Universidade do Estado da Bahia

Departamento de Ciências Exatas e da Terra

Colegiado de Sistemas de Informação

Um Sistema para Videomonitoramento não Invasivo de Idosos que Moram ou Ficam Sozinhos

Bárbara Aniele Santos de Jesus

Salvador-BA

2015

Bárbara Aniele Santos de Jesus

Um Sistema para Videomonitoramento não Invasivo de Idosos que Moram ou Ficam Sozinhos

Monografia submetida ao Colegiado de Sistemas de Informação da Universidade do Estado da Bahia como parte dos requisitos necessários para obtenção do grau de Bacharel em Sistemas de Informação.

Área de Concentração: Sistemas de Informação
Linhas de Pesquisa: Videomonitoramento e Visão Computacional

Cláudio Alves de Amorim
(Orientador)
Manoel Carvalho Marques Neto
(Coorientador)

Salvador-BA
2015

Bárbara Aniele Santos de Jesus

Um Sistema para Videomonitoramento não Invasivo de Idosos que Moram ou Ficam Sozinhos

Monografia submetida ao Colegiado de Sistemas de Informação da Universidade do Estado da Bahia como parte dos requisitos necessários para obtenção do grau de Bacharel em Sistemas de Informação.

Aprovada em: _____ / _____ / _____

Cláudio Alves de Amorim
Doutorado em Educação
Universidade do Estado da Bahia

Eduardo Manuel de Freitas Jorge
Doutorado em Difusão do Conhecimento
Universidade do Estado da Bahia

Manoel Carvalho Marques Neto
Doutorado em Ciência da Computação
Instituto Federal de Educação, Ciência e Tecnologia da Bahia

Resumo

A população brasileira, assim como em outros países, tem se apresentado mais envelhecida. Direcionando os esforços para idosos que moram sozinhos, um modelo que seja capaz de fazer um monitoramento não-invasivo e identificar um comportamento irregular pode contribuir para a redução de uma situação de perigo na sua rotina. Nesse contexto, o presente trabalho, utilizou as técnicas de Visão Computacional para detecção de movimento e reconhecimento de padrões com a finalidade de propor e avaliar um algoritmo capaz de monitorar padrões anômalos de movimentação entre os cômodos em residências de idosos que moram ou ficam sozinhos. Analisando os resultados foi possível detectar e rastrear o movimento das pessoas no vídeo. A biblioteca OpenCV foi usada para auxiliar o desenvolvimento do sistema para validação.

Palavras-chave: Idosos, Reconhecimento de padrões, Sistema de monitoramento.

Abstract

The Brazilian population, as well as in other countries, has been performing more aged. Directing efforts to elderly people living alone, a model that is able to make a non-invasive monitoring and identify an irregular behavior can contribute to the reduction of a hazard in your routine. In this context, the present study, has used the computer vision techniques for motion detection and pattern recognition in order to propose and evaluate an algorithm to monitor anomalous patterns of movement between the rooms in the homes of elderly people who live or are alone. Analyzing the results it was possible to detect and track the movement of people in the video. The OpenCV library was used to assist the development of the system for validation.

Keywords: Elderly people, Pattern recognition, Monitoring system.

Agradecimentos

Primeiramente, agradeço a DEUS, por ter me dado força, saúde e persistência para que eu pudesse concluir mais uma etapa da minha vida.

Agradeço aos meus pais, Mary Anne Souza Santos e Elias Santos de Jesus, por todo amor, carinho e apoio que me deram. Agradeço por terem me dado uma excelente educação e me proporcionado uma formação de qualidade. Muito obrigada por todos os ensinamentos. Agradeço também aos familiares que entenderam a minha ausência em algumas comemorações.

Agradeço também ao meu noivo, Itamar Santos da Silva, pelo companheirismo nessa jornada. Agora já posso começar a pensar em casamento.

Agradeço aos amigos com os quais construí muitas histórias: Lucas Trindade, Nicole Príncipe, Roberval Júnior e Gustavo Santana.

Agradeço ao professor Cláudio Amorim, pela orientação fundamental durante a concepção e execução deste trabalho e pelos sábios conselhos que foram um aprendizado para mim.

Agradeço também ao professor Manoel Neto pela disponibilidade e orientação de uma aprendiz distante. Obrigada por ter confiado em mim.

Agradeço aos colegas do SIARQ, local onde pude crescer profissionalmente e criar laços de amizade. Agradeço também aos colegas do NTO por colaborarem para o meu amadurecimento pessoal e profissional.

Muito obrigado a todos, que direta ou indiretamente contribuíram para esta conquista.

"Tente uma, duas, três vezes e se possível tente a quarta, a quinta e quantas vezes for necessário. Só não desista nas primeiras tentativas, a persistência é amiga da conquista. Se você quer chegar aonde a maioria não chega, faça o que a maioria não faz."

Bill Gates

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Algoritmos

Lista de Abreviaturas e Siglas

AAL	*Ambient Assisted Living*
ADL	*Activities of Daily Living*
BSD	*Berkeley Software Distribution*
CDT	C/C++ *Development Tools*
CMYK	*Cyan, Magenta, Yellow and Black (Key)*
GB	*Gigabytes*
HSV	*Hue Saturation Value*
MAC OS	*Macintosh Operating System*
MP	*Megapixel*
OpenCV	*Open Source Computer Vision*
RF	Requisitos Funcionais
RGB	*Red, Green and Blue*
RNA	Rede Neural Artificial
RNF	Requisitos não Funcionais

Capítulo 1

Introdução

O aumento do envelhecimento populacional é uma realidade no Brasil e no mundo. Segundo o Ibge (2011) ocorreu uma mudança na pirâmide etária brasileira nos últimos anos. Enquanto a taxa de fecundidade se apresenta abaixo do nível de reposição populacional, o aumento da longevidade ampliou o número de idosos (IBGE, 2010b). A Figura 1.1 mostra dados do Censo do ano 2010, quando comparado com o Censo do ano 2000, a proporção da população mais jovem reduziu, enquanto a porcentagem populacional da faixa etária maior que 60 anos aumentou.

