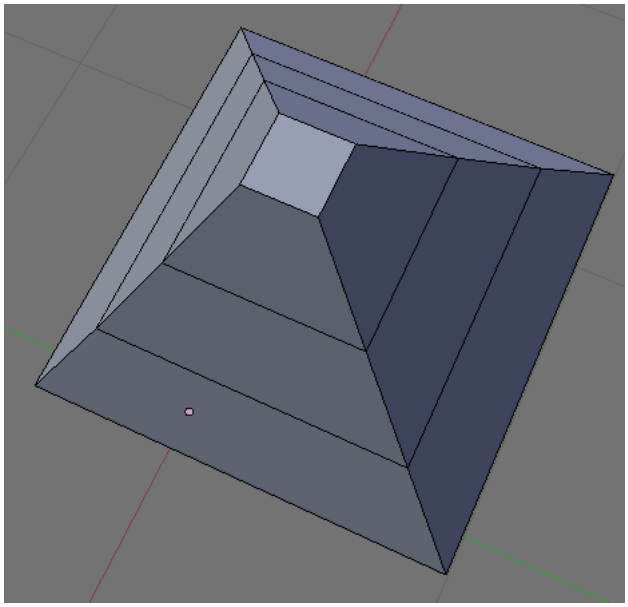

Figura 2.7: Regiões de mapeamento formam uma pirâmide (4 níveis).

cena original é re-amostrada, resultando numa espécie de fóvea artificial. Do outro lado, imagens com menor resolução incluem menos detalhes dos objetos, mas cobrem regiões maiores da cena, o que seria equivalente a regiões da periferia do campo visual, onde detalhes não são importantes. Ou seja, as imagens nesta estrutura são re-amostragens da imagem original cobrindo áreas maiores desta à medida que diminui a resolução, por isso são todas do mesmo tamanho. A redução drástica nos tamanhos das imagens em multi-resolução permite a implementação em tempo-real de tarefas de atenção e reconhecimento [Segundo & Gonçalves 2004] e também neste trabalho, como será visto. O resultado da aplicação do modelo nas imagens mostradas na Figura 2.9 pode ser visto na Figura 2.10.

2.3 Fóvea Móvel

Visando contornar os problemas expostos acima, dos modelos baseados em multi-resolução tradicionais [Gonçalves et al. 2000, Segundo & Gonçalves 2004, Boyling & Siebert 2000, Boyling 2004], no presente trabalho, propomos um modelo em representação de dados usando uma estrutura em multi-resolução, visando diminuir drasticamente a quantidade de dados, combinado com um esquema de fóvea móvel (em software) que permite reduzir a quantidade de movimentos desnecessários de um robô. Neste modelo, denominado por multi-resolução com fóvea móvel, a posição da fóvea é dada por uma coordenada que assume uma posição no plano de projeção onde se forma a imagem, permitindo obter regiões mais ou menos acuradas em qualquer parte do campo de visão.

Fazendo uma analogia, a possibilidade de mover a fóvea corresponde à capacidade de prestar atenção a um objeto que se move no campo visual sem mover os olhos, mas com a mesma percepção que se os estivéssemos movendo. Este processo é conhecido como

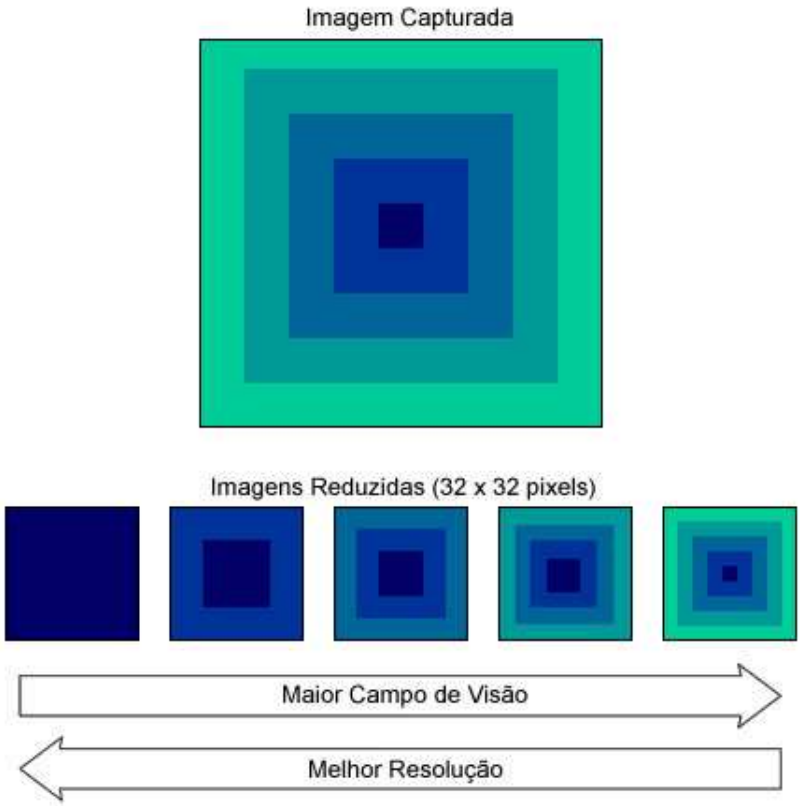

Figura 2.8: Exemplo de imagens em multi-resolução.

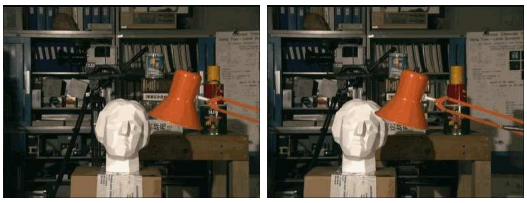

Figura 2.9: Imagens de Tsukuba utilizadas no modelo de multi-resolução.

Figura 2.10: Exemplo de imagens em multi-resolução.

atenção encoberta (ou *covert attention*), ou seja, olhamos para uma direção enquanto nosso foco de atenção está em outra direção. Infelizmente, nós (humanos) não temos grande resolução na percepção de detalhes (cores) em regiões fora da fóvea, assim não conseguimos perceber o objeto com a mesma quantidade de detalhes que se ele estivesse na fóvea. Porém, numa implementação em um robô provido de um sistema de visão por computador, é possível que se tenha a mesma resolução que na fóvea, estando o objeto em qualquer posição dentro do campo de visualização. Note que se o objeto sair do campo de visão, um movimento físico do olho (vergência), da cabeça (*pan/tilt*), ou mesmo do seu corpo, se faz necessário para que o objeto volte a estar presente no campo de visão. Similarmente, para o robô, o movimento físico dos dispositivos é então somente necessário quando a região de interesse estiver prestes a sair do campo de visão de uma ou de ambas as câmeras.

Capítulo 3

O Modelo de Fóvea Móvel

Uma das motivações do uso da fóvea móvel é que este recurso permita o desenvolvimento de sistemas de visão ativa, que implica na implementação de processos visuais reativos e de comportamentos visuais elementares em tempo real [Marr 1993, Batista et al. 1997]. O resultado da estrutura em multi-resolução abordada por Gonçalves e colaboradores [Segundo & Gonçalves 2004] aplicada em uma imagem na qual uma bola aparece no centro da imagem pode ser vista na Figura 3.1. Nesta Figura, as imagens foram superpostas e re-escaladas visando mostrar a região que cobrem na imagem original, notando que a resolução é ruim na periferia. A fóvea ou região central da imagem possui mais detalhes.

Caso a bola se mova para a região periférica, usando o modelo proposto por Gonçalves e colaboradores [Segundo & Gonçalves 2004], não temos mais como obter resolução máxima na região para a qual a bola se moveu. Obter mais detalhe em algum local fora do centro da imagem, nesse modelo, seria possível de duas maneiras. Na primeira delas, pode-se mover os dispositivos de aquisição de imagens visando adquirir uma nova imagem, com a região de interesse (por exemplo, a bola) no centro da imagem, e então uma nova estrutura em multi-resolução pode ser construída. Na segunda forma, a idéia é simplesmente recalcular somente a estrutura em multi-resolução passando como parâmetro a nova posição para a fóvea, desde que esta outra posição esteja dentro de um determinado limite na imagem original, quer dizer, não muito na periferia dela. Note que isso evitaria mover as câmeras para tomar novas imagens, o que é mais caro num robô, no sentido de que levaria bem mais tempo para se ter a nova região no centro da imagem, do que simplesmente re-calculando de novo a estrutura em multi-resolução. Relembrando que as regiões de mapeamento do modelo proposto por Gonçalves e colaboradores [Segundo & Gonçalves 2004] podem ser vistas como uma pirâmide (vide Figura 2.7), o modelo proposto no presente trabalho permite que a mesma pirâmide possa se distorcer, como ilustra a Figura 3.3. O resultado de distorcer a pirâmide, isto é, mover a fóvea, de forma a obter máxima resolução na região da bola, mesmo estando esta na região periférica da imagem, é ilustrado na Figura 3.2.

No caso, então, a idéia principal do uso de uma fóvea móvel é permitir que uma determinada região no modelo de multi-resolução usado neste trabalho possa ser vista com resolução maior, evitando movimentos desnecessários dos recursos robóticos (motores e câmeras). Claro que a partir de um determinado limite da posição da fóvea para a periferia da imagem, o movimento seria inevitável.

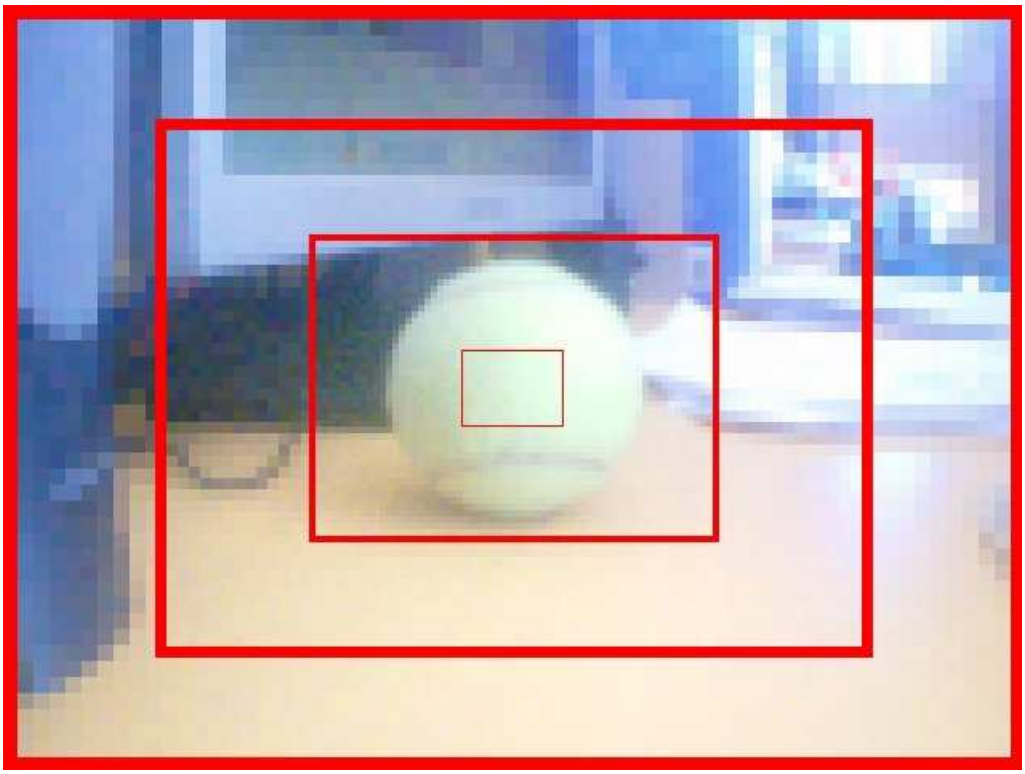

Figura 3.1: Reconstrução da imagem onde a fóvea está centralizada em uma bola.

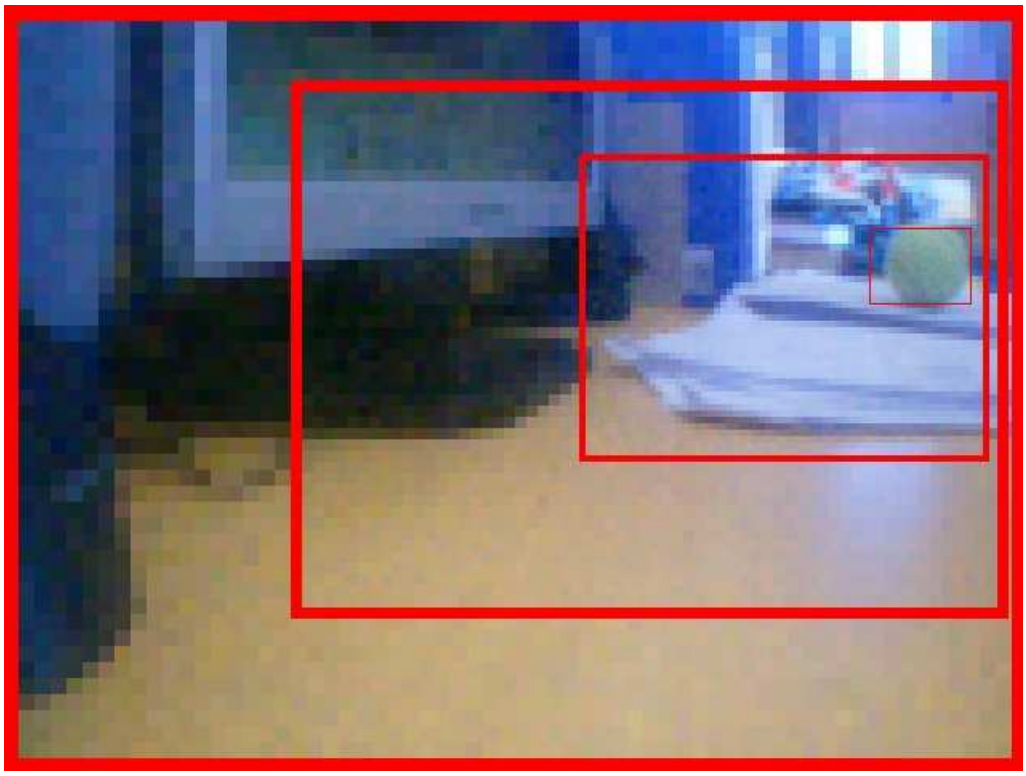

Figura 3.2: Reconstrução da imagem onde a fóvea está centralizada em uma bola.

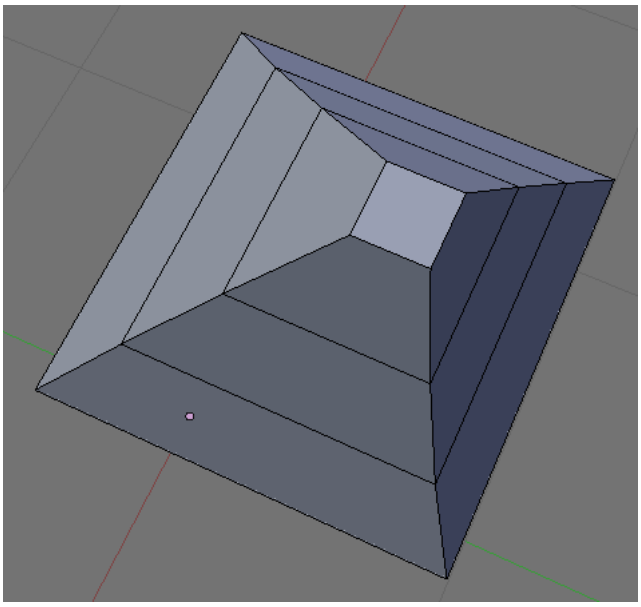

Figura 3.3: Regiões de mapeamento formam uma pirâmide distorcida pela posição da fóvea (4 níveis).

Assim, a nossa proposta principal é justamente formalizar matematicamente, desenvolver algoritmos e operações relacionadas num sistema e mostrar resultados de testes realizados usando o modelo de fóvea móvel. Para isto, alguns conceitos e pressupostos são introduzidos neste Capítulo.

Embora o modelo de multi-resolução seja, geralmente, definido para imagens digitais, optamos por introduzir alguns conceitos a partir do espaço contínuo (espaço $\mathbb{R}^2$), supondo que a imagem é uma função de domínio contínuo, a fim de postergar possíveis truncamentos forçados pelo espaço discreto. Neste trabalho, denominamos esse modelo no espaço contínuo como modelo multi-níveis. Este modelo servirá como base para a definição em seguida do modelo no espaço discreto.