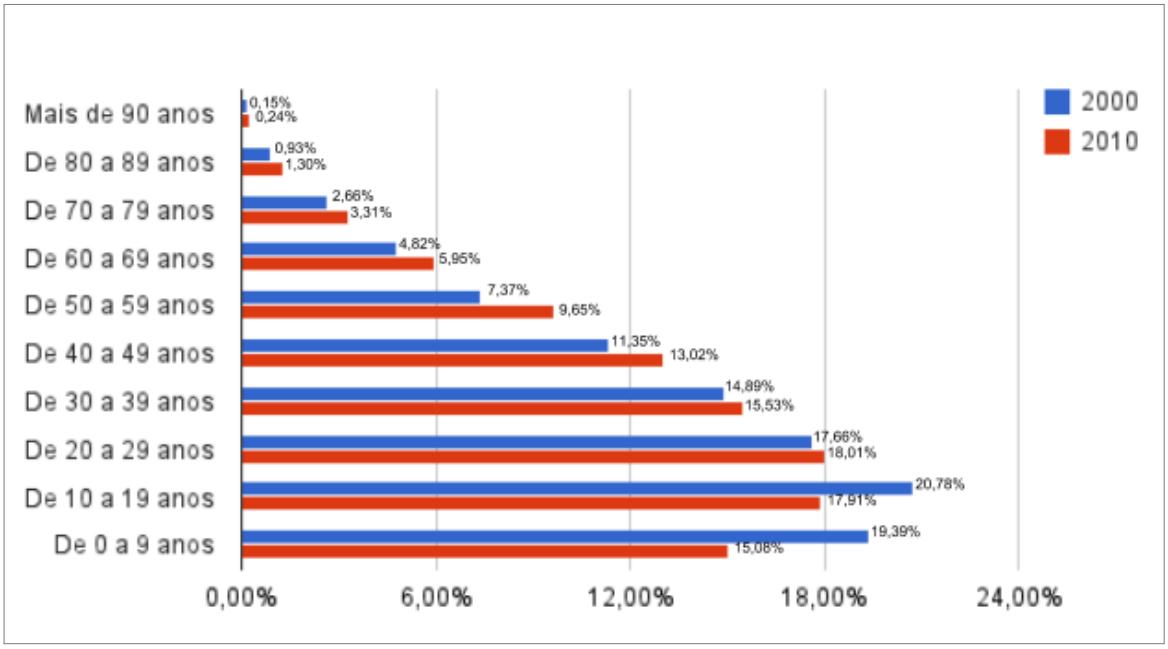

Figura 1.1: Distribuição da população por grupos de idade no Brasil.
Fonte: Adaptada de Ibge (2010a).

Noticiado pelo Ibge (2014), a tábua de mortalidade projetada para o ano de 2013 forneceu uma expectativa de vida de 74,9 anos para ambos os sexos. Com o aumento das chances de os indivíduos chegarem a idades mais avançadas, percebe-se um direcionamento de políticas voltadas para esse segmento populacional.

São idosos que constituem ou permanecem em domicílios unipessoais e necessitam de autonomia para realizar suas atividades diárias (CAMARGOS; RODRIGUES; MACHADO, 2011). Combinado a este fator é necessário reduzir os riscos e mitigar os danos causados por incidentes na rotina dessas pessoas. O envelhecimento por si só torna o idoso mais suscetível, pois diminui a capacidade funcional do ser humano (IBGE, 2010b).

O risco de acidentes domésticos é elevado nesse ambiente. O uso de dispositivos que identificam automaticamente comportamentos anormais podem encurtar o tempo de detecção e reconhecimento do perigo (KREKOVIć et al., 2012). Dispositivos em forma de pulseira, por exemplo, precisam estar a todo momento próximo do usuário. Esta é uma desvantagem deste modelo, uma vez que o uso constante pode causar desconforto, além de demandar trocas constantes de bateria.

Neste cenário, um sistema de monitoramento de atividade pode ser mais eficiente. Ele consiste em sensores para coletar os dados e um modelo de reconhecimento para inferir as ocorrências a partir dos dados coletados no sensor (KASTEREN; ENGLEBIENNE; KRÖSE, 2010). No monitoramento por vídeo, a análise é feita usando a alimentação em fluxo contínuo de uma câmera conectada ao computador. Com essa estrutura, a privacidade do idoso pode ser preservada, uma vez que não há a necessidade de gravação dos vídeos.

Nesse contexto, o presente trabalho busca propor e avaliar um algoritmo capaz de monitorar padrões anômalos de movimentação entre os cômodos em residências de idosos que moram ou ficam sozinhos. Este modelo será fundamentado nos aspectos de visão computacional aplicada em um sistema de monitoramento não invasivo. A análise automática de movimento humano através de vídeo vem sendo um dos principais temas de pesquisa em visão computacional (COSTA, 2008).

De acordo com Souza (2011), a visão computacional procura imitar o comportamento da visão humana, usando uma imagem como entrada, para que ocorra um processamento.

Apesar de sistemas automáticos requisitarem a atenção humana somente quando necessário, o videomonitoramento gera um grande volume de dados. Dessa forma, é fundamental o uso de um sistema com a finalidade de selecionar somente as informações importantes. O uso de uma tecnologia aberta, como o OpenCV *(Open Source Computer Vision)*, pode fomentar o desenvolvimento de aplicativos nesta área, uma vez que o acesso é mais facilitado. Existem empresas privadas usando métodos sofisticados de detecção de faces, por exemplo, porém os algoritmos não estão disponíveis para a comunidade científica (MACHADO et al., 2009).