3.1 O modelo multi-níveis no espaço contínuo

Essa formulação foi definida com a idéia que I represente uma imagem digital no espaço contínuo de tamanho representado por $U = (U_0, U_1)$, que deverá ser redimensionada em $m+1$ imagens (níveis), onde a k-ésima imagem é definida por R_k e possui tamanho fixo $W = (W_0, W_1)$. A Figura 3.4 ilustra cada uma das variáveis em um exemplo onde há 4 níveis. A função ϕ assume o papel de redimensionar a imagem I em cada um dos níveis. Cada nível k é o redimensionamento de uma região na imagem original denominada A_k, cujo tamanho é S_k. O conjunto de regiões A_k formam uma pirâmide, assim como ilustrada na Figura 3.3. O tamanho desta região para o nível de menor resolução, isto é, o nível 0, corresponde ao tamanho da imagem original, ou seja, $S_0 = U$. O tamanho dessa região para o nível de maior resolução, isto é, o nível m, corresponde ao tamanho do próprio nível, ou seja, $S_m = W$. Assim, para este último nível, teremos simplesmente a cópia da região A_m na imagem original para a imagem R_m. Convencionamos ao longo do texto que um índice i nos vetores do modelo indica a i-ésima componente deste vetor (por exemplo, U_i indica a i-ésima componente do vetor U).

O centro da região A_m referente ao nível de maior resolução R_m é posicionado por um vetor fóvea $\mathbf{F}$. Como R_0 mapeia toda a imagem, esse nível permanece com os mesmos valores de pixels, não importando onde a fóvea é posicionada. Para o nível R_m, o centro da fóvea pode mover-se de sua origem $(0,0)$ (o centro de I) até um limite no qual o mapeamento está junta à borda da imagem original. Assim, o centro da fóvea deve estar a uma distância de no mínimo $W/2$ de qualquer uma das bordas. Por definição, optamos fazer com que o vetor fóvea seja $(0,0)$ caso a fóvea esteja localizada no centro da imagem original, uma vez que, assim, torna-se fácil reformular as equações do modelo para o modelo sem fóvea móvel (fóvea no centro da imagem). Na definição a seguir o vetor fóvea é indicado por F, entretanto utilizamos F' para auxiliar na transformação do vetor fóvea. O centro da fóvea $\mathbf{F}$ deve estar em $(W-U)/2 \times (U-W)/2$.

Podemos compreender a estrutura descrita anteriormente como uma diversidade de níveis, tal que cada nível pode ser utilizado ou não, pode estar ativado ou desativado. O descarte de níveis na estrutura permite que se simule uma pirâmide de multi-resolução com relação não linear entre os níveis, embora cada nível tenha sido obtido através de uma interpolação linear. O modelo multi-níveis é então definido por:

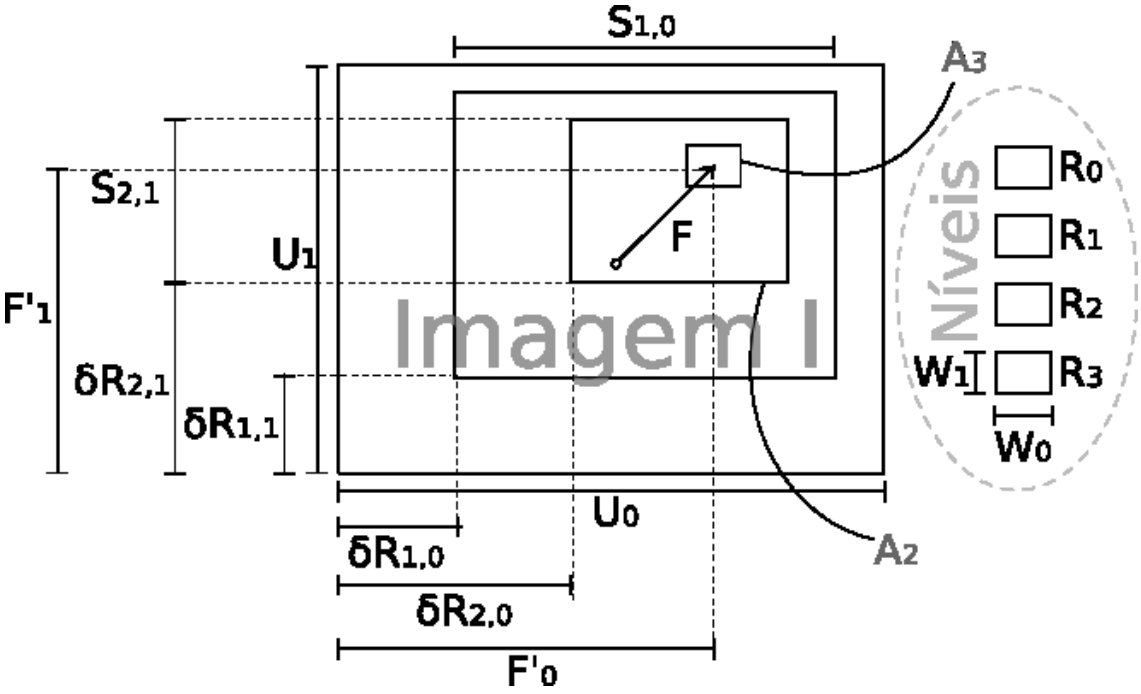

Figura 3.4: Reconstrução da imagem onde a fóvea está centralizada em uma bola.

Definição 3.1.1. *Definição do modelo multi-níveis*

O modelo multi-níveis é composto por uma estrutura definida pela tupla $(I, \boldsymbol{U}, \boldsymbol{W}, m, \boldsymbol{F}, B)$. *A função* $I : \mathbb{R}^2 \to \mathbb{R} \subset [0...U_0, 0...U_1]$, *onde* $\boldsymbol{U} = (U_0, U_1) \in \mathbb{R}^2$, *é redimensionada em níveis numerados de* 0 *a m (*$m+1$ *níveis). Cada k-ésimo nível* $R_k : \mathbb{R}^2 \to \mathbb{R} \subset [0...W_0, 0...W_1]$, *tal que* $\boldsymbol{W} = (W_0, W_1) \in \mathbb{R}^2$ *e* $W_i < U_i$, *é definido por uma função* $\phi : [0...W_0] \times [0...W_1] \subset \mathbb{R}^2 \to [0...U_0] \times [0...U_1] \subset \mathbb{R}^2$, *isto é,* $R_k(x,y) = I(\phi(x,y))$. ϕ *é tal que:* $\phi(0,0) = \delta R_k$, $\phi(\boldsymbol{W}) = \delta R_k + S_k$ *e cada componente dos demais pontos desta função são definidos como uma interpolação linear dos respectivos componentes entre* $\phi(0,0)$ *e* $\phi(\boldsymbol{W})$ *(vide Figura 3.5). A região* A_k *é definida pelo produto cartesiano* $[\delta R_{k,0}, \delta R_{k,0} + S_{k,0}] \times [\delta R_{k,1}, \delta R_{k,1} + S_{k,1}]$, *onde* $\boldsymbol{S}_k = (S_{k,0}, S_{k,1}) \in \mathbb{R}^2$ *define o tamanho dessa região. Por definição,* $\delta R_0 = (0,0)$, $S_0 = \boldsymbol{U}$ *e* $S_m = \boldsymbol{W}$. *Cada componente de* S_k *é definido como uma interpolação linear dos respectivos componentes entre* S_0 *e* S_m. *Cada componente de* δR_k *é definido como uma interpolação linear dos respectivos componentes entre* δR_0 *e* δR_m. A_k *deve variar de forma que* F' *seja o centróide de* A_m. F, *o vetor fóvea, é definido por* $F' - \frac{U}{2}$. B *é uma lista de* $m+1$ *elementos, tal que* $B_i = 0$ *indica que o i-ésimo nível é descartado. Se B for omitido, considera-se B tal que todos os seus elementos são igual a* 1.

Proposição 3.1.2. *O tamanho de cada região* A_k *é dado por:*

$$S_k = \frac{kW - kU + mU}{m} \tag{3.1}$$

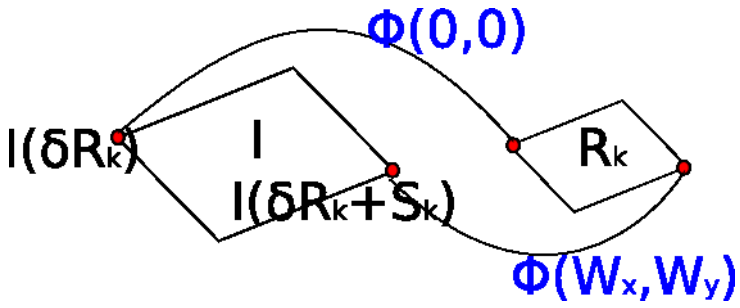

Figura 3.5: Restrições da função ϕ.

Demonstração. *Pela definição 3.1.1 temos que cada componente de S_k é uma interpolação linear dos respectivos componentes de S_0 e S_m. Ainda pela mesma definição temos que $S_0 = U$ e $S_m = W$. Logo, fazendo a interpolação linear para cada componente temos:*

$$\frac{S_{m,i} - S_{0,i}}{S_{k,i} - S_{0,i}} = \frac{m}{k} \tag{3.2}$$

$$k(S_{m,i} - S_{0,i}) = m(S_{k,i} - S_{0,i}) \tag{3.3}$$

$$k(W_i - U_i) = m(S_{k,i} - U_i) \tag{3.4}$$

$$kW_i - kU_i + mU_i = mS_{k,i} \tag{3.5}$$

$$S_{k,i} = \frac{kW_i - kU_i + mU_i}{m} \tag{3.6}$$

□

Proposição 3.1.3. *O deslocamento δR_k é dado por:*

$$\delta R_k = \frac{k(U - W + 2F)}{2m} \tag{3.7}$$

Demonstração. *Pela definição 3.1.1 temos que F' é o centróide de A_k, isto é, $\delta R_m + S_m/2 = F'$. Logo:*

$$2\delta R_m + S_m = 2F' \tag{3.8}$$

$$\delta R_m = \frac{2F' - W}{2} \tag{3.9}$$

Ainda pela mesma definição, cada componente de δR_k é uma interpolação linear entre os

respectivos componentes de δR_0 *e* δR_m*, lembrando que* $\delta R_0 = (0,0)$*. Portanto:*

$$\frac{\delta R_{m,i} - \delta R_{0,i}}{\delta R_{k,i} - \delta R_{0,i}} = \frac{m}{k} \tag{3.10}$$

$$\frac{2F_i' - W_i}{2\delta R_{k,i}} = \frac{m}{k} \tag{3.11}$$

$$k(2F_i' - W_i) = 2m\delta R_{k,i} \tag{3.12}$$

$$\delta R_{k,i} = \frac{k(2F_i' - W_i)}{2m} \tag{3.13}$$

$$\delta R_{k,i} = \frac{k(U_i - W_i + 2F_i)}{2m} \tag{3.14}$$

□

Observe que δR_k está determinado apenas para $m > 0$, isto é, a estrutura deve conter pelo menos dois níveis.

Proposição 3.1.4. *A função* ϕ *que, dado um ponto em um nível de resolução* R_k*, resulte no ponto correspondente em I, é dada pela definição 3.1.1:*

$$\phi_{k,i}(p) = \frac{kW_i(U_i - W_i) + 2kW_iF_i + 2p_i(mU_i - kU_i + kW_i)}{2mW_i} \tag{3.15}$$

Demonstração. *Pela definição 3.1.1 temos que:* $\phi(0,0) = \delta R_k$*,* $\phi(\boldsymbol{W}) = \delta R_k + S_k$ *e os componentes dos demais pontos são definidos como uma interpolação linear dos respectivos componentes entre esses dois valores. Logo:*

$$\frac{\phi_{k,i}(p) - \phi_{k,i}(0,0)}{\phi_{k,i}(W) - \phi_{k,i}(0,0)} = \frac{p_i - (0,0)_i}{W_i - (0,0)_i} \tag{3.16}$$

$$\frac{\phi_{k,i}(p) - \delta R_{k,i}}{\delta R_{k,i} + S_{k,i} - \delta R_{k,i}} = \frac{p_i}{W_i} \tag{3.17}$$

$$W_i(\phi_{k,i}(p) - \delta R_{k,i}) = S_{k,i}p_i \tag{3.18}$$

$$W_i\phi_{k,i}(p) = S_{k,i}p_i + W_i\delta R_{k,i} \tag{3.19}$$

$$\phi_{k,i}(p) = \frac{S_{k,i}p_i + W_i\delta R_{k,i}}{W_i} \tag{3.20}$$

$$\phi_{k,i}(p) = \frac{\frac{p_i(kW_i - kU_i + mU_i)}{m} + W_i\frac{k(U_i - W_i + 2F_i)}{2m}}{W_i} \tag{3.21}$$

$$\phi_{k,i}(p) = \frac{\frac{2p_i(kW_i - kU_i + mU_i) + W_ik(U_i - W_i + 2F_i)}{2m}}{W_i} \tag{3.22}$$

$$\phi_{k,i}(p) = \frac{2p_i(kW_i - kU_i + mU_i) + W_ik(U_i - W_i + 2F_i)}{2mW_i} \tag{3.23}$$

$$\phi_{k,i}(p) = \frac{kW_i(U_i - W_i) + 2kW_iF_i + 2p_i(kW_i - kU_i + mU_i)}{2mW_i} \tag{3.24}$$

□

3.1.1 Exemplo

O seguinte exemplo ilustra os valores das variáveis para uma dada tupla MNFM especificada na Tabela 3.1.1. O cálculo das variáveis utilizadas na construção do modelo pode ser visto na Tabela 3.1.1 e o resultado da amostragem pode ser visto na Figura 3.6.

Tabela 3.1: Variáveis utilizadas no exemplo

Variável	Valor
I	$e^{-((x-160)^2+(y-120)^2)/8000}$
U	(320, 240)
W	(32, 24)
F	(60, 60)
m	3

Tabela 3.2: Cálculo das variáveis no modelo

Variável	Equação	Valor
δR_k	$(68k, 56k)$	{(0,0),(68,56),(136,112),(204,168)}
S_k	$(320-96k,240-72k)$	{(320,240),(224,168),(128,96),(32,24)}

3.1.2 Funções de Mapeamento

A possibilidade de mapeamento entre posições referentes à imagem original (incluindo a posição da fóvea), posições referentes a cada nível e posições de nível para nível, é importante para a utilização do modelo de fóvea móvel com sucesso. Definimos aqui ϕ como sendo a função que, dado um ponto (píxel) em um determinado nível de resolução R_k, mapeia este no ponto correspondente na imagem original I. Então, a sua inversa ϕ_k^{-1} mapeia um ponto da imagem original I em um ponto do nível de resolução R_k. Esses mapeamentos são formalizados melhor a seguir. Os algoritmos de nosso modelo de fóvea móvel que serão descritos em seções posteriores utilizam tais mapeamentos.