Para a execução deste trabalho, o percurso metodológico foi divido em três fases. Na **primeira fase** pretende-se analisar as métricas para reconhecer o padrão de movimentação em um vídeo. Na **segunda fase** será feita a adaptação da biblioteca do OpenCV para a detecção de movimento. Na **terceira fase** pretende-se realizar a experimentação no ambiente determinado, obter e analisar os resultados.

Este trabalho está dividido em 6 capítulos. No **capítulo 1** foi realizada uma introdução com a finalidade de contextualizar o problema. No **capítulo 2** serão abordados os conceitos para fundamentação do trabalho. No **capítulo 3** estarão descritos alguns trabalhos correlatos que foram importantes na pesquisa. O **capítulo 4** explora a metodologia aplicada. O **capítulo 5** descreve os exemplos de uso executados e os resultados observados. No **capítulo 6** encontram-se as considerações finais do projeto, bem como propostas de trabalhos fututros.

Capítulo 2

Monitoramento de Padrões

Neste capítulo serão abordados os conceitos que fundamentam o presente trabalho, como os aspectos ligados a reconhecimento de padrões e videomonitoramento.

2.1 Sistema de Monitoramento de Idosos

A decisão de morar sozinho advém de uma série de fatores no arranjo familiar do indivíduo (CAMARGOS; RODRIGUES; MACHADO, 2011). Em alguns casos, separação ou divórcio, ou até mesmo a morte do cônjuge, como também a saída dos filhos de casa são eventos relatados como motivo para morar sem o convívio de outra pessoa. Independente da razão, ao idoso deve ser assegurada a manutenção da independência e autonomia para a execução das atividades de sua rotina.

Segundo Machado e Oliveira (2013), Ambientes Inteligentes podem ajudar a melhorar a qualidade de vida na residência provendo assistência as pessoas que nela residem. Essa tecnologia deve ser aplicada de forma não intrusiva, moldando-se ao ambiente e considerando as preferências de quem nela vive. A captura das situações vivenciadas é feita por sensores. Estes sensores produzem dados brutos e em grande quantidade. Isoladamente, eles não são capazes de determinar se existe ou não anormalidade nos elementos que foram captados. Para um aproveitamento mais eficiente, é realizada uma análise baseada no

contexto ao qual o ambiente está inserido.

Em um Ambiente de Vivência Assistida (AAL do inglês *Ambient Assisted Living*), o sistema combina produtos e serviços para possibilitar independência na execução das atividades do usuário (MACHADO; OLIVEIRA, 2013). Como um sistema adaptativo, o AAL provê interoperabilidade entre os diversos dispositivos "inteligentes" (televisão, aparelho de som, geladeiras, lâmpadas, celular) presentes na residência. Segundo Seguin, Lamotte e Philippe (2012), a arquitetura do *hardware* de um sistema AAL deve abranger todo a ambiente e permitir o gerenciamento de possíveis conflitos. A arquitetura do *software* deve atender as especificações do local a ser implantado e não depender do *hardware* utilizado. Um AAL deve adaptar-se à heterogeneidade dos dispositivos da residência, modelando interfaces amigáveis para atenuar a dificuldade no uso devido as limitações de pessoas idosas (MACHADO; OLIVEIRA, 2013).

Os sistemas de monitoramento de Atividades de Vida Diária (ADL do inglês *Activities of Daily Living*) causam menor desconforto, uma vez que não é necessário o uso de um acelerômetro preso ao corpo do usuário (OUCHI; DOI, 2013). Segundo Lin et al. (2011), o monitoramento de ADLs em tempo real podem detectar situações inseguras ou de perigo. O fluxo contínuo de eventos é comparado ao processo e as definições de risco para que o sistema avalie a ocorrência de uma situação de perigo. Para a análise do fluxo pode ser aplicada a técnica de rastreamento de movimento. De acordo com Kim et al. (2013), esta técnica consiste no processo de localização um objeto em movimento ou vários objetos ao longo do tempo usando uma câmera. Na Seção 2.2, a seguir, será discorrido sobre as técnicas aplicadas nos processos de Visão Computacional.

2.2 Visão Computacional

Conforme Sousa (2013), o objetivo da Visão Computacional é emular a visão humana usando uma imagem como entrada e uma interpretação desse conteúdo é então fornecida como saída. Para que esse processo seja executado, é necessária a aplicação de filtros com a finalidade de remover os ruídos que possam estar presentes e, posteriormente, o emprego da técnica de

segmentação para que seja feito o reconhecimento de padrões.

Segundo Marengoni e Stringhini (2009), os processos de visão computacional iniciam com o processamento das imagens. Nas imagens selecionadas, a extração requer, em alguns casos, que sejam convertidas para um determinado formato, tamanho ou ainda a aplicação de um filtro para remoção de ruídos que possam aparecer. Ruído representa qualquer informação não desejada na imagem (SOUSA, 2013).

2.2.1 Processamento de Imagens

Para facilitar o processamento computacional é necessário o uso de um modelo que descreva matematicamente como uma determinada cor deve ser representada. Existem diversas opções que podem, dependendo da aplicação, ser mais apropriadas que outras (SOUSA, 2013). Conforme Souza (2011), no modelo RGB (do inglês *Red, Green and Blue*), por exemplo, a imagem é decomposta nas cores vermelho, verde e azul para reproduzir outras cores. Um outro modelo encontrado é o CMYK (do inglês *Cyan, Magenta, Yellow and Black (Key)*). Similar ao RGB, o CMYK usa as cores ciano, magenta, amarelo e preto principalmente em impressão. Para sistemas em que o espaço de cor não é uniforme é utilizado o modelo de cores HSV (do inglês *Hue Saturation Value*) que separa os componentes de iluminação, matiz e saturação. Neste modelo, a matiz é responsável pelo tipo de cor representada na faixa do espectro de luz visível. O segundo componente é a saturação. Por último, o terceiro componente é relacionado com a iluminação (SOUSA, 2013). Tanto a saturação como a iluminação aceitam valores entre zero e cem porcento (0 - 100%).