Mapeamento da imagem original I para um nível R_k

Proposição 3.1.5. *A posição de um ponto em um determinado nível k, pode ser calculada a partir de sua posição p na imagem original pela função:*

$$\phi_{k,i}^{-1}(p) = \frac{2mW_i p_i - kW_i(U_i - W_i) - 2kW_i F_i}{2(mU_i - kU_i + kW_i)} \tag{3.25}$$

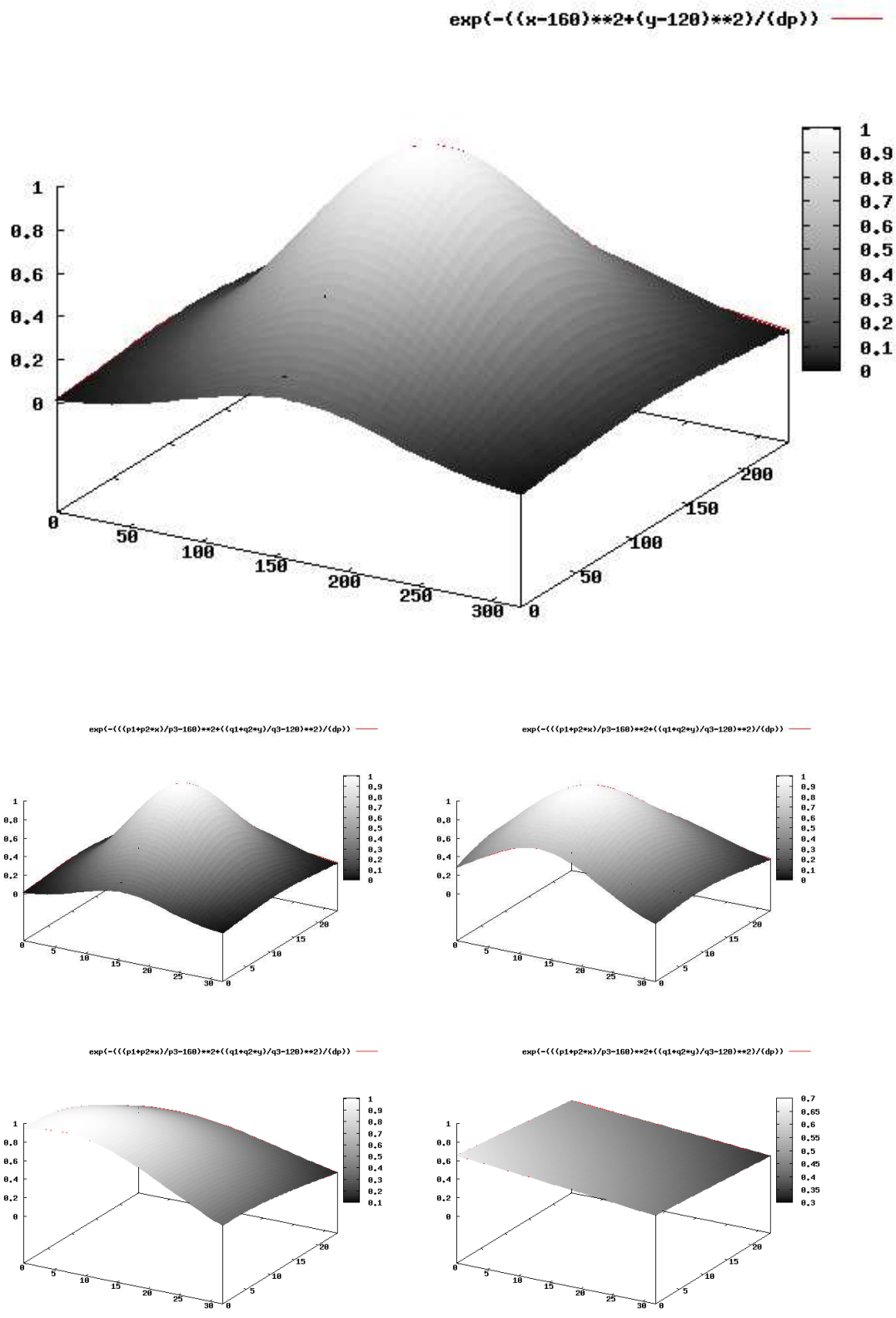

Figura 3.6: Reamostragem multi-níveis.

restrito a:

$$0 \leq \phi_{k,i}^{-1}(p) \leq W_i \tag{3.26}$$

Demonstração. *Reescrevendo a função ϕ da proposicão 3.1.4, temos:*

$$2mW_i\phi_{k,i}(p) = kW_i(U_i - W_i) + 2kW_iF_i + 2p_i(mU_i - kU_i + kW_i) \tag{3.27}$$

$$p_i = \frac{2mW_i\phi_{k,i}(p) - kW_i(U_i - W_i) - 2kW_iF_i}{2(mU_i - kU_i + kW_i)} \tag{3.28}$$

$$\phi_{k,i}^{-1}(p) = \frac{2mW_ip_i - kW_i(U_i - W_i) - 2kW_iF_i}{2(mU_i - kU_i + kW_i)} \tag{3.29}$$

□

Mapeamento entre níveis

Suponha agora que queiramos mapear um ponto no nível k para o nível j. Podemos fazer essa transformação primeiro transformando o ponto para o referencial em I através de ϕ e segundo transformando esse novo ponto para o referencial no nível j através de ϕ^{-1}. Seja $\omega_{k,j}$ esta função que mapeia um ponto do nível k para o nível j, obtemos:

Proposição 3.1.6.

$$\omega_{k,j,i}(p) = \frac{W_i(U_i - W_i)(k-j) + 2W_i(k-j)F_i + 2p_i(mU_i - kU_i + kW_i)}{2(mU_i - jU_i + jW_i)} \tag{3.30}$$

Demonstração.

$$\omega_{k,j,i}(p) = \phi_{j,i}^{-1}(\phi_{k,i}(p)) \tag{3.31}$$

$$\omega_{k,j,i}(p) = \phi_{j,i}^{-1}\left(\frac{kW_i(U_i - W_i) + 2kW_iF_i + 2p_i(mU_i - kU_i + kW_i)}{2mW_i}\right) \tag{3.32}$$

$$\omega_{k,j,i}(p) = \frac{2mW_i\left(\frac{kW_i(U_i - W_i) + 2kW_iF_i + 2p_i(mU_i - kU_i + kW_i)}{2mW_i}\right) - jW_i(U_i - W_i) - 2jW_iF_i}{2(mU_i - jU_i + jW_i)} \tag{3.33}$$

$$\omega_{k,j,i}(p) = \frac{kW_i(U_i - W_i) + 2kW_iF_i + 2p_i(mU_i - kU_i + kW_i) - jW_i(U_i - W_i) - 2jW_iF_i}{2(mU_i - jU_i + jW_i)} \tag{3.34}$$

$$\omega_{k,j,i}(p) = \frac{W_i(U_i - W_i)(k-j) + 2W_i(k-j)F_i + 2p_i(mU_i - kU_i + kW_i)}{2(mU_i - jU_i + jW_i)} \tag{3.35}$$

□

O domínio de ω é $A_k \cap A_j$, isto é, pontos que pertencem a ambos os níveis. Vale observar que o domínio da função para o caso $k < j$, isto é, o ponto correspondente no nível j pode estar fora do seu domínio. Nas implementações, deve-se, então, verificar esta possibilidade, visando evitar problemas.

3.2 O modelo multi-resolução com fóvea móvel no espaço discreto

A discretização do modelo anterior é feita mediante algumas modificações na definição do modelo. A imagem I passa a ser uma imagem digital, cujo domínio é $\mathbb{N}^2$ e cujo contra-domínio é um espaço de cores qualquer $\mathbb{S}$. Com a discretização, a indexação da imagem é de 0 a $U_i - 1$ e não mais de 0 a U_i, assim como em cada nível (0 a $W_i - 1$). A principal diferença está no redimensionamento. No espaço contínuo, podemos calcular com exatidão os respectivos pontos no redimensionamento através de uma interpolação linear. Poderíamos realizar o mesmo para o espaço discreto, isto é, fixar os pixels extremos e interpolar os demais, assim como fixamos os pontos extremos no contínuo (vide Figura 3.5). Entretanto há uma correspondência de 1 pixel para 1 pixel e devido ao fator de redimensionamento ser relativamente grande, pode-se produzir artefatos indesejados, principalmente ocasionados por ruído. Desse forma, é suficiente para nossos propósitos deixar a operação de redimensionamento livre para escolha, pois quem utilizar o modelo pode escolher esta operação e optar por uma ou outra operação de acordo com o seu custo benefício, principalmente no tocante razão sinal-ruído e custo computacional. Ademais, o usuário do modelo pode optar pelo redimensionamento utilizando a função ϕ que será redefinida para o espaço discreto. As mesmas proposições definidas na seção anterior são redefinidas devido às modificações na definição do modelo.

Há diversas abordagens na literatura para redimensionar uma imagem discreta. Na interpolação de ordem zero, arredondanda-se para o inteiro mais próximo, conforme a Equação 3.36.

$$R_k(p) = I(nint(\phi_k(p))) \tag{3.36}$$

A grande desvantagem dessa abordagem é a produção de artefatos indesejáveis [Gonzales & Woods 1992]. Outros métodos obtêm o valor não somente de uma posição específica da imagem, mas também de um conjunto de valores no qual é aplicada uma função como, por exemplo, uma média aritmética (c é uma constante):

$$R_k(x,y) = \frac{I(\phi_k(x-c,y)) + I(\phi_k(x+c,y)) + I(\phi_k(x,y+c)) + I(\phi_k(x,y-c))}{4} \tag{3.37}$$

No método de interpolação por convolução cúbica, os valores são obtidos através da fixação de uma superfície do tipo $(senx)/x$ através de um grupo de vizinhos do ponto desejado. Apesar de se obter estimativas de valores mais suaves, esse método possui grande custo computacional [Gonzales & Woods 1992]. Um método que contempla boa estimativa a baixo custo é a interpolação bilinear, que consiste em realizar uma interpolação dos valores dos 4 vizinhos mais próximos [Gonzales & Woods 1992].

Definição 3.2.1. *Definição do modelo multi-resolução com fóvea móvel*

O modelo multi-resolução com fóvea móvel é composto por uma estrutura definida pela tupla $(I, \boldsymbol{U}, \boldsymbol{W}, m, \boldsymbol{F}, B)$. *A imagem digital* $I : \mathbb{N}^2 \rightarrow \mathbb{S} \subset [0...U_0 - 1, 0...U_1 - 1]$, *onde* $\boldsymbol{U} = (U_0, U_1)$ *e* $\mathbb{S}$ *é o espaço de cores, é redimensionada em níveis numerados de* 0 *a*

m (m + 1 níveis). Cada k-ésimo nível $R_k : \mathbb{N}^2 \to \mathbb{S} \subset [0...W_0 - 1, 0...W_1 - 1]$, *tal que* $\boldsymbol{W} = (W_0, W_1) \in \mathbb{N}^2$ *e* $W_i < U_i$, *é definido por uma operação* Φ, *isto é,* $R_k = \Phi(I)$. Φ *é uma operação de redimensionamento de imagens. A região* A_k *é definida pelo produto cartesiano* $[\delta R_{k,0}, \delta R_{k,0} + S_{k,0}] \times [\delta R_{k,1}, \delta R_{k,1} + S_{k,1}]$, *onde* $\boldsymbol{S}_k = (S_{k,0}, S_{k,1}) \in \mathbb{N}^2$ *define o tamanho dessa região. Por definição,* $\delta R_0 = (0,0)$, $S_0 = \boldsymbol{U}$ *e* $S_m = \boldsymbol{W}$. *Cada componente de* S_k *é definido como uma interpolação linear dos respectivos componentes entre* S_0 *e* S_m. *Cada componente de* δR_k *é definido como uma interpolação linear dos respectivos componentes entre* δR_0 *e* δR_m. A_k *deve variar de forma que* F' *seja o centróide de* A_m. F, *o vetor fóvea, é definido por* $F' - \frac{U}{2}$. B *é uma lista de* $m+1$ *elementos, tal que* $B_i = 0$ *indica que o i-ésimo nível é descartado. Se B for omitido, considera-se B tal que todos os seus elementos são igual a* 1.

3.2.1 Funções de mapeamento

De forma similar ao caso contínuo, a possibilidade de mapeamento entre posições referentes à imagem original (incluindo a posição da fóvea), posições referentes a cada nível e posições de nível para nível, é importante para a utilização do modelo de fóvea móvel com sucesso. Assim, redefinimos aqui as funções de mapeamento.

Mapeamento de um nível R_k para a imagem original

No caso discreto, utilizamos uma interpolação diferente, pois quando $p = (0,0)$ queremos que o resultado seja δR_k (no contínuo também) e quando $p = W - 1$ seja $\delta R_k + S_k - (1,1)$ (diferente do contínuo que seria $\delta R_k + S_k$ para $p = W$). Ademais, assim como no contínuo, os componentes dos demais pontos são definidos como uma interpolação linear dos respectivos componentes entre esses dois valores.

Proposição 3.2.2.

$$\phi_{k,i}(p) = \frac{k(W_i - 1)(U_i - W_i) + 2k(W_i - 1)F_i + 2p_i(mU_i - kU_i + kW_i - m)}{2m(W_i - 1)} \tag{3.38}$$

Demonstração.

$$\frac{\phi_{k,i}(p)-\phi_{k,i}(0,0)}{\phi_{k,i}(W-(1,1))-\phi_{k,i}(0,0)} = \frac{p_i-(0,0)_i}{(W-(1,1))_i-(0,0)_i} \tag{3.39}$$

$$\frac{\phi_{k,i}(p)-\delta R_{k,i}}{\delta R_{k,i}+S_{k,i}-1-\delta R_{k,i}} = \frac{p_i}{W_i-1} \tag{3.40}$$

$$(W_i-1)(\phi_{k,i}(p)-\delta R_{k,i}) = (S_{k,i}-1)p_i \tag{3.41}$$

$$(W_i-1)\phi_{k,i}(p) = (S_{k,i}-1)p_i+(W_i-1)\delta R_{k,i} \tag{3.42}$$

$$\phi_{k,i}(p) = \frac{(S_{k,i}-1)p_i+(W_i-1)\delta R_{k,i}}{W_i-1} \tag{3.43}$$

$$\phi_{k,i}(p) = \frac{\frac{p_i(kW_i-kU_i+mU_i)}{m}-p_i+(W_i-1)\frac{k(U_i-W_i+2F_i)}{2m}}{W_i-1} \tag{3.44}$$

$$\phi_{k,i}(p) = \frac{\frac{2p_i(kW_i-kU_i+mU_i)-2mp_i+(W_i-1)k(U_i-W_i+2F_i)}{2m}}{W_i-1} \tag{3.45}$$

$$\phi_{k,i}(p) = \frac{2p_i(kW_i-kU_i+mU_i)-2mp_i+(W_i-1)k(U_i-W_i+2F_i)}{2m(W_i-1)} \tag{3.46}$$

$$\phi_{k,i}(p) = \frac{k(W_i-1)(U_i-W_i)+2k(W_i-1)F_i+2p_i(kW_i-kU_i+mU_i-m)}{2m(W_i-1)} \tag{3.47}$$

□

Mapeamento da imagem original I para um nível R_k

Proposição 3.2.3. *No caso discreto, esta posição em um determinado nível k pode ser calculada pela função:*

$$\phi_k^{-1}(p) = \frac{2m(W-1)p-k(W-1)(U-W)-2k(W-1)F}{2(mU-kU+kW-m)} \tag{3.48}$$

restrito a:

$$0 \leq \phi_k^{-1}(p) \leq W-1 \tag{3.49}$$

Demonstração. *Reescrevendo 3.2.2, temos:*

$$2mW\phi_k(p) = kW(U-W)+2kWF+2p(mU-kU+kW) \tag{3.50}$$

$$p = \frac{2mW\phi_k(p)-kW(U-W)-2kWF}{2(mU-kU+kW)} \tag{3.51}$$

$$\phi_k^{-1}(p) = \frac{2mWp-kW(U-W)-2kWF}{2(mU-kU+kW)} \tag{3.52}$$

□

Mapeamento entre níveis

Suponha agora que queiramos mapear um ponto no nível k para o nível j. Podemos fazer essa transformação primeiro transformando o ponto para o referencial em I através de ϕ e segundo transformando esse novo ponto para o referencial no nível j através de ϕ^{-1}. Seja $\omega_{k,j}$ esta função que mapeia um ponto do nível k para o nível j, obtemos:

Proposição 3.2.4.

$$\omega_{k,j,i}(p) = \frac{(W_i-1)(U_i-W_i)(k-j)+2(W_i-1)(k-j)F_i+2p_i(mU_i-kU_i+kW_i-m)}{2(mU_i-jU_i+jW_i-m)} \quad (3.53)$$

Demonstração.

$$\omega_{k,j,i}(p) = \phi_{j,i}^{-1}(\phi_{k,i}(p)) \quad (3.54)$$

$$\omega_{k,j,i}(p) = \phi_{j,i}^{-1}(\frac{k(W_i-1)(U_i-W_i)+2k(W_i-1)F_i+2p_i(mU_i-kU_i+kW_i-m)}{2m(W_i-1)}) \quad (3.55)$$

$$\omega_{k,j,i}(p) = \frac{2m(W_i-1)(\frac{k(W_i-1)(U_i-W_i)+2k(W_i-1)F_i+2p_i(mU_i-kU_i+kW_i-m)}{2m(W_i-1)}) - j(W_i-1)(U_i-W_i) - 2j(W_i-1)F_i}{2(mU_i-jU_i+jW_i-m)} \quad (3.56)$$

$$\omega_{k,j,i}(p) = \frac{k(W_i-1)(U_i-W_i)+2k(W_i-1)F_i+2p_i(mU_i-kU_i+kW_i-m) - j(W_i-1)(U_i-W_i) - 2j(W_i-1)F_i}{2(mU_i-jU_i+jW_i-m)} \quad (3.57)$$

$$\omega_{k,j,i}(p) = \frac{(W_i-1)(U_i-W_i)(k-j)+2(W_i-1)(k-j)F_i+2p_i(mU_i-kU_i+kW_i-m)}{2(mU_i-jU_i+jW_i-m)} \quad (3.58)$$

□

O domínio de ω é $A_k \cap A_j$, isto é, pontos que pertencem a ambos os níveis. Vale observar que o domínio da função para o caso $k < j$, isto é, o ponto correspondente no nível j pode estar fora do seu domínio. Nas implementações, deve-se, então, verificar esta possibilidade, visando evitar problemas.