Segundo Marengoni e Stringhini (2009), os filtros utilizados para remover os ruídos podem ser classificados como espaciais ou de frequência. Os filtros espaciais atuam no domínio espacial manipulando diretamente os *pixels* da imagem. A Equação 2.1 caracteriza os processos executados no domínio espacial, onde $f(x, y)$ é a imagem original, $T(.)$ é a transformação da imagem e $g(x, y)$ é a imagem transformada.

$$g(x, y) = T(f(x, y)) \tag{2.1}$$

Uma aplicação do operador T é a transformação de intensidade, realizada quando é necessária a binarização de uma imagem. A binarização consiste em utilizar um valor de corte para determinar que os valores acima deste limite, ou seja, regiões mais claras, sejam mapeadas para a cor branca e os valores abaixo do limite, ou seja, regiões mais escuras, sejam mapeadas para a cor preta.

Os filtros de frequência usam, inicialmente, a transformada de Fourier [1] para que a imagem seja transformada para o domínio de frequência e então possa ser aplicado o filtro. Em seguida, a imagem resultante é transformada novamente para o domínio de espaço.

2.3 Reconhecimento de Padrões

O reconhecimento de padrões pode proporcionar uma intersecção da visão computacional com a área de inteligência artificial, no momento em que um conjunto de objetos podem ser reconhecidos utilizando técnicas de aprendizado de máquina (MARENGONI; STRINGHINI, 2009).

2.3.1 Segmentação

A segmentação consiste em subdividir uma imagem de entrada em regiões, ou objetos distintos. Segundo Marengoni e Stringhini (2009), as técnicas de segmentação mais utilizadas com o OpenCV são as seguintes:

- Segmentação por detecção de borda.
- Segmentação por corte.
- Segmentação por crescimento de região.

A detecção de borda utiliza a identificação de uma mudança repentina no nível de intensidade dos *pixels*. Se os *pixels* variantes estiverem próximos, eles podem ser conectados, formando

[1] Transformada integral que expressa uma função em termos de funções de base sinusoidal, ou seja, como soma ou integral de funções sinusoidais multiplicadas por coeficientes ("amplitudes").

uma borda ou contorno na região ou objeto. Na segmentação por corte é feita a verificação do histograma da imagem. As regiões de picos e vales do histograma determinam o nível da segmentação. O procedimento para a segmentação por crescimento de região origina de um conjunto de pontos, denominado sementes. Na vizinhança das sementes são analisadas as propriedades similares como, por exemplo, cor, intensidade de nível de cinza, textura e momentos.

Com a conclusão da segmentação pode ser aplicado o processo de rastreamento, associando a informação sobre o movimento do objeto que está sendo rastreado com a finalidade de minimizar a busca entre as imagens em uma sequência (MARENGONI; STRINGHINI, 2009). Com o auxílio de uma ferramenta que aplique os conceitos de visão computacional é possível identificar dois componentes principais nas técnicas de rastreamento: identificação de objetos e modelagem da trajetória. Essas características devem ser consideradas em um projeto de sistema de monitoramento.

Segundo Kreković et al. (2012), o videomonitoramento é a maneira mais simples de detectar uma situação de perigo. A câmera pode ser instalada em um ponto estratégico de um cômodo da casa. A maior vantagem desse tipo de monitoramento é que não é preciso o uso de dispositivos, que muitas vezes são desconfortáveis e exigem mudanças frequentes de bateria.

2.3.2 OpenCV

OpenCV (Open Source Computer Vision) é uma biblioteca de programação, de código aberto, desenvolvida inicialmente pela Intel Corporation no ano de 2000. Segundo Marengoni e Stringhini (2009), foi idealizada com o objetivo de tornar a visão computacional acessível a usuários e programadores em áreas tais como a interação humano-computador e a robótica.

Escrita inicialmente em C, OpenCV foi projetada para ser multiplataforma. A partir da versão 2.0, os novos algoritmos passaram a ser desenvolvidos em C++ (OPENCV, 2014). *Wrappers* para linguagens como Python e Java foram desenvolvidos para incentivar a adoção por um público mais vasto.

De acordo com Machado et al. (2009), os algoritmos do OpenCV oferecem filtros de imagem como remoção de ruídos, detecção de bordas, linhas, quadrados, círculos e triângulos, calibração de câmera, reconhecimento e rastreamento de objetos, análise estrutural, operações matemáticas com matrizes de imagens de blocos de dados e apredizado de máquina.

Uma maneira simples de compilar os algoritmos desenvolvidos com a biblioteca OpenCV é utilizando o CMake [1]. Essa extensão de código aberto gerencia o processo de construção em um sistema operacional e de uma forma independente do compilador. Com este aplicativo não há necessidade de mudar nada no código ao portar entre Linux e Windows pois, ele é projetado para ser usado em conjunto com o ambiente de compilação nativa.

2.4 Considerações do Capítulo

Este capítulo objetivou explicar e fundamentar as características importantes que serão aplicadas neste trabalho. Inicialmente foi feita uma explanação dos modelos de monitoramento de idosos existentes na literatura. Por fim, foi feita uma descrição da metodologia necessária ao processamento de imagens, usando os conceitos de visão computacional.

Todos os autores citados colaboraram de alguma forma para a concepção do presente trabalho, em especial, os trabalhos relacionados que serão descritos no Capítulo 3.

[1] http://www.cmake.org/

Capítulo 3

Trabalhos Relacionados

Neste capítulo serão descritos alguns trabalhos que possuem relação com a detecção de movimento e o reconhecimento de padrões. Estes abordam duas áreas da literatura: monitoramento de idosos e processamento de imagens geradas a partir de vídeos. Ao final, será apresentada a oportunidade de pesquisa encontrada.