3.3 Resolução e custo da estrutura

Nessa seção, introduzimos e apresentamos alguns índices visando analisar a robustez e precisão da estrutura proposta em relação a alguns parâmetros. O índice G refere-se à resolução da estrutura e tenta medir a perda de resolução em relação à imagem original, variando de 0 a 1. Se $G = 1$, temos a mesma resolução da imagem original. O índice C refere-se ao total de píxels na estrutura em relação ao total de píxels na imagem original. Se $C = 1$, temos o mesmo numero de pixels da imagem original e provavelmente a configuração da estrutura não está adequada. Este índice dá uma idéia do ganho computacional.

3.3.1 Índice G

No índice proposto, G é influenciado pela resolução de cada um dos níveis R_k. A influência da resolução de R_k é ponderada por um subconjunto B_k da região de redimensionamento A_k, de tal forma que esse subconjunto é composto apenas pelos pontos que não pertencem aos níveis de maior resolução, isto é, $B_k = A_k - A_{k+1}$ para $k < m$ e $B_k = A_k$ para $k = m$. Essa idéia é motivada por desejarmos obter na estrutura completa, a maior

cobertura de regiões da imagem original com maior resolução possível. Um exemplo do subconjunto para cada nível é ilustrado na Figura 3.7.

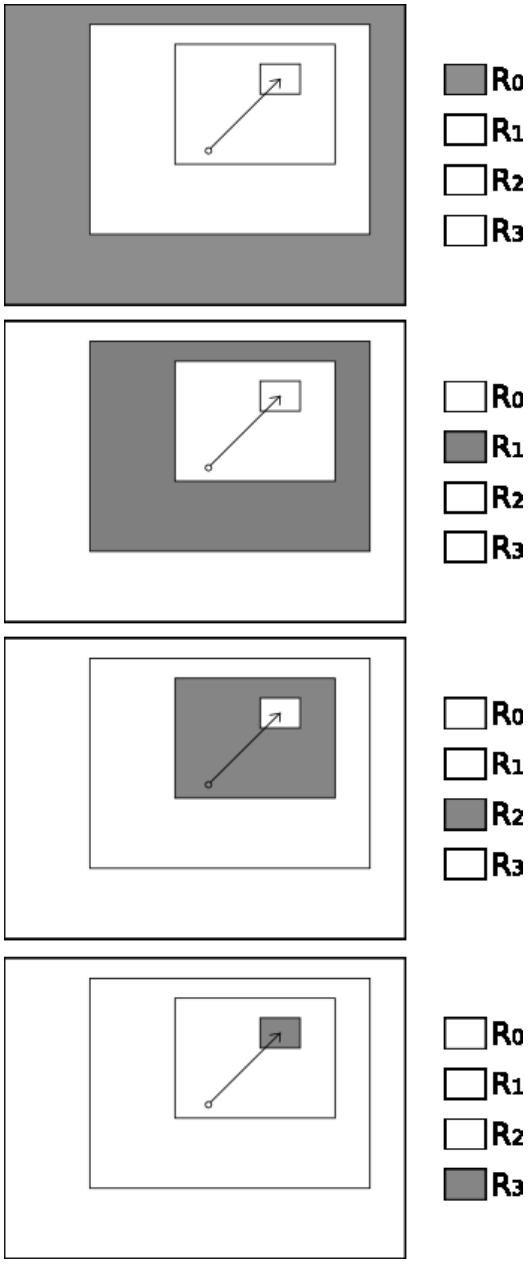

Figura 3.7: Área de ponderação para cada um dos níveis na determinação do índice G.

Para calcular G, realizamos uma iteração em todos os níveis. Para cada nível, calculamos a área de pontos que pertencem a este nível e não pertença a níveis de maior resolução. Essa área deve ser relativa ao tamanho da imagem original. Então, por exemplo, no último nível temos W_0W_1 pixels, que representam $\frac{W_0W_1}{U_0U_1}$ do total da imagem original. Generalizando para qualquer nivel k teremos: $\frac{S_{k,0}S_{k,1}-S_{k+1,0}S_{k+1,1}}{U_0U_1}$. A resolução de

cada nível é dada por $\frac{W_0 W_1}{S_{k,0} S_{k,1}}$, logo a fórmula geral fica:

$$G = \frac{W_0 W_1}{U_0 U_1} + \sum_{k=0}^{m-1} \frac{W_0 W_1 (S_{k,0} S_{k,1} - S_{k+1,0} S_{k+1,1})}{S_{k,0} S_{k,1} U_0 U_1}$$

$$G = \frac{W_0 W_1}{U_0 U_1} + \frac{W_0 W_1}{U_0 U_1} \sum_{k=0}^{m-1} 1 - \frac{S_{k+1,0} S_{k+1,1}}{S_{k,0} S_{k,1}}$$

3.3.2 Índice C

O custo é simplesmente a soma da quantidade de pixels de todos os níveis em relação ao total de pixels da imagem original:

$$C = \frac{W_0 W_1 (m+1)}{U_0 U_1}$$

3.3.3 Análise de índices

Um gráfico com os índices G e C foi plotado para $U = (320, 320)$ e $W = (32, 32)$ (Figura 3.8).

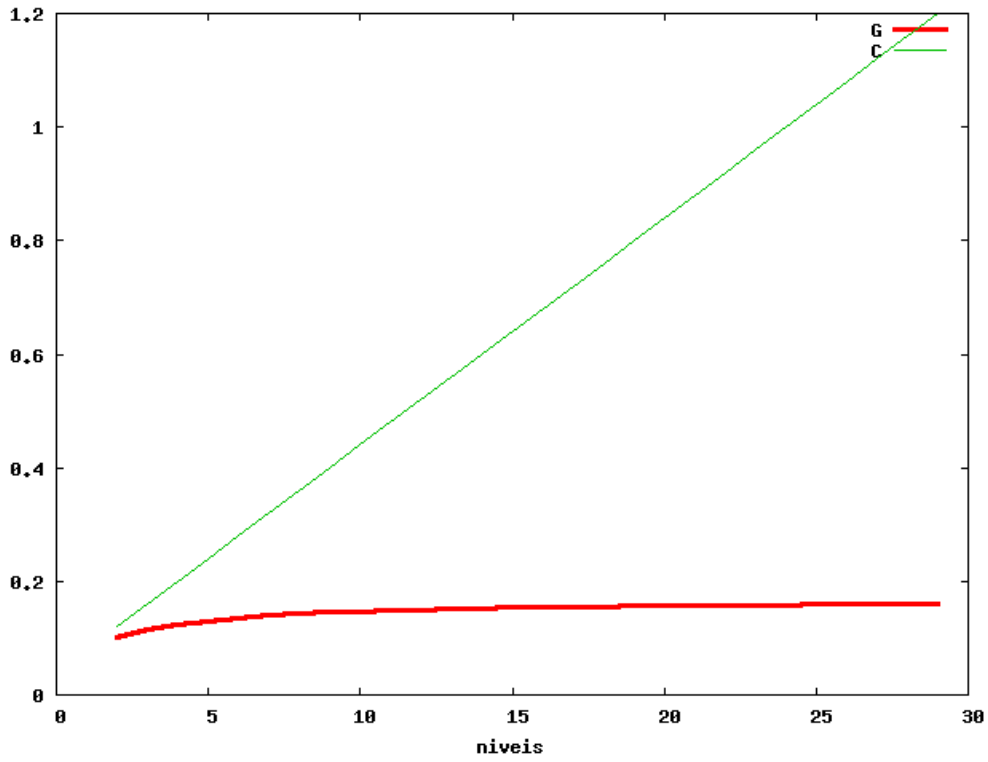

Figura 3.8: Gráfico com os índices G e C para $U = (320, 320)$ e $W = (32, 32)$

Podemos observar que a partir de certo ponto (por exemplo, a partir de 7 níveis) não vale a pena incluir mais níveis, pois melhoramos muito pouco a qualidade da estrutura a um custo muito alto.

Capítulo 4

Implementações em Visão Robótica

As atividades autônomas de um robô geralmente envolvem tarefas consideradas de mais alto nível, tal como reconhecimento ou atenção visual, exigindo para isso a extração de características, em nosso caso, a partir de imagens obtidas do ambiente. Isso significa que é necessário realizar processamentos sobre essas imagens capturadas para fornecer informações para processamento de mais alto nível, visando análise da cena. As características podem incluir por exemplo o gradiente da imagem, uma filtragem Gaussiana, a estimação de movimento e a disparidade estéreo. Para produzir características úteis, neste trabalho, nós utilizamos uma estrutura enxuta de multi-características (MC) extraídas a partir da representação em multi-resolução (MR), introduzida anteriormente[Gonçalves et al. 2000]. As dicotomias MC e MR representam o mapeamento de índices topológicos/espaciais dos sensores para tarefas de atenção múltipla ou reconhecimento. A junção de ambas (multi-resolução-multi-característica) refere-se ao conjunto da estrutura de multi-resolução mais as informações processadas a partir desta. Porém, essas informações não se limitam a serem obtidas exclusivamente da estrutura de multi-resolução, mas também de fontes externas, tais como conhecimento a priori e outra estrutura de multi-resolução (necessário, por exemplo, para o cálculo de disparidade estéreo).

Nas implementações realizados no contexto de visão robótica, trabalhamos simultaneamente com duas estruturas MRMC, uma para cada câmera de um sistema visual estéreo. Nesse caso, convencionamos que todas as variáveis da estrutura possuam como primeiro índice o identificador da estrutura. Por exemplo, $R_{1,0}$ é o primeiro nível da primeira estrutura e $R_{2,0}$ é o primeiro nível da segunda estrutura.

4.1 Calibrando as câmeras

A calibração tem como objetivo principal permitir realizar mapeamentos (ou transformações) entre os sistemas de coordenadas do mundo, coordenadas de câmera e coordenadas de imagem. Além disso, ela permite corrigir as distorções inerentes ao sistema ótico da câmera. Há basicamente duas categorias de parâmetros que são recuperados a partir do procedimento de calibração: intrínsecos e extrínsecos [Trucco & Verri 1998]. Os intrínsecos referem-se aos parâmetros do sistema ótico: foco, centro da imagem, dimensão do pixel e coeficientes de distorção. Esses podem ser calculados só uma vez, desde que o sistema óptico não mude (por exemplo, modifique o foco). Já os extrínsecos referem-se

à transição entre os sistemas de coordenadas. São úteis, por exemplo, para auxiliar na odometria, no sistema de posicionamento e em estimativas da projeção de um ponto no mundo para a imagem. À medida que o robô navega, é necessário re-calibrar a câmera quanto aos parâmetros extrínsecos. A maior parte dos métodos obtém tais parâmetros através da associação de pontos no mundo e pontos correspondentes na imagem.

O fato é que podemos aproveitar que estamos fazendo uma re-amostragem para referenciar às coordenadas corrigidas pela calibração da câmera acelerando os cálculos, ao invés de tirar a distorção em toda a imagem e em seguida realizar a re-amostragem. Seja (x',y') a coordenada de imagem, por exemplo pelo método de tsai [Tsai 1987], a coordenada corrigida pelos parâmetros intrínsecos pode ser dada por (u,v):

$$u = f_x x'' + c_x$$

$$v = f_y y'' + c_y$$

onde:

$$x'' = x'(1 + k_1 r^2 + k_2 r^4) + 2p_1 x' y' + p_2(r^2 + 2x'^2)$$

$$y'' = y'(1 + k_1 r^2 + k_2 r^4) + 2p_2 x' y' + p_1(r^2 + 2y'^2)$$

f é a distância focal, (c_x, c_y) é a localização do centro da imagem, (k_1, k_2) são coeficientes de distorção radial e (p_1, p_2) são coeficientes de distorção tangencial.

4.2 Estrutura Multi-Resolução Multi-Característica

Nas implementações realizadas, a estrutura MRMC é composta pela aplicação de filtros e pelo cálculo de movimento e de disparidade estéreo. As subseções a seguir tratam da implementação de cada um desses componentes e de como é possível integrá-los ao esquema de multi-resolução com fóvea móvel.

4.2.1 Filtros

O uso de filtros em processamento de imagens permite alterar ou extrair informações de uma imagem digital. Entre os processos de modificação incluem: suavização de imagens, eliminação de ruídos, realce de características, detecção de bordas, dentre outros. A efetivação de um filtro em uma imagem é ser feita pela operação de convolução entre a imagem em questão e uma máscara determinada pelo filtro. Nos experimentos realizados, foram aplicados filtros passa-baixa (filtros que eliminam componentes de baixa freqüência, como o Gaussiano e o Gradiente do Gaussiano) que visam atenuar ruídos e abstrair detalhes ao mesmo tempo e filtros passa-alta (filtros que eliminam componentes de alta frequência, como o Laplaciano do Gaussiano e outros) que visam detectar contornos dos objetos.

4.2.2 Análise de movimento

De acordo com Horn et al [Horn & Schunck 1992], fluxo óptico é a distribuição das aparentes velocidades de movimento pelos padrões de brilho em uma imagem. A obtenção de fluxo óptico é uma das formas através da qual pode-se analisar movimentos a partir de sequências de imagens. Dado um ponto P, o fluxo óptico estima um vetor que represente o deslocamento deste ponto projetado no plano imagem a partir do brilho. Para isso, assume-se que esse ponto não muda de intensidade, isto é: $\frac{dE(x,y,t)}{dt} = 0$ [Trucco & Verri 1998]. Essa equação pode ser reescrita como $\frac{\partial E}{\partial x}\frac{dx}{dt} + \frac{\partial E}{\partial y}\frac{dy}{dt} + \frac{\partial E}{\partial t} = 0$ [Trucco & Verri 1998]. Os termos $\frac{\partial E}{\partial x}$ e $\frac{\partial E}{\partial y}$ são os gradientes em x e em y da imagem e podem ser obtidos da própria estrutura MRMC. O termo $\frac{\partial E}{\partial t}$ é a variação de brilho no tempo e $(\frac{dx}{dt}, \frac{dy}{dt})$ é o vetor do fluxo óptico para o ponto P.

Podemos obter o vetor de fluxo óptico, através da seguinte equação [Trucco & Verri 1998], onde o somatório é realizado sobre uma vizinhança Q em torno do ponto P

$$\begin{bmatrix} \sum E_x^2 & \sum E_x E_y \\ \sum E_x E_y & E_y^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum -E_t \\ \sum -E_t \end{bmatrix} \tag{4.1}$$

Esse sistema pode ser então resolvido por SVD (*Singular Value Decomposition*).

Nas Figuras 4.1 e 4.2, ilustram dois exemplos do cálculo de fluxo óptico.

4.2.3 Cálculo de disparidade estéreo

A imagem capturada por dispositivos de vídeo, assim como por nossos olhos, são projeções de um domínio tridimensional para um ambiente de domínio perceptualmente bidimensional. Obter a inversa dessa transformação significa que podemos, dada uma imagem, conhecer a distâncias de todos os pontos desta imagem em relação ao plano de projeção, e portanto, conhecer a forma dos objetos e o quão estão distantes. A reconstrução do ambiente possui grande aplicabilidade em robótica, como por exemplo, permitir que um robô perceba a aproximação de um objeto ou desvie de um obstáculo [Thompson & Kagami 2005].

Já para obtenção aproximada de distância, é necessário duas ou mais câmeras. Dado um ponto que é projetado em ambas as câmeras, se for possível determinar em qual ponto de cada imagem esse primeiro ponto foi projetado, é possível, através de uma triangulação, obter a distância deste ponto em relação ao plano de projeção. Determinar dois pontos correspondentes em duas imagens é um problema descrito na literatura como correspondência estéreo. Scharstein et al. [Scharstein & Szeliski 2002] classificam os algoritmos de correspondência estéreo em 4 categorias: os métodos locais, os que usam otimização global, os que usam programação dinâmica e os algoritmos cooperativos.