3.1 Segmentação, Rastreamento de Objetos e Detecção de Eventos Primitivos com Aplicação no Monitoramento Automático de Ações Humanas em Vídeo

O trabalho de Costa (2008), é focado na análise automática de movimento humano em vídeo digital. Os conceitos de Visão Computacional foram aplicados em um sitema de vigilância excecutado no domínio chamado de *looking at people* (olhando as pessoas, traduzindo do inglês). Para o autor, sistemas desse tipo são compostos por módulos agrupados em três áreas: detecção de movimento, rastreamento de pessoas e classificação da ação de acordo com as necessidades da aplicação. Conforme ilustrado na Figura 3.1, o autor afirma que a detecção do movimento é o módulo de mais baixo nível de abstração, enquanto o de classificação das ações o de mais alto nível.

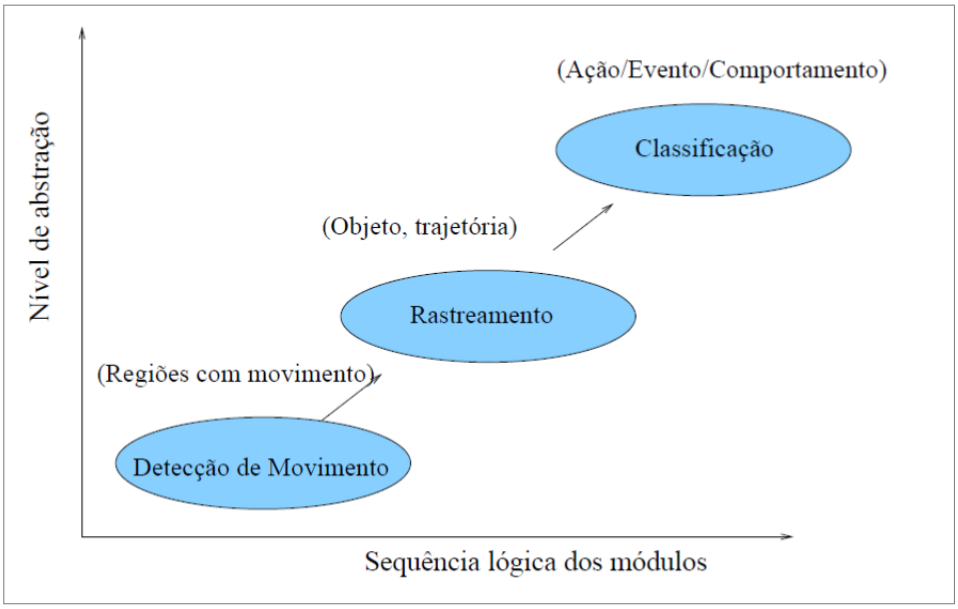

Figura 3.1: Níveis de abstração de um sistema típico de classificação/reconhecimento de ações humanas.

Fonte: Costa (2008).

De acordo com Costa (2008), o algoritmo terá maior robustez se, em um ambiente não controlado, requisitar um menor número de suposições. Considerando este aspecto, o modelo adaptativo pode apresentar maior eficácia na detecção de movimento, uma vez que a subtração de imagem de fundo é atualizada a cada novo quadro do vídeo.

3.2 Algoritmo para reconhecimento e acompanhamento de trajetória de padrões em vídeos

Neste projeto, Souza (2011) visa o desenvolvimento de um algoritmo para reconhecimento e acompanhamento de trajetória de padrões em vídeos. Para isso ele utilizou técnicas de processamento digital de imagens, de reconhecimento de padrões a partir de Redes Neurais Artificiais (RNAs) e de detecção de trajetória. Apesar do alto consumo computacional inicial, o uso de RNAs para o reconhecimento de padrões possibilita rápida execução após o treinamento da rede, ou seja, quando ocorre a aprendizagem de máquina. A detecção de trajetória foi executada utilizado o algoritmo de rastreamento de cor conhecido como *Camshift*. Para a implementação do algoritmo foi utilizada como ferramenta computacional

de apoio a biblioteca OpenCV. O algoritmo desenvolvido tinha a finalidade de efetuar a etapa de pré-processamento de imagens, o reconhecimento de um padrão nela contido através de Redes Neurais Artificiais e o acompanhamento desse objeto em uma seqüência sucessiva de imagens.

3.3 Um Método para Detecção em Tempo Real da Queda Humana a Partir de Vídeo

O medo de queda é uma das principais razões de uma pessoa idosa não morar sozinha (KREKOVIć et al., 2012). Os autores desse artigo afirmam ainda que, o risco de ferimentos graves é aumentado se a pessoa permanece inconsciente ou imobilizada depois da queda por causa da sua incapacidade de chamar alguém para obter ajuda. Por causa disto, os dispositivos que detectam automaticamente a queda estão no foco de interesse.

Dessa forma, os autores propuseram um método de detecção executado em quatro etapas: uma estimativa de fundo, seguida de extração da movimentação de objetos, extração das características do movimento e, finalmente, a detecção de queda. A detecção é baseada em características que quantificam a dinâmica do movimento humano e a orientação do corpo. Os algoritmos foram implementados em C++, usando a biblioteca OpenCV. As Figuras 3.2, 3.3 e 3.4 representantam a sequência do processamento das imagens para que a queda seja detectada.

Figura 3.2: Resultado da subtração de fundo.

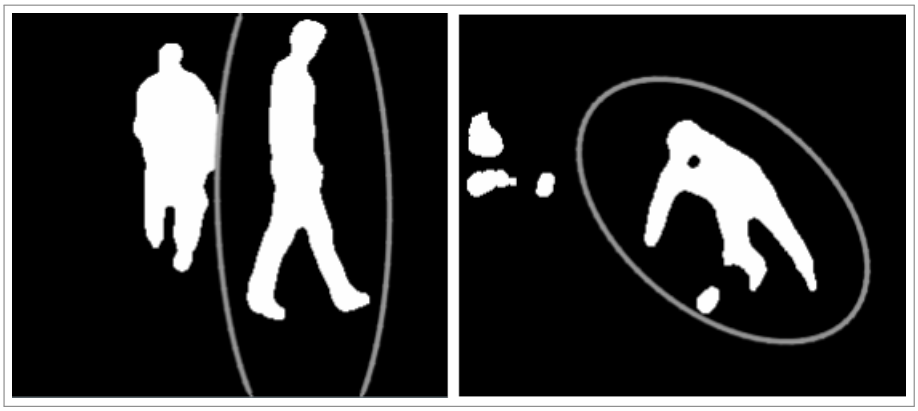

Figura 3.3: Resultado da aproximação elíptica do objeto

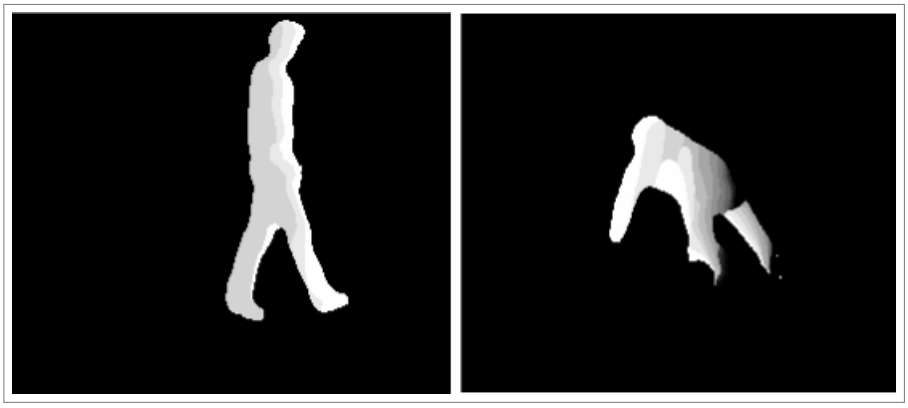

Figura 3.4: Histórico do movimento da imagem.
Fonte: Adaptada de Krekovič et al. (2012).

3.4 Conclusões e oportunidades de pesquisa

Os trabalhos apresentados nas Seções 3.1, 3.2 e 3.3 contribuirão para o desenvolvimento e validação do modelo proposto no presente trabalho. As similaridades encontradas nessas pesquisas serão consideradas na proposta de desenvolvimento do presente estudo, como por exemplo o uso da biblioteca OpenCV. A página oficial do OpenCV [1] confirma que a biblioteca é liberada sob a licença BSD, ou seja, é gratuito tanto para uso acadêmico como para comercial.

O trabalho desenvolvido por Costa (2008) servirá de arcabouço para o uso dos conceitos de Visão Computacional aplicados neste trabalho. A abstração em módulos, representada por este autor, permitiu um entendimento da sequência lógica necessária para um sistema de

[1] Disponível em: http://opencv.org/

videomonitoramento. O estudo feito por Souza (2011) será aproveitado em alguns pontos. Por outro lado, o uso de RNAs para o reconhecimento de padrões pode ficar compremetido se for aplicado em uma arquitetura menos robusta, como as máquinas para uso doméstico, por exemplo. A pesquisa de Kreković et al. (2012) contribuiu para a escolha do público ao qual o presente trabalho será direcionado. Pessoas idosas, em algumas situações, sentem dificuldade em usar as novas tecnologias. Sendo assim, um método não invasivo poderá permitir maior conforto do usuário.

A Tabela 3.1 resume as características dos trabalhos relacionados. O presente estudo difere dos demais principalmente quanto ao método inicial de subtração. A subtração absoluta permite que o sistema seja adaptativo e tenha a imagem de referência atualizada a cada no quadro. O que aproxima todos os trabalhos é a linguagem utilizada e o uso de algumas ferramentas computacionais, como o OpenCV, por exemplo.

	Este projeto	**COSTA, 2008**	**SOUZA, 2011**	**KREKOVIĆ et al., 2012**
Ambiente	Residência	Estação de metrô	Figuras geométricas simples	Residência
Linguagem usada	C++	C	C++	C++
Ferramentas	OpenCV, Eclipse com o plugin CDT e CMake	OpenCV, Eclipse com o plugin CDT e FFMPEG	OpenCV, Dev C++ e Format Factory 2.50	OpenCV
Métodos	Subtração absoluta	Subtração de *background*	Filtro *gaussiano*	Subtração de *background*

Tabela 3.1: Comparação entre os trabalhos relacionados.
Fonte: Própria autora.

O referencial teórico descrito foi utilizado para o planejamento do estudo realizado. As etapas do percurso metodológico serão apresentadas no Capítulo 4, a seguir.

Capítulo 4

Sistema para reconhecimento de padrões

Este capítulo apresenta, na Seção 4.1, o detalhamento da metodologia de desenvolvimento para o sistema proposto, na Seção 4.2, os requisitos descritos para o sistema, e na Seção 4.3, a arquitetura desenhada para este modelo.

4.1 Metodologia de desenvolvimento

Para o desenvolvimento do modelo de reconhecimento de padrões proposto neste trabalho foi desenhado este percurso metodológico, que tem como objetivo a definição e apresentação das fases e atividades necessárias para alcançar o desenvolvimento completo do sistema. O primeiro passo foi a busca de uma ferramenta *open source* para que o sistema pudesse estar acessível ao maior número de usuários. A metodologia de desenvolvimento escolhida está dividida em três fases. Essas fases estão agrupadas, de modo que, possam ser obtidos os melhores resultados.