O sistema que implementamos para validação inclui o cálculo de disparidade que envolve determinar a correspondência estéreo para cada pixel. Além de ser um processo comum em visão robótica, a correspondência estéreo permite avaliar pelos menos dois aspectos importantes do modelo de multi-resolução com fóvea móvel: o refinamento do cálculo de disparidade através dos níveis e a interação entre duas imagens, cada qual com

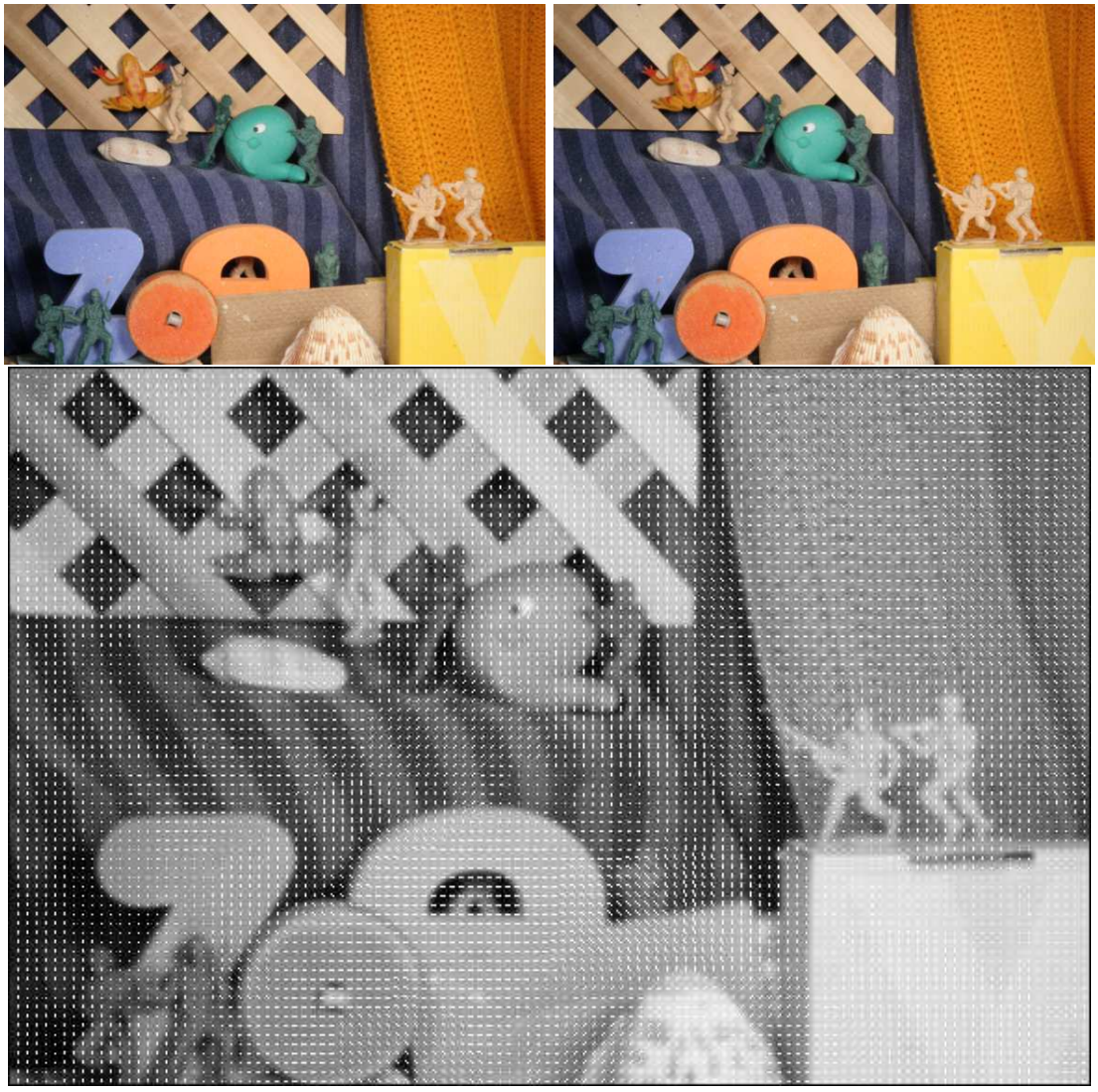

Figura 4.1: Exemplo do cálculo do fluxo óptico para a primeira sequência de imagens: (a) Primeira imagem da sequência. (b) Segunda imagem da sequência. (c) Fluxo ótico calculado.

Figura 4.2: Exemplo do cálculo do fluxo óptico para a segunda sequência de imagens: (a) Primeira imagem da sequência. (b) Segunda imagem da sequência. (c) Fluxo ótico calculado.

uma coordenada individual do centro da fóvea. Assumimos aqui que ambas as estruturas possuem as seguintes variáveis iguais: U, W e m.

Uma vez que as duas estruturas MRMC estejam construídas, a disparidade é calculada usando a operação de correlação entre os pixels de ambas as imagens, que é na prática implementado através de várias operações de convolução [Marr 1982]. Entretanto, o ideal é obter a disparidade como se houvéssemos a calculado na imagem original, pois os pixels em cada nível representam localidades diferentes da imagem original. Então, dada a disparidade calculada em um nível, a seguinte proposicão estabelece a mesma disparidade, caso essa disparidade fosse calculada na imagem original.

Proposição 4.2.1. *Sejam dois pontos a (esquerda) e b (direita) correspondentes. A disparidade local (relativa aos níveis, digamos k) é $D(a) = a - b$. A disparidade global (relativa à imagem original), no caso discreto, é dada por:*

$$d(a) = \frac{k(W_0-1)(F_{1,0}-F_{2,0}) + (a-b)(mU_0 - kU_0 + kW_0 - m)}{m(W_0-1)} \tag{4.2}$$

Demonstração.

$$d(a) = \phi_{1,k,0}(a) - \phi_{2,k,0}(b) \tag{4.3}$$

$$d(a) = \frac{2k(W_0-1)F_{1,0} + 2a(mU_0 - kU_0 + kW_0 - m) - 2k(W_0-1)F_{2,0} - 2b(mU_0 - kU_0 + kW_0 - m)}{2m(W_0-1)} \tag{4.4}$$

$$d(a) = \frac{2k(W_0-1)(F_{1,0}-F_{2,0}) + 2(a-b)(mU_0 - kU_0 + kW_0 - m)}{2m(W_0-1)} \tag{4.5}$$

$$d(a) = \frac{k(W_0-1)(F_{1,0}-F_{2,0}) + (a-b)(mU_0 - kU_0 + kW_0 - m)}{m(W_0-1)} \tag{4.6}$$

□

A disparidade estéreo pode ser calculada independentemente para cada nível ou através de um algoritmo de refinamento, o que pode resultar em ganho de tempo computacional. A Figura 4.3 ilustra a idéia do refinamento: deseja-se calcular a disparidade do pixel a em $R_{1,1}$. Suponha que a disparidade no nível R_0 já foi calculado. A estimativa inicial é de que o seu correspondente estéreo a' é o mesmo pixel correspondente estéreo (b') de b no nível k, isto é, $b' = c$. b pode ser calculado através da função $\omega_{1,k,k-1}$. O pixel c é dado no nível $k-1$, então a estimativa inicial é dado por esse ponto no nível k, isto é, o ponto d. Este ponto pode ser calculado através da função $\omega_{2,k-1,k}$. Dada a estimativa inicial d, é feito uma busca entre os vizinhos de d (pixels entre $d-t$ e $d+t$ em $R_{2,1}$) na tentativa de achar um melhor correspondente estéreo.

Para o cálculo de correspondência estéreo usando multi-resolução com fóvea móvel, primeiro calculamos a disparidade em R_0, de menor resolução. A disparidade para os demais níveis pode ser calculada usando uma estimativa a partir do nível anterior e através do esquema de refinamento descrito acima. Assim, podemos calcular a disparidade para todos os níveis usando o seguinte algoritmo:

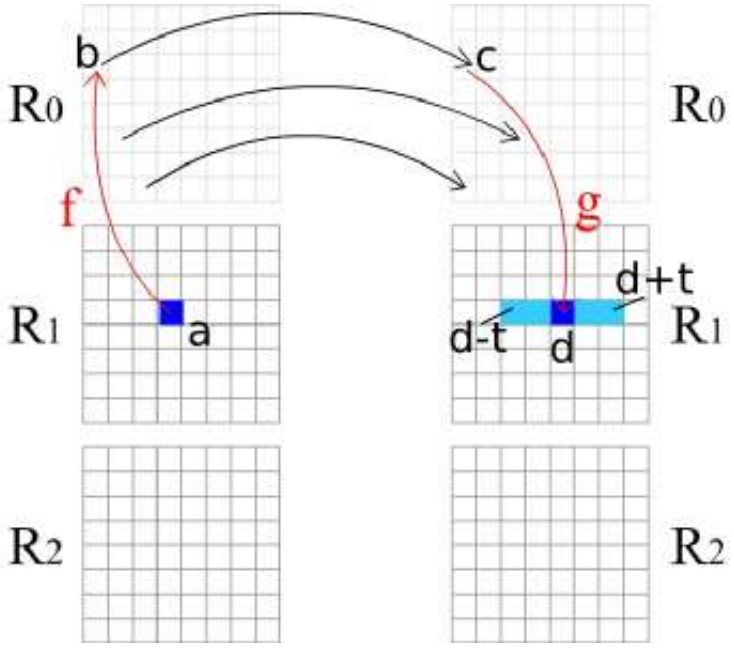

Figura 4.3: Estimativa e refinamento do cálculo de disparidade estéreo.

4.3 Posicionando a fóvea

Vale ressaltar que, em se tratando de um modelo para visão ativa, algumas das dicotomias do processo de atenção visual são importantes, como atenção *bottom-up* em que o estímulo aparece sem estar sendo esperado, ou *top-down* em que se espera que o estímulo apareça em um determinado local da imagem. Temos ainda que considerar a atenção encoberta ou descoberta. No nosso modelo, propomos uma maneira de se realizar a atenção encoberta, isto é, prestar atenção em uma determinada região sem que a câmera esteja efetivamente com esta região no centro da imagem. Isto implica em mover a fóvea sem mover os "olhos".

Mover a fóvea significa que podemos determinar a sua posição seguindo os princípios de visão ativa. Isto exige o uso de uma metodologia para prover um local onde prestar atenção. Várias técnicas podem ser encontradas na literatura visando determinar o foco de atenção, algumas inspiradas em modelos biológicos [Treisman 1964, Treisman 1985, Koch & Ullman 1985, Tsotsos 1987, Olshausen et al. 1964, Desimone & Duncan 1995], outras usando modelos estatísticos e/ou computacionais [Sandon 1990, Gonçalves et al. 2000, Rao & Ballard 2004]. Pode-se utilizar um mapa de saliências [Gonçalves et al. 2000], ou uma busca por objeto realizando um acompanhamento (ou *tracking*) [Nishihara et al. 1984, Wessler 1996, Batista et al. 2000] etc. O mapa de saliências pode ser calculado em tempo real, de acordo com experimentos realizados [Butko et al. 2008]. Os tempos obtidos no cálculo do mapa de saliências para diferentes resoluções de imagem executado em um Mac Mini com processador Intel 1.87 GHz Core Duo podem ser vistos na Tabela 4.3 [Butko et al. 2008].

Ainda, no caso de o sistema visual artificial possuir *n* imagens, no contexto de visão

Algoritmo 1 Correspondência estéreo para o nível k ($k > 0$) de I_1

para $i = 0$ para W_y **faça**
  para $j = 0$ para W_x **faça**
    $b \Leftarrow \phi_{1,k,k-1}(i, j)$
    $c \Leftarrow p + D_{1,k-1}(b)$
    $d \Leftarrow \phi_{2,k-1,k}(c)$
    se d está dentro de R_{2_k} **então**
      $D_{1_k}(i, j) = -\infty$
      para $k = -t$ para t **faça**
        se $(d_x + k, d_y)$ está dentro de R_{2_k} **então**
          $D_{1_k}(i, j) = MAX(D_{1_k}(i, j), corr(R_{2_k}, i, j + k))$
        fim se
      fim para
    fim se
  fim para
fim para

	80 x 60	160 x 120	320 x 240	640 x 480
Tempo	2.93ms	10.82ms	44.96ms	214.82ms

Tabela 4.1: Tempos para cálculo do mapa de saliências.

estéreo, é necessário fixar a fóvea de uma das imagens e, em seguida, determinar a posição da fóvea das demais imagens de forma que apontem para o mesmo ponto da cena. Esse problema é o mesmo da correspondência estéreo para um único ponto e, portanto, bem abordado na literatura. A correspondência pode ser realizada nos mais diversos níveis, por exemplo, no nível mais grosseiro. Entretanto, uma coordenada de um nível não corresponde com exatidão ao mesmo ponto em outro nível. Uma possível abordagem é a de realizar a correspondência no primeiro nível e, em seguida, refinar a busca nos níveis de maior resolução. Uma vez que a posição da fóvea esteja determinada em cada imagem, então, pode-se calcular a estrutura em multi-resolução e partir para a extração de características, visto a seguir.

Os valores de disparidade entre duas estruturas MRMC variam de acordo com a posição da fóvea. Com exceção do nível R_0, todos os níveis podem mapear regiões distintas de acordo com a posição da fóvea. Então, dada a posição da fóvea em uma das imagens (digamos I_1), precisamos determinar a posição da fóvea na outra imagem. Nós optamos por posicionar as fóveas direcionando-as a um mesmo ponto da cena. Essa busca pode ser feita usando correlação cruzada, considerando a restrição epipolar, isto é, $F_{2,y} = F_{1,y}$, para as imagens de Tsukuba. Para encontrar a posição da fóvea na outra imagem, procuramos o valor $F_{2,x}$ que maximiza a correlação entre uma janela centralizada em F_2 com uma janela centralizada em F_1, isto é, $F_{2,x} = \omega_0(p)$, tal que p maximiza $f_c(R_{1,0}(\upsilon_0(F_1)), R_{2,0}(p))$, onde f_c é uma função de correlação.

4.4 Aspectos implementacionais

Aqui fazemos algumas análises referentes a aspectos implementacionais que melhoram a performance dos algoritmos implementados. Reaproveitamento de cálculos anteriores e uso de tabelas são exemplos de melhorias introduzidas por nós.

4.4.1 Reaproveitando calculos

Para determinado nível k pode-se reaproveitar resultados que foram obtidos no nível $k+1$, uma vez que o nível $k+1$ sempre está contido no nível k. A Figura 4.4 ilustra um exemplo com 4 níveis, onde a região escura composta por A_2 é a região que pode ser reaproveitada para o nível R_1. Reaproveitar cálculos pode ser útil para operações independente de escala, por exemplo, uma conversão do espaço de cores, ou a transmissão de vídeo através do modelo proposto, uma vez alguns pixels nunca serão exibidos, pois cada nível k quando redimensionado sempre sobrepõe uma região do nível $k-1$. A questão a ser respondida é: qual a porcentagem de pixels que não precisam ser recalculados? A área de pontos de um nível k que está no nível $k-1$ é:

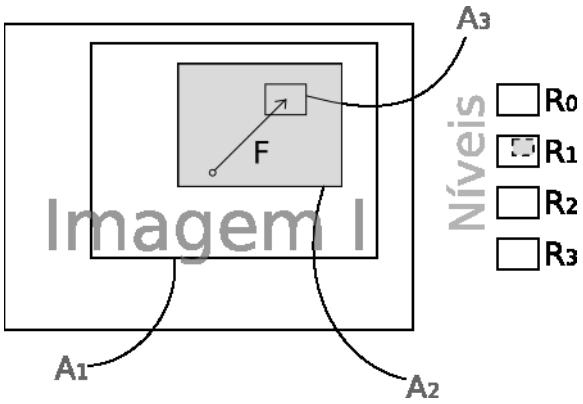

Figura 4.4: Reaproveitando cálculos.

Um nível k é determinado pelo retângulo

$$(\delta R_{k,0}, \delta R_{k,1}, \delta R_{k,0} + S_{k,0}, \delta R_{k,1} + S_{k,1})$$

Esses pontos para o nível j correspondem a:

$$(\phi_{j,0}^{-1}(\delta R_{k,0}), \phi_{j,1}^{-1}(\delta R_{k,1}), \phi_{j,0}^{-1}(\delta R_{k,0} + S_{k,0}), \phi_{j,1}^{-1}(\delta R_{k,1} + S_{k,1}))$$

cujo área é:

$$A = (\phi_{j,0}^{-1}(\delta R_{k,0} + S_{k,0}) - \phi_{j,0}^{-1}(\delta R_{k,0}))(\phi_{j,1}^{-1}(\delta R_{k,1} + S_{k,1}) - \phi_{j,1}^{-1}(\delta R_{k,1}))$$

$$A = \frac{2mW_0 S_{k,0}}{2(mU_0 - jU_0 + jW_0)} \frac{2mW_1 S_{k,1}}{2(mU_1 - jU_1 + jW_1)}$$

$$A = \frac{W_0(mU_0 - kU_0 + kW_0)}{mU_0 - jU_0 + jW_0} \frac{W_1(mU_1 - kU_1 + kW_1)}{mU_1 - jU_1 + jW_1}$$

Como $j = k - 1$ temos:

$$A = \frac{W_0(mU_0 - kU_0 + kW_0)}{mU_0 - kU_0 + kW_0 + U_0 - W_0} \frac{W_1(mU_1 - kU_1 + kW_1)}{mU_1 - kU_1 + kW_1 + U_1 - W_1}$$

No total teremos em porcentagem de pixels que podem ser reaproveitados:

$$\frac{\sum_{k=1}^{m} \frac{W_0(mU_0 - kU_0 + kW_0)}{mU_0 - kU_0 + kW_0 + U_0 - W_0} \frac{W_1(mU_1 - kU_1 + kW_1)}{mU_1 - kU_1 + kW_1 + U_1 - W_1}}{(m+1)W_0W_1}$$

$$\frac{\sum_{k=1}^{m} \frac{mU_0 - kU_0 + kW_0}{mU_0 - kU_0 + kW_0 + U_0 - W_0} \frac{mU_1 - kU_1 + kW_1}{mU_1 - kU_1 + kW_1 + U_1 - W_1}}{m+1}$$

4.4.2 Aceleração de cálculos e *look-up table*

A princípio as equações do modelo envolvem várias operações de soma e multiplicação. As equações foram expandidas ao máximo para, primeiro, evitar qualquer erro de arredondamento que possa se propagar através do uso de várias equações (além disso, a aritmética inteira é mais rápida) e, segundo, para agregar constantes e reduzir o número de operações necessárias. Por exemplo, sejam $U = (320, 240)$, $W = (32, 24)$ e $m = 4$ constantes, então a equação da proposicão 3.1.6, para o primeiro componente, pode ser reescrita como:

$$\omega_{k,j,0}(p) = \frac{(k-j)(9216 + 64F_i) + p_i(2560 - 704k)}{2560 - 704j} \tag{4.7}$$

Look-up tables são estruturas de dados que permitem acelerar procedimentos substituindo cálculos por uma busca por valores pre-calculados na memória, uma vez que buscar um valor na memória pode ser menos custoso do que o procedimento realizado via cálculos. Entretanto, o espaço para uma equação pode exigir grandes quantidades de memória. Pode-se, porém, escolher um subconjunto de variáveis não constantes para serem livres. Por exemplo, para a equação acima podemos considerar todo o espaço $k \times j \subset \mathbb{N}^2$, que é um espaço com $m(m-1)$ elementos. Sendo, portanto, $t = (k, j)$, teremos:

$$\omega_{k,j,0}(p) = \frac{A + BF_i + Cp_i}{D} \tag{4.8}$$

onde A, B, C e D são constantes. Assim reduzimos 5 somas, 5 produtos e 1 divisão por 2 somas, 2 produtos e 1 divisão com o custo de 6 referências de memória (2 referências para acessar 4 constantes).

Capítulo 5

Experimentos e Resultados

Para validar o modelo proposto, foram realizados experimentos envolvendo desempenho, atenção, reconhecimento e transmissão de vídeo. Lembrando que o que está sendo testado é o desempenho do modelo proposto em termos de tempo, não os algoritmos clássicos de visão em si. O sistema utilizado como experimento para validação do modelo proposto é constituído por 4 etapas:

1. Captura das imagens: processo de captura a partir de um dispositivo de captura de vídeo;
2. Pré-processamento: possíveis correções, como por exemplo, distorção geométrica ou equalização;
3. Extração de características:
   - filtros (gaussiana, Sobel, laplaciano, por exemplo);
   - análise de movimento (fluxo óptico, por exemplo);
   - cálculo de disparidade estéreo.
4. Tarefas de alto nível: reconhecimento de objetos e rastreamento.

Todos os experimentos foram realizados em um Notebook AMD Turion Dual Core 2GHz com 2 GBytes de memória RAM. Os programas foram codificados integralmente em linguagem C utilizando a biblioteca OpenCV [Bradski & Kaehler 2008] destinada à visão computacional. As imagens utilizadas foram obtidas através de duas webcameras Creative NX Ultra conectadas via USB.

5.1 Experimento de estímulos bottom-up

Utilizamos o mapa de saliências baseado no artigo de Butko et al [Butko et al. 2008] e implementado no *toolbox* NMPT [Butko 2008]. A Figura 5.1 contém na coluna esquerda 3 quadros de um vídeo capturado e na coluna direita os respectivos mapa de saliências. O mapa de saliências foi calculado sobre o primeiro nível, isto é, o nível de menor resolução, no modelo de multi-resolução com fóvea móvel. A média do tempo obtido para calcular o mapa de saliências foi 6,340ms e 2,131 de desvio padrão para $W = (64, 48)$.

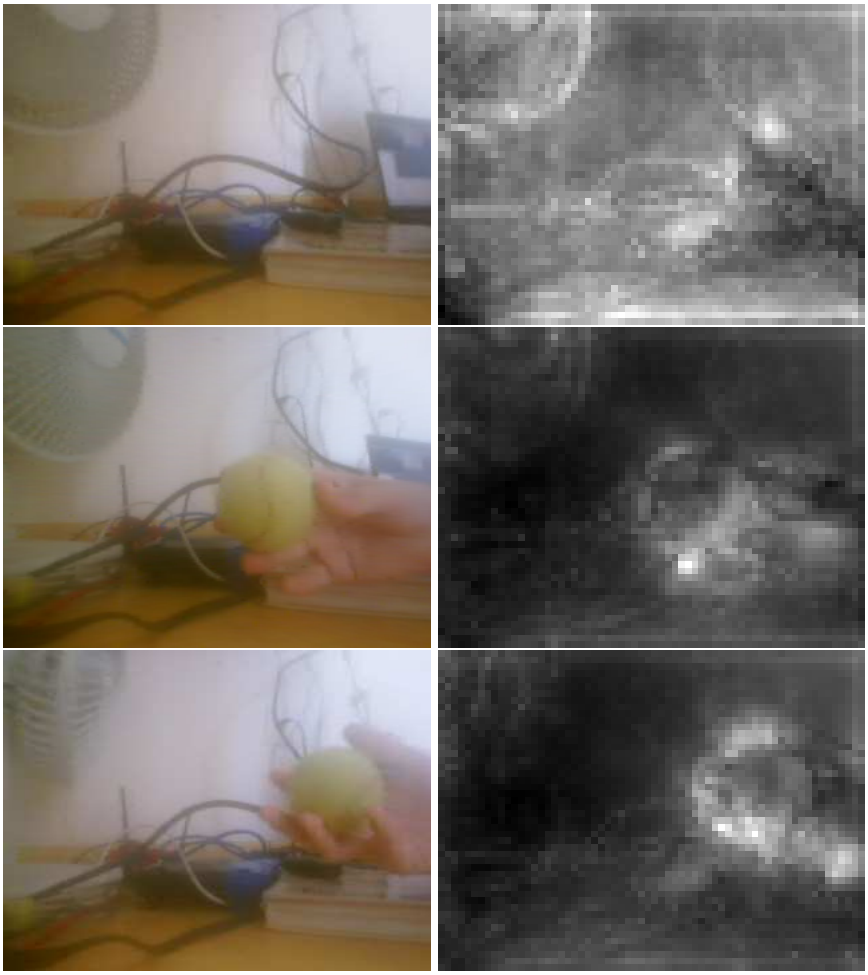

Figura 5.1: Exemplo de mapa de saliências (escala 50% para a imagem original e 250% para o mapa de saliências)

5.2 Experimentos de desempenho

O custo computacional para obter um nível é simplesmente o custo de redimensionar a imagem original I para este nível. Como a obtenção de um nível é independente de outro nível, temos que o custo total é o custo de um nível vezes o número de níveis. Dessa forma, os custos para um número de níveis diferente é facilmente estimável e, portanto, não fazem parte desse experimento. As variáveis que teoricamente influenciam no tempo de cálculo de cada nível do modelo são: W e o operador de redimensionamento Φ. Dessa forma, os experimentos de desempenho foram divididos em 2, cada qual variando apenas uma das variáveis.

5.2.1 Variando W

Para esse teste foram escolhidos $U = (1920, 1440)$, 4 níveis e interpolação bilinear. A Tabela 5.1 contém os resultados obtidos.

W	Tempo médio (desvio padrão)
32x24	0,216ms (0,082)
64x48	0,579ms (0,138)
96x72	1,362ms (0,239)
128x96	1,986ms (0,250)
160x120	3,133ms (0,548)
320x240	12,810ms (0,934)

Tabela 5.1: Desempenho para construir a multi-resolução com fóvea móvel, variando W.

5.2.2 Variando Φ

Para esse teste foram escolhidos $U = (640, 480)$, 4 níveis e $W = (64, 48)$. A Tabela 5.2 contém os resultados obtidos.

Método de interpolação no OpenCV	Tempo médio (desvio padrão)
CV_INTER_NN	0,157ms (0,065)
CV_INTER_LINEAR	0,614ms (0,095)
CV_INTER_CUBIC	2,668ms (0,303)
CV_INTER_AREA	3,740ms (1,190)

Tabela 5.2: Desempenho para construir a multi-resolução com fóvea móvel, variando Φ.

5.3 Experimento de estímulo top-down

Nos experimentos de rastreamento (*tracking*), uma mão segurando uma bola aparece em frente às câmeras. Um usuário aponta e clica, através da interface gráfica, a posição inicial da fóvea no processo de rastreamento. Essa posição inicial deve ser de preferência no centro da bola. O sistema então deve rastrear a bola apenas mudando a posição da fóvea. Quando a bola está perto de sair do campo de visão, o sistema alerta com uma solicitação de movimentação física dos dispositivos. Um exemplo de rastreamento pode ser visto na Figura 5.2.

Através da abordagem com fóvea móvel, é possível mudar a atenção de uma região para outra de um quadro para o quadro seguinte em tempo real. Se a movimentação dos dispositivos físicos for necessária a toda hora que a região de interesse muda, isso levaria cerca de 500 ms [Gonçalves et al. 2000].

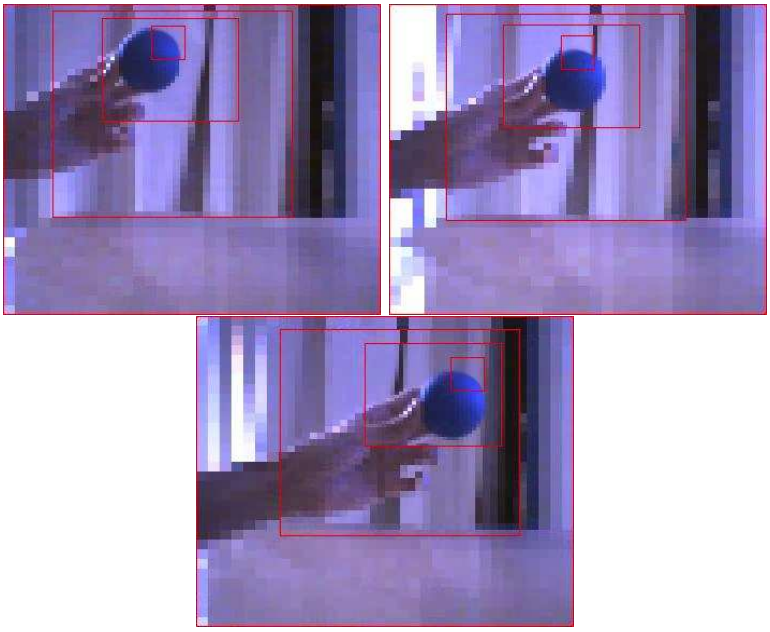

Figura 5.2: Perseguindo uma bola usando fóvea móvel.

5.4 Experimento de filtragem

Realizamos um experimento com a aplicação de 4 filtros em uma imagem 320×240 e em uma estrutura de multi-resolução com fóvea móvel formada por 4 níveis, tal que $W = (64, 48)$. Os filtros usados foram: gradiente (em x e em y), gaussiana (máscara 3x3) e laplaciano da gaussiana. A Tabela 5.3 contém os resultados obtidos tanto na imagem original como na estrutura de multi-resolução proposta. Vale lembrar que todos os testes foram realizados com as imagens em tons de cinza.

	Tempo médio (desvio padrão)
Imagem original	6,602ms (1,619)
Multi-resolução	1,264ms (0,614)

Tabela 5.3: Tempos obtidos na filtragem.

Com a redução do número de pixels na estrutura de multi-resolução, como esperado, o tempo foi bastante reduzido. A diferença entre os tempos pode ser maior ou menor dependendo das variáveis da estrutura e do tamanho da imagem original. O mais importante para ressaltar nesse experimento é que aplicar diversos filtros na imagem original pode demandar mais tempo do que se tem disponível, ainda mais considerando o processamento de imagens coloridas e para duas câmeras. Vale lembrar que a idéia não é somente calcular os filtros, mas também extrair diversas características (*features*) da imagem e realizar processamento de alto nível sobre estas.

5.5 Experimento de disparidade estéreo

Para analisar adequadamente os resultados dos cálculos de disparidade, usamos os resultados verdadeiros (*ground truth*) das imagens de Tsukuba (Figura 5.3), dos Cones (Figura 5.4), do Teddy (Figura 5.5) e de Venus (Figura 5.6) disponibilizados por Scharstein e Szeliski [Scharstein & Szeliski 2007] com tamanhos 384×288, 450×375, 450×375 e 434×383, respectivamente.

Figura 5.3: Image esquerda, direita e *ground truth* de Tsukuba

Figura 5.4: Image esquerda, direita e *ground truth* dos Cones

Figura 5.5: Image esquerda, direita e *ground truth* do Teddy

Utilizamos dois algoritmos para cálculo de disparidade estéreo: método de correlação com SSD (*sum of squared difference*) e *graph-cut*. Realizamos os testes para os dois

Figura 5.6: Image esquerda, direita e *ground truth* de Venus

métodos, para W sendo $(32, 24)$ ou $(64, 48)$ ou $(96, 72)$ e usando 4 níveis. A fóvea foi posicionada manualmente em ambas imagens de forma a apontarem para o mesmo ponto da imagem. Os níveis obtidos para cada um dos exemplos podem ser vistos na Figura 5.7. Em seguida, os algoritmos foram aplicados em cada um dos níveis, normalizados pela equação 4.2 e reconstruídos para o tamanho da imagem original. Os algoritmos também foram executados sobre a imagem original.

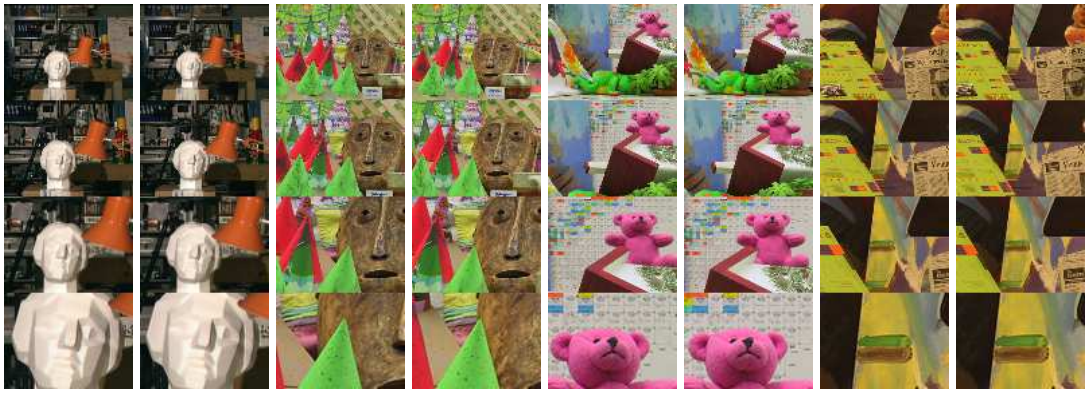

Figura 5.7: Níveis (esquerda e direita) para as imagens de Tsukuba, dos Cones, do Teddy e de Venus, respectivamente

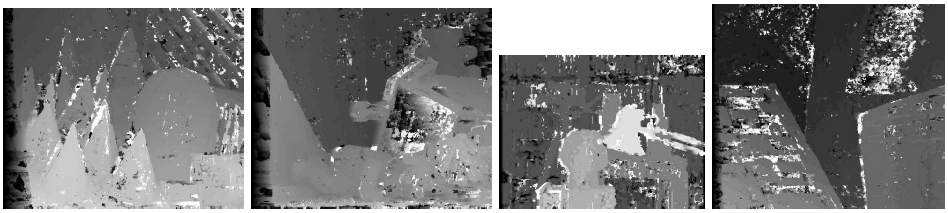

Figura 5.8: Resultado das disparidades na imagem original usando o método SSD

Parâmetros	**Tempo médio**	**Figura**
SSD na imagem original	1152ms	5.8
GC na imagem original	17203ms	5.9
SSD e $W = (32, 24)$	2,8ms	5.10
SSD e $W = (64, 48)$	12,0ms	5.11
SSD e $W = (96, 72)$	44,2ms	5.12
GC e $W = (32, 24)$	50,3ms	5.13
GC e $W = (64, 48)$	76,6ms	5.14
GC e $W = (96, 72)$	182,4ms	5.15

Tabela 5.4: Tempos obtidos no cálculo de disparidade estéreo.