A **primeira fase** consiste na seleção das técnicas necessárias à detecção do movimento e ao rastreamento do objeto alvo presente no vídeo de amostra. De acordo com o referencial apresentado no Capítulo 2, o primeiro passo é o processamento das imagens obtidas. A Figura 4.1 representa a sequência lógica que deve ser seguida. Inicialmente ocorreu a aquisição da imagem que vai ser submetida ao processamento. Logo após, foi executada uma

preparação da imagem, com a finalidade de remover ruídos indesejados. Conforme Delai e Coelho (2010), foi dada preferência a procedimentos no domínio espacial da imagem, ao invés do domínio da frequência como apresentado na literatura teórica do Capítulo 2. Esta escolha deve-se ao fato que, as transformadas de Fourier demandam um alto custo computacional frente ao domínio espacial podendo comprometer a velocidade total dos processos. Após serem processadas, as imagens podem ser encaminhadas ao módulo de detecção de movimento.

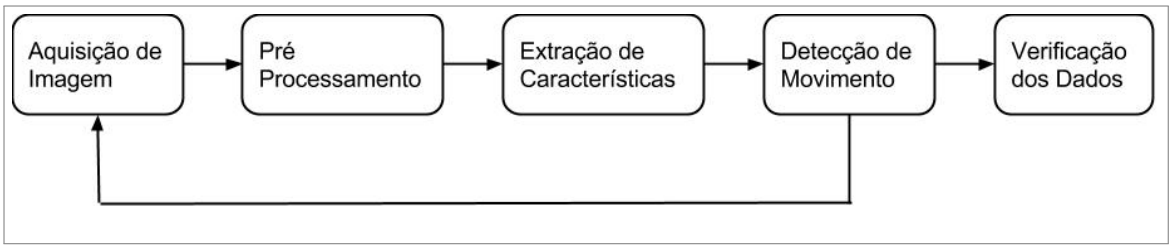

Figura 4.1: Metodologia adotada para o sistema de monitoramento.
Fonte: Própria autora.

Para a **segunda fase** foi necessário o conhecimento das funções do OpenCV com a finalidade de adaptá-las ao modelo proposto. A segmentação dos objetos serviu para realizar a extração das características de interesse. Sendo assim, a subtração absoluta de fundo foi escolhida como função para detectar o movimento. A aplicação de um modelo estatístico contribuiu para a minimização de ruídos nas imagens, colaborando para o reconhecimento de padrões. A função de transformação de intensidade foi utilizada para separar os componentes de interesse e isolá-los. A imagem foi binarizada a partir de um ponto de corte definido nas tentativas de ajustes. Neste momento já foi possível fazer a detecção de movimento. Se nenhum movimento for detectado, o sistema adquire as próximas imagens para serem submetidas ao processamento.

Por fim, na **terceira fase** foram feitos os exemplos de uso aplicados em um ambiente simulando um cômodo de uma residência. Após o processamento da imagem adquirida foi feita a análise dos dados extraídos. Na saída do vídeo é sinalizada a detecção de movimento e o rastreamento do objeto.

Na Seção 4.2, a seguir, serão descritos os requisitos necessários para a implementação deste projeto.

4.2 Requisitos do sistema

Para a execução deste projeto foram selecionadas as funcionalidades indispensáveis para o atendimento dos requisitos do sistema. A documentação padronizada é um item fundamental para que outras pessoas entendam as características do software. Na Seção 4.2.1 estarão descritos os requisitos funcionais e na Seção 4.2.2 estarão descritos os requisitos não funcionais.

4.2.1 Requisitos funcionais

Os requisitos funcionais determinam explicitamente o que o sistema deve realizar a partir de entradas específicas. Na Tabela 4.1 estão representados esses requisitos.

Código	**Descrição**
RF1	O Sistema deve monitorar a câmera quando a alimentação for em fluxo contínuo.
RF2	O Sistema deve aceitar apenas uma forma de alimentação das imagens por execução, diretamente da câmera ou arquivo de vídeo.
RF3	O Sistema deve informar a detecção do movimento.
RF4	O Sistema deve rastrear o objeto quando o movimento for detectado.

Tabela 4.1: Requisitos funcionais.
Fonte: Própria autora.

De acordo com a Tabela 4.1, o Sistema deve gerenciar a alimentação do vídeo e informar caso ocorra a detecção do movimento no ambiente testado.

4.2.2 Requisitos não funcionais

Os requisitos não funcionais estão relacionados ao uso da aplicação em termos de desempenho, usabilidade, confiabilidade, segurança, disponibilidade, manutenibilidade e tecnologias envolvidas. Na Tabela 4.2 estão representados esses requisitos.

Código	Descrição
RNF1	O Sistema deve aceitar ser operacionalizado em plataformas variadas sem alteração no desempenho.
RNF2	O Sistema deve suportar grande volume de dados.
RNF3	O Sistema deve permitir um videomonitoramento não invasivo dos usuários.

Tabela 4.2: Requisitos não funcionais.
Fonte: Própria autora.

Conforme a Tabela 4.2, o Sistema deve ser multiplataforma, ter o planejamento de gerenciamento de um grande volume de dados e ainda executar o videomonitoramento de forma não invasiva.

4.3 Arquitetura do sistema

Auxiliando no esboço da arquitetura do sistema proposto, o modelo conceitual apresentado na Figura 4.2 contém os elementos do domínio de um sistema de monitoramento.