Figura 5.9: Resultado das disparidades na imagem original usando o método *graph-cut*

Figura 5.10: Resultado das disparidades para $W = (32, 24)$ usando o método SSD

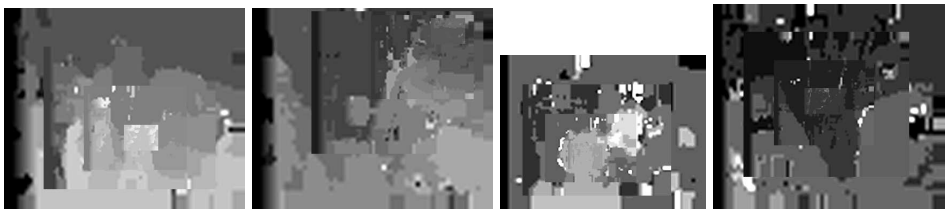

Figura 5.11: Resultado das disparidades para $W = (64, 48)$ usando o método SSD

5.6 Testando a estrutura MRMC com reconhecimento

Em outro experimento envolvendo reconhecimento, três objetos, uma bola de tênis, uma bola de borracha azul e um dominó foram apresentados em diversas posições ao sistema. Com o sistema visual já em execução, cerca de 35 imagens foram tiradas para

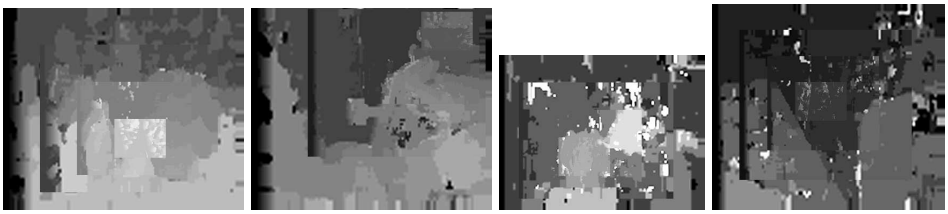

Figura 5.12: Resultado das disparidades para $W = (96, 72)$ usando o método SSD

Figura 5.13: Resultado das disparidades para $W = (32, 24)$ usando o método *Graph-cut*

Figura 5.14: Resultado das disparidades para $W = (64, 48)$ usando o método *Graph-cut*

Figura 5.15: Resultado das disparidades para $W = (96, 72)$ usando o método *Graph-cut*

as bolas e outras 35 para o dominó. As bordas (obtidos através do filtro gradiente) foram usadas para o treinamento de uma rede neural. No experimento, o usuário apertava a tecla B caso o objeto fosse uma bola e a tecla D caso fosse um dominó, o que dava ao conjunto de treinamento o valor real de cada caso de teste. Em seguida, o usuário solicitava o treinamento da rede neural que utilizava 1300 iterações. Os mesmos objetos foram apresentados em seguida. O resultado foi cerca de 70% de identificações positivas para a bola e cerca de 85% para o dominó, mesmo que o objeto tenha sido apresentado na região periférica do campo de visão das câmeras.

As 3 imagens capturadas e seus respectivos gradientes para treinamento da rede neural podem ser vistos na Figura 5.16.

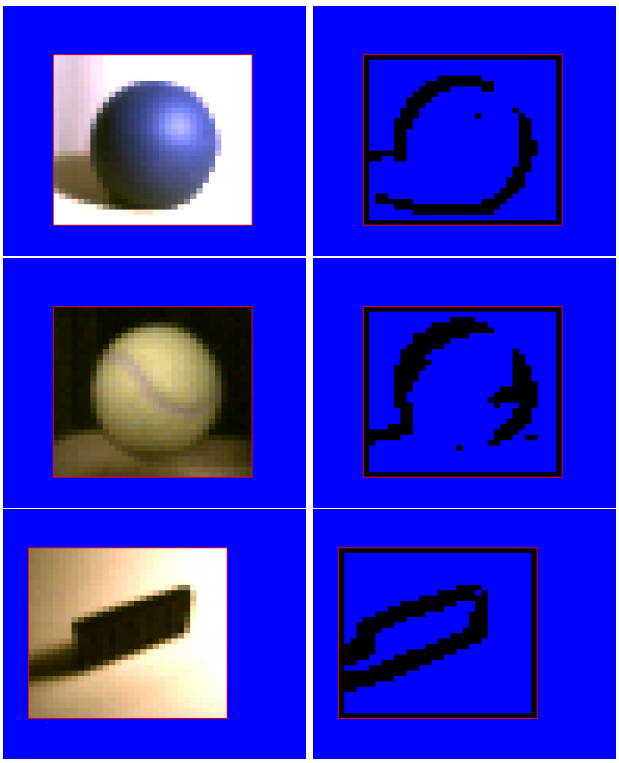

Figura 5.16: À esquerda, um dos níveis reconstruído e à direita o resultado da aplicação do filtro gradiente nesse nível

Este trabalho propõe um útil mecanismo para redução de dados utilizando um modelo de multi-resolução utilizando fóvea móvel que pode ser integrado e testado em controle de atenção, reconhecimento de objetos e navegação. Parte da implementação já está realizada para o robô galateia, presente no laboratório Natalnet/DCA.

5.7 Outras aplicações

O modelo proposto tem como principal motivação acelerar a visão de baixo nível, provendo informações visuais em tempo real no contexto de um sistema de visão robótico, onde a demanda por processamento é alta. Entretanto, o modelo também pode servir para outras aplicações, entre elas: a transmissão de vídeo e localização de objetos, este último abordado por Butko et al [Butko et al. 2008].

5.7.1 Transmissão de vídeo

Foram realizados dois experimentos para transmissão de vídeo. O intuito desses experimentos foi o de reduzir a quantidade de informações de vídeo que precisa ser repas-

sada de um servidor para um cliente (por exemplo, via internet). Em um primeiro experimento, um usuário começa visualiza um vídeo transmitido via internet e pode a qualquer momento indicar com o mouse a posição da fóvea, o que chamamos de visualização ativa. Em um segundo experimento, o servidor determina a posição da fóvea e o usuário não precisa efetuar nenhuma intervenção, o que chamamos de visualização passiva. Para testar a redução de banda necessária para transferir vídeo, codificamos o vídeo original e os níveis do modelo no formato MPEG-4 e traçamos uma comparação de tamanho.

Quanto à escolha das variáveis, o ideal é que o servidor decida quais devam ser utilizadas, uma vez que este detém as informações de vídeo e possui maior capacidade de escolher o que é melhor para cada situação.

O custo adicional para transferir informações sobre o modelo é relativamente desprezível. Mesmo que todos os parâmetros necessários (W, U, F, m) sejam repassados a cada quadro, supondo que cada inteiro ocupe 4 bytes, isso corresponde a apenas 28 bytes por quadro.

A redução do tamanho do vídeo não significa necessariamente que o vídeo seja codificado em menos bytes no caso de codecs que utilizam informações de movimento pois nada pode-se afirmar sobre a mudança de intensidade dos pixels em função do tempo. Assim, a estrutura de multi-resolução com fóvea móvel pode ajudar na codificação do movimento, pois cada nível pode permanecer aproximadamente constante. Por outro lado, a fóvea ao mover-se pode acrescentar grande oscilações na intensidade dos pixels, mesmo embora esses pixels possam estar sem oscilações na imagem original. Neste caso, o modelo pode acrescentar demasiadamente informações espaço temporal, acarretando em um maior número de bytes para codificar o vídeo.

Há duas formas como o vídeo pode ser codificado para ser enviado entre um cliente e um servidor. Em uma primeira forma (vide Figura 5.17), o servidor codifica o quadro com todos os níveis organizados de acordo com um *layout*. Este *layout* pode ser simplesmente os níveis em forma de fila, em forma de pilha ou um *layout* que possa ser codificado com maior compactação por um *codec*. Na segunda forma (vide Figura 5.18), o servidor codifica cada nível independentemente do outro através de um multiplexador. O cliente, por sua vez, utiliza um demultiplexador para separar cada nível e reconstruir a imagem original. Observe que essa segunda forma permite que possamos codificar cada nível de uma forma diferente. Assim, podemos utilizar diferentes parâmetros de codificação para diferentes níveis. Por exemplo, podemos utilizar *bitrates* menores para níveis de menor resolução e *bitrates* mais altos para as melhores resoluções, ou mesmo diferentes taxas de atualização para cada nível. Além disso, devido à possibilidade de facilmente dividir o fluxo de dados em vários fluxos, cada um representando um nível, não precisamos entender como o *codec* funciona, mas apenas utilizá-lo como um sistema caixa-preta com diferentes parâmetros para cada fluxo de dado.

Observe que o serviço pode ser escalável se os clientes estiverem usando visualização passiva, pois apenas uma estrutura de multi-resolução com fóvea móvel precisa ser calculada e ou se o servidor pré-calcular a estrutura para todas as posições de fóvea possível (ou um subconjunto), o que demanda uma maior quantidade de memória e processamento.

Para o experimento de transmissão de vídeo foi utilizado o vídeo Elephants Dream [Foundation & Institute 2009] cuja resolução é 1920×1080. Selecionamos 96 frames

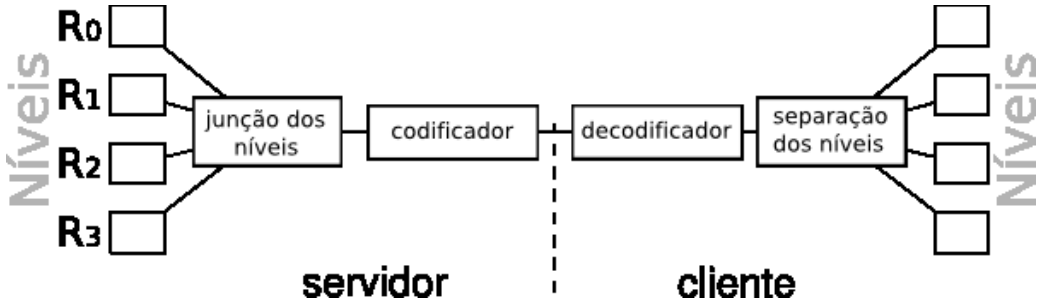

Figura 5.17: Modo de transmissão de vídeo onde o vídeo é codificado integrando todos os níveis

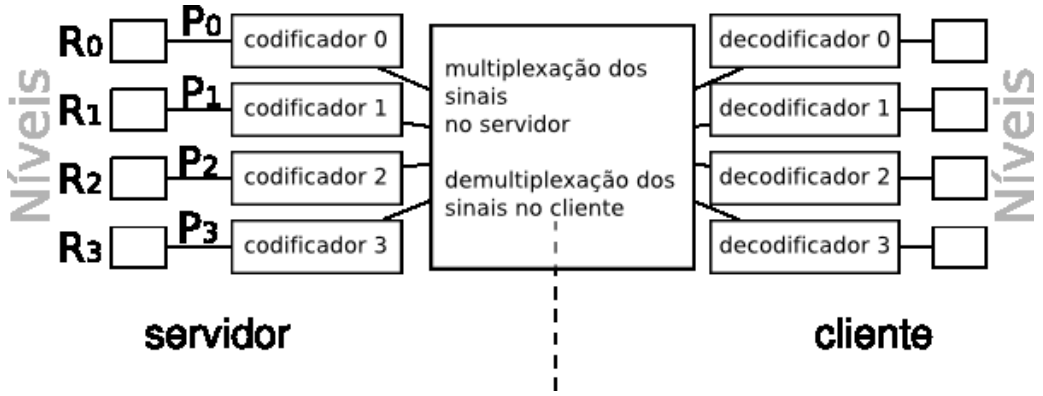

Figura 5.18: Modo de transmissão de vídeo onde cada nível corresponde a um vídeo diferente e parâmetros de codificação específicos (parâmetro p_k para o nível R_k)

do vídeo original para compor um vídeo a ser transmitido (vide Quadro 71 na Figura 5.19). Compactamos este novo vídeo através do *codec* MPEG4 utilizando a ferramenta *mencoder* a uma taxa média de 2040kbps, resultando em um vídeo de 1003kB (vide Quadro 71 na Figura 5.20). Aplicamos o modelo de multi-resolução com fóvea móvel com 4 níveis, tal que $W = (224,168)$, na transmissão onde a fóvea era controlada com o mouse pelo cliente e codificamos cada nível através do *codec* MPEG4 com diferentes parâmetros de codificação (vide os níveis na Figura 5.22). Observe que preenchemos um retângulo preto em cada nível k a região que pertence ao nível $k+1$, resultando em uma maior compactação. A Tabela 5.5 contém a taxa média de transmissão para cada um dos 4 níveis e o respectivo tamanho do vídeo. Os 4 vídeos juntos totalizam 338kB (33% do vídeo original compactado). O quadro 71 restaurado a partir dos 4 vídeos pode ser visto na Figura 5.21. A Figura 5.23 mostra a mesma região ao redor da fóvea (correspondente a A_3) no vídeo compactado e no vídeo compactado pela multi-resolução com fóvea móvel.

Nível	Taxa média	Tamanho do vídeo
0	47,9kbps	36kB
1	99,8kbps	66kB
2	152,4kbps	97kB
3	270,4kbps	139kB

Tabela 5.5: Taxas e tamanhos do arquivo para cada nível.

Figura 5.19: Quadro 71 original (escala = 23%)

5.7.2 Localização de objetos

O presente trabalho também foi utilizado por um método denominado MI-POMDP no contexto de detecção de objetos [Butko & Movellan 2009]. Basicamente, o paralelepípedo de multi-resolução com fóvea móvel é utilizado para acelerar a detecção de um objeto na imagem deslocando-se a fóvea pela imagem e realizando uma busca em cada nível do modelo. Dada uma fixação da fóvea na imagem, a fóvea é movida para o ponto onde há maior probabilidade de o objeto ser encontrado. Com isso, em experimentos realizados é possível reduzir praticamente pela metade o tempo necessário para detectar uma face em uma imagem em relação ao algoritmo de Viola Jones, com uma pequena perda no erro médio [Butko & Movellan 2009]. A Figura 5.24 contém imagens de várias fixações da fóvea na detecção de faces de uma imagem e a Tabela 5.7.2 contém os resultados do experimento realizado [Butko & Movellan 2009].