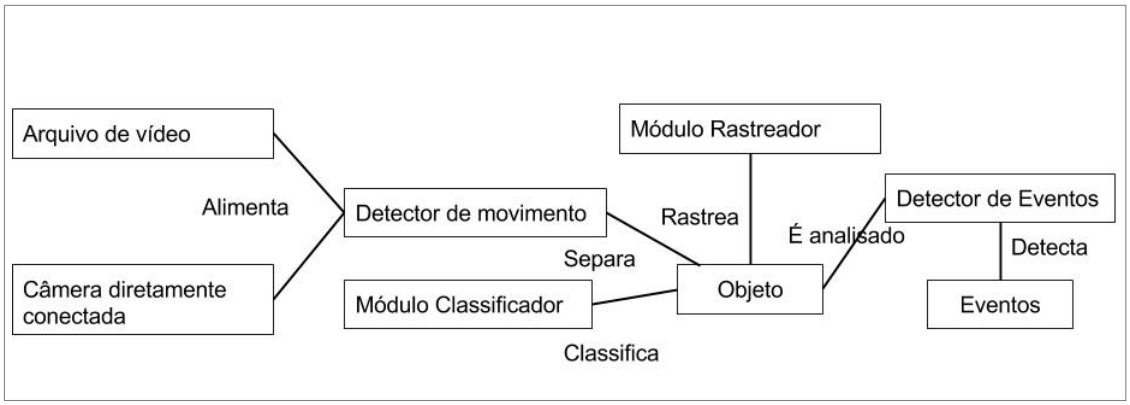

Figura 4.2: Modelo conceitual do sistema de monitoramento.
Fonte: Adaptada de Costa (2008).

Os principais conceitos apresentados são o detector de movimento, o módulo rastreador e o detector de eventos. O detector de movimento é alimentado pela sequência de quadros do vídeo, capturado por uma câmera em tempo real ou armazenado em arquivo. Seguindo

o fluxo, os objetos são segmentados e, posteriormente, classificados de acordo com os requisitos da aplicação. Os objetos alvo são, então, rastreados. Por último, o detector de eventos coleta informações para detectar a ocorrência ou não de algum evento relevante.

Na Figura 4.3 é possível observar o diagrama de atividades apresentado para o módulo detector de movimento. Se os dispositivos para a captura estiverem em conformidade, as imagens são lidas e tratadas para serem usadas como entrada no detector de movimento.

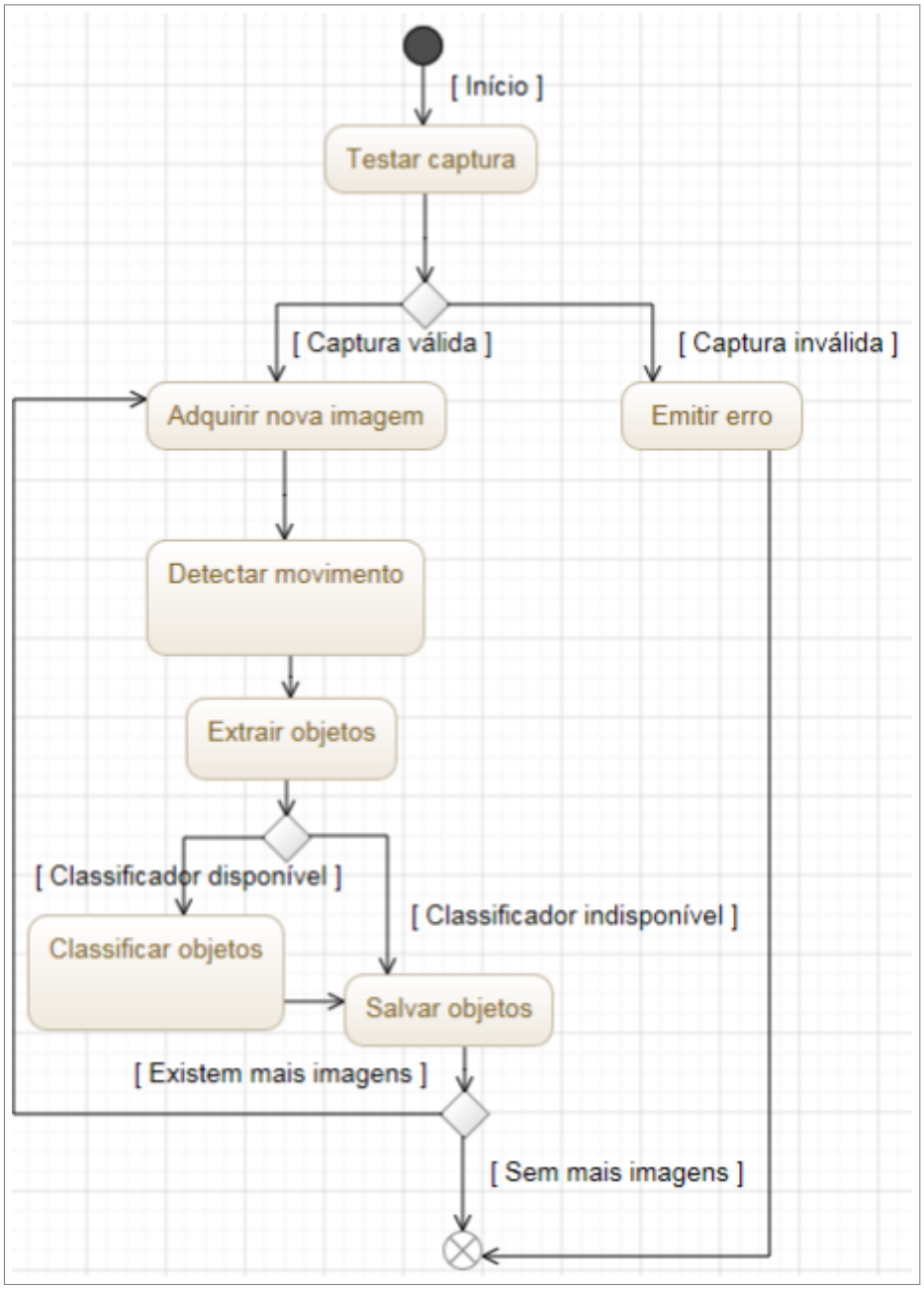

Figura 4.3: Diagrama de atividades para o detector de movimento.
Fonte: Adaptada de Costa (2008).

Na Figura 4.4 está representado o diagrama de atividades referente ao módulo rastreador de objetos. As saídas em imagem desse processo são os próprios quadros do vídeo de entrada, marcados com o retângulo mínimo do objeto rastreado. A cada iteração, o módulo de rastreamento carrega os objetos do