Figura 5.20: Quadro 71 compactado (escala = 23%)

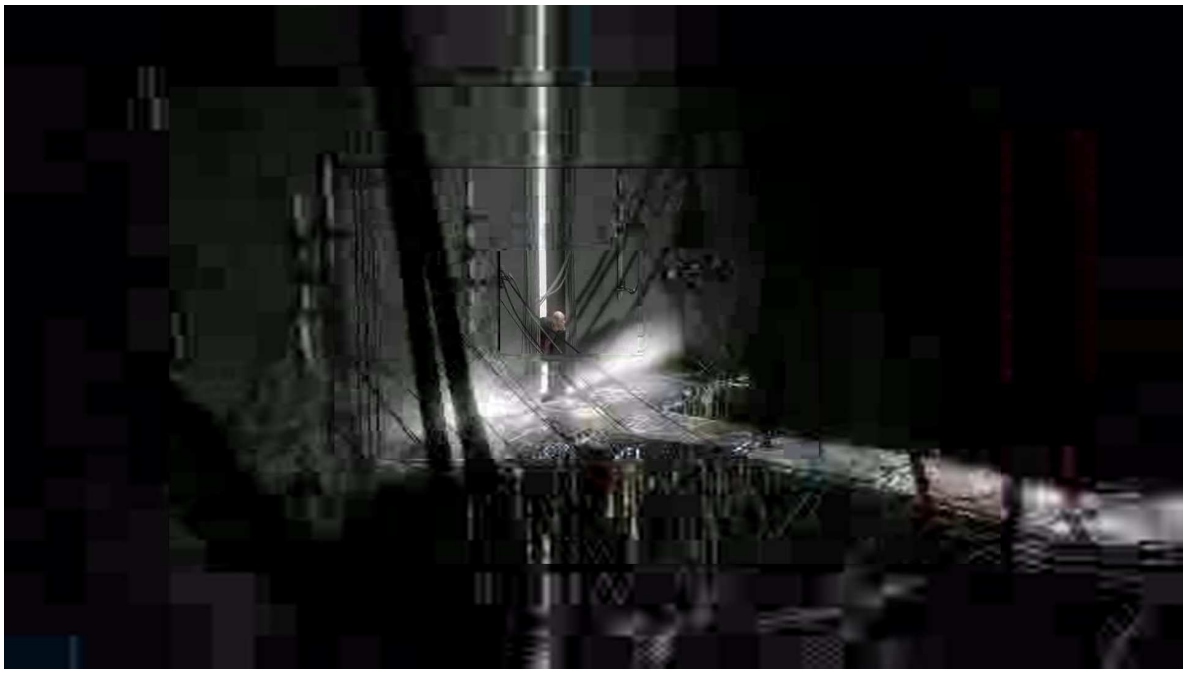

Figura 5.21: Quadro 71 restaurado (escala = 23%)

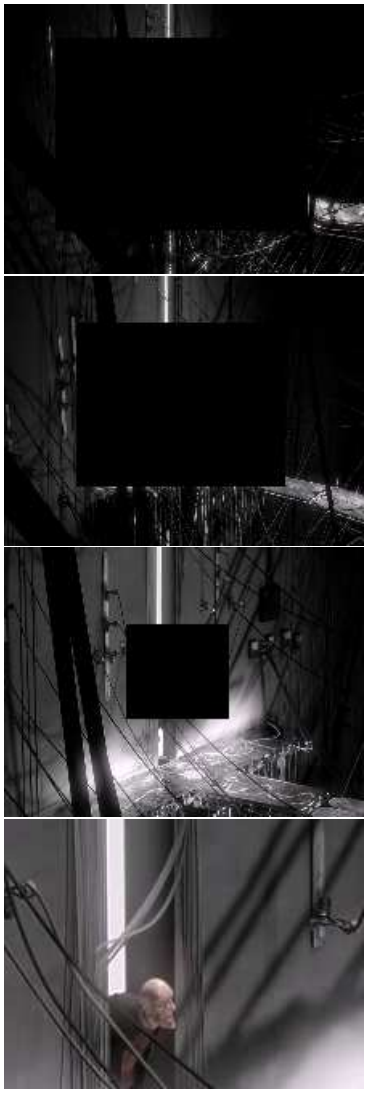

Figura 5.22: Níveis (escala = 60%)

Medida	MI-POMDP	Viola Jones
Tempo (ms)	37.9	73.4
Erro	1.59	1.26

Tabela 5.6: Resultados obtidos na detecção de faces

Figura 5.23: Níveis (escala = 120%)

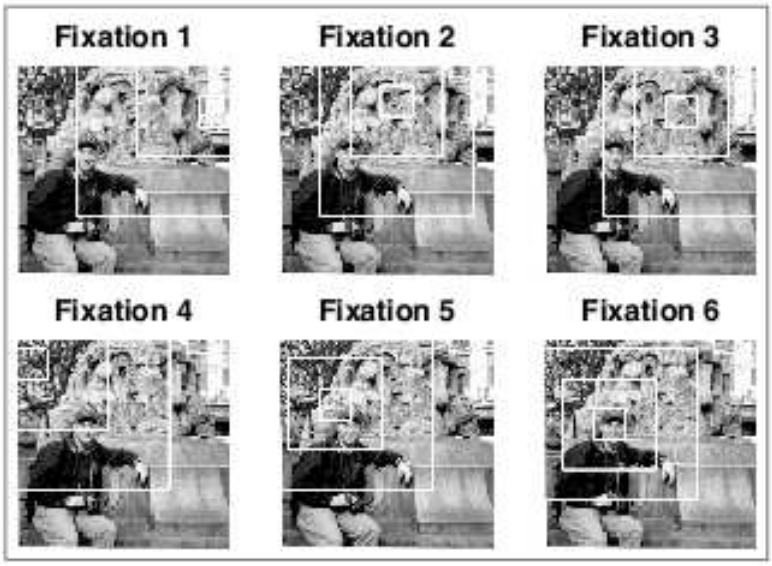

Figura 5.24: Fixações para detecção de faces.

Capítulo 6

Conclusões e trabalhos futuros

Neste trabalho, propomos um mecanismo para redução de dados utilizando um modelo de multi-resolução combinado com fóvea móvel que pôde ser integrado e testado em tarefas envolvendo controle da atenção visual, reconhecimento de objetos entre outras envolvendo visão estéreo ativa. Para isso, o primeiro passo é determinar a posição na qual a fóvea deva estar. O vetor fóvea pode ser determinado através de estímulos *bottom-up* ou *top-down*. Isso pode ser feito, por exemplo, utilizando o primeiro nível de resolução, onde a fóvea pode ser rastreada e o correspondente na outra imagem pode ser facilmente obtido por correlação. Uma estrutura em multi-resolução utilizando fóvea móvel é então construída em tempo real (cerca de 0.5ms em um Notebook 2 GHz, para 4 imagens com tamanho 64x48 pixels). Em seguida, técnicas de visão computacional, tais como a reconstrução estéreo (*shape from stereo*) aplicada para calcular um mapa de disparidade, e outras características úteis a processos de mais alto nível podem ser calculadas. Isto pode ser feito ainda em tempo real, pois a quantidade de dados a serem processados é irrisória usando a estrutura em multi-resolução.

Utilizando o modelo proposto, nós testamos comportamentos que obtiveram desempenho em tempo real devido à redução de dados. Além disso, a representação de fóvea móvel proposta permitiu efetuar tarefas como o controle de atenção visual em tempo real, podendo ser aplicado para acelerar algumas tarefas.

Então, em nosso trabalho, introduzimos duas contribuições científicas principais. A primeira é o aprimoramento feito a partir de um modelo de redução de dados e abstração de características (MRMC), com a sua melhor formalização matemática, possibilitando uma otimização da sua implementação. A segunda contribuição é a introdução do modelo de fóvea móvel. Experimentos em atenção e reconhecimento foram realizados. Baseado na robustez verificada nesses experimentos e na melhoria substancial introduzida na eficiência do sistema visual, a metodologia proposta pode ser usada em outros processos de alto nível, ou em outras tarefas, tais como, por exemplo, navegação, reconhecimento, busca e manipulação de objetos por sistemas robóticos. Tais contribuições, resultado direto deste trabalho, foram apresentadas em congressos importantes (ICRA 2008) [Gomes et al. 2008] e também tivemos recentemente um Capítulo de livro aceito para publicação [Gomes, Goncalves, Gardiman & Leite 2009]. Pudemos também utilizar resultados conseguidos com a detecção de movimento em um artigo em periódico relevante, em tema relacionado ao desta dissertação [Gomes, Oliveira, Britto-Neto, Santos, Andrade, Carvalho & Goncalves 2009].

A possibilidade de mudar o foco da atenção é a base não somente para as tarefas descritas, mas também para outras tarefas mais complexas envolvidas em cognição robótica [Gonçalves et al. 2000]. O modelo proposto aprimora uma metodologia de certa forma inspirada em modelos biológicos no sentido de que os níveis de resolução mais precisos são localizados no centro da imagem. Dessa forma, os níveis de menor resolução podem ser usados, por exemplo, para detectar movimento ou características a serem usadas em tarefas de navegação e os níveis de maior resolução podem ser aplicados para tarefas envolvendo maior necessidade de detalhes, como reconhecimento, leitura de texto ou manipulação de objetos. Uma tarefa de busca de um determinado padrão em uma imagem pode utilizar a combinação de um ou mais níveis. Obviamente, nesse caso, a fóvea móvel é importante ao evitar que a cabeça (robótica) tenha que fazer movimentos desnecessários.

O modelo também mostra-se útil na transmissão de vídeo, principalmente em ambientes onde a largura de banda restringe a quantidade de dados que podem ser transmitidas. Uma alta compressão pode resultar em uma perda uniforme no vídeo, degradando a qualidade na região de interesse. Nesse sentido, os *codecs* podem atuar como caixa-preta, evitando que o desenvolvedor tenha que entender o funcionamento destes para realizar a compressão através da multi-resolução com fóvea móvel. Entretanto, dado que o desenvolvedor tenha acesso ao funcionamento do *codecs*, acreditamos que o modelo pode ser aplicado para obter taxas de compressão maiores que os obtidos nos experimentos, umas vez que não há nenhuma compactação inter-níveis.

Figura 6.1: Robô Galatéia

Atualmente, uma cabeça estéreo (vide Figura 6.2) está sendo construída e será integrada ao robô Galatéia (vide Figura 6.1). Este robô possui controles para movimentação e rotação, além de sonares e um PC embarcado. Futuramente, este robô pode utilizar o sistema de visão estéreo para ajudar na navegação, reconhecimento de pessoas e outras tarefas, isto é, torná-lo um robô autônomo. Acreditamos que com o processamento das imagens em tempo real, esta autonomia esteja mais próxima.

A interação entre a determinação do vetor fóvea e o controle físico das câmeras é outra questão a ser abordada.

Figura 6.2: Cabeça estéreo

Referências Bibliográficas

Batista, J., P. Peixoto & H. Araújo (1997), Real-time visual behaviors with a binocular active vision system, *em* 'Proceedings of 1997 IEEE International Conference on Robotics and Automation', Vol. 4, pp. 3391–3396 vol.4.

Batista, J., P. Peixoto & H. Araujo (2000), Binocular tracking and accommodation controlled by retinal motion flow, *em* 'Pattern Recognition, 2000. Proceedings. 15th International Conference on', Vol. 1, pp. 171–174.

Bernardino, Alexandre & José Santos-Victor (2002), A binocular stereo algorithm for log-polar foveated systems, *em* 'Proceedings of the Second International Workshop on Biologically Motivated Computer Vision', Springer-Verlag, London, UK, pp. 127–136.

Boyling, T. & J. Siebert (2000), A fast foveated stereo matcher, *em* 'Proceedings of Conference. on Imaging Science Systems and Technology', Las Vegas, USA, pp. 417–423.

Boyling, T.A.;Siebert, J.P.; (2004), Foveated vision for space-variant scene reconstruction, *em* 'Proceedings of the 35th International Symposium on Robotics, ISR2004', pp. 1–6.

Bradski, Gary & Adrian Kaehler (2008), *Learning OpenCV*, O'Reilly, Sebastopol, CA, USA.

Burt, P. & E. Adelson (1983), 'The laplacian pyramid as a compact image code', *IEEE Transactions on Communications* **31**(4), 532–540.

Butko, Nicholas J. (2008), 'Nicks machine perception toolbox'.
URL: *http://mplab.ucsd.edu/nick/NMPT*

Butko, Nicholas J. & Javier R. Movellan (2009), Optimal scanning for faster object detection, *em* 'IEEE International Conference on Computer Vision and Pattern Recognition'.

Butko, N.J., Lingyun Zhang, G.W. Cottrell & J.R. Movellan (2008), Visual saliency model for robot cameras, *em* 'IEEE International Conference on Robotics and Automation', pp. 2398–2403.

Chang, Ee-Chien & Chee K. Yap (1997), A wavelet approach to foveating images, *em* 'Proceedings of the thirteenth annual symposium on Computational geometry', ACM, New York, NY, USA, pp. 397–399.

Churchland, Patricia S. & Terrence J. Sejnowski (1992), *The Computational Brain*, MIT Press.

Desimone, R. & J. Duncan (1995), 'Neural mechanisms of selective visual attention', *Annual Reviews of Neuroscience* **18**, 193–222.

Foundation, Blender & Netherlands Media Art Institute (2009), 'Elephants dream'. **URL:** *http://www.elephantsdream.org/*

Gomes, Herman Martins & Robert Fisher (2001), Learning and extracting primal-sketch features in a log-polar image representation, *em* 'Proc. of Brazilian Symp. on Comp. Graphics and Image Processing', IEEE Computer Society, pp. 338–345.

Gomes, R. B., L. M. G. Goncalves & B. M. Carvalho (2008), Real time vision for robotics using a moving fovea approach with multi resolution, *em* 'IEEE International Conference on Robotics and Automation', pp. 2404–2409.

Gomes, Rafael B., L. M. G. Goncalves, Renato Q. Gardiman & Luiz E. C. Leite (2009), *Robot Vision*, IN-TECH, Vukovar, Croácia.

Gomes, Rafael B., Lucas M. Oliveira, Laurindo S. Britto-Neto, Tiago S. Santos, Gilbran S. Andrade, Bruno M. Carvalho & L. M. G. Goncalves (2009), 'Producing stylized videos using the animvideo rendering tool', *Int. J. Imaging Syst. Technol.* **19**(2), 100–110.

Gonçalves, Luiz M. G., Roderic A. Grupen, Antonio A. Oliveira, David Wheeler & Andrew Fagg (2000), 'Tracing patterns and attention: Humanoid robot cognition', *The Intelligent Systems and their Applications* **15**(4), 70–77.

Gonzales, Rafael C. & Richard E. Woods (1992), *Digital Image Processing*, Addison-Wesley Publication Company.

Horn, Berthold K. P (1986), *Robot Vision*, MIT Press.

Horn, Berthold K. P. & Brian G. Schunck (1992), 'Determining optical flow', pp. 389–407.

Koch, C. & S. Ullman (1985), 'Shifts in selective visual attention: Towards the underlying neural circuitry', *Human Neurobiology* **4**, 219–227.

Kortum, Philip & Wilson S. Geisler (1996), Implementation of a foveated image coding system for image bandwidth reduction, *em* 'Proceedings of SPIE', Vol. 2657, pp. 350–360.

Marr, D. (1982), *Vision – A Computational Investigation into the Human Representation and Processing of Visual Information*, The MIT Press, Cambridge, MA.

Marr, D. (1993), *Active Perception: Advances in Computer Vision Series*, Vol. 1, Lawrence Erlbaum Associates, New York, NY.

Nishihara, K., H. J. Thomas & E. Huber (1984), Real-time tracking of people using stereo and motion, Ai lab technical report, Massachusetts Institute of Technology.

Olshausen, B., C. Anderson & D. Van Essen (1964), 'A neurobiological model of visual attention and invariant pattern recognition based on dynamic routing of information.', *Journal of Neuroscience* **13**(1), 4700–4719.

Rao, Rajesh P. N. & Dana H. Ballard (2004), 'Probabilistic models of attention based on iconic representations and predictive coding', *Neurobiology of Attention* .

Sandon, P. (1990), 'Simulating visual attention.', *Journal of Cognitive Neuroscience* **2**, 213–231.

Scharstein, D. & R. Szeliski (2007), 'Stereo vision research page'.
URL: *http://cat.middlebury.edu/stereo/*

Scharstein, Daniel & Richard Szeliski (2002), 'A taxonomy and evaluation of dense two-frame stereo correspondence algorithms'.

Schwartz, Eric L., Douglas N. Greve & Giorgio Bonmassar (1995), 'Space-variant active vision: definition, overview and examples', *Neural Networks* **8**(7-8), 1297–1308.

Segundo, Sávio & Luiz M. G. Gonçalves (2004), Redução e abstração de dados no projeto robosense, *em* 'X Brazilian Symposium of Computer Graphic and Image Processing - WTDCGPI'.

Thompson, S. & S. Kagami (2005), Humanoid robot localisation using stereo vision, *em* 'Humanoid Robots, 2005 5th IEEE-RAS International Conference on', pp. 19–25.

Tortora, Gerard J. (2000), *Corpo humano: fundamentos de anatomia e fisiologia*, ArtMed.

Treisman, A. (1964), 'Selective attention in man', *British Medical Bulletin* .

Treisman, Anne (1985), 'Preattentive processing in vision', *Comput. Vision Graph. Image Process.* **31**(2), 156–177.

Trucco, Emanuele & Alessandro Verri (1998), *Introductory Techniques for 3-D Computer Vision*, Prentice Hall PTR, Upper Saddle River, NJ, USA.

Tsai, R. (1987), 'A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses', *IEEE Journal of Robotics and Automation* **3**(4), 323–344.

Tsotsos, John K. (1987), ‘Knowledge organization and its role in representation and interpretation for time-varying data: the alven system’, pp. 498–514.

Uhr, Leonard (1972), Layered ‘recognition cone’ networks that preprocess, classify and describe, *em* ‘IEEE Transactions on Computers’, pp. 758–768.

Wessler, Mike. (1996), A modular visual tracking system, Ai lab technical report, Massachusetts Institute of Technology.

Witkin, A. P (1983), ‘Scale-space filtering’, *Proc. 8th International Joint Conference on Artificial Intelligence* **1**(1), 1019–1022.

Wu, H. R. & K. R. Rao (2005), *Digital Video Image Quality and Perceptual Coding (Signal Processing and Communications)*, CRC Press, Inc., Boca Raton, FL, USA.

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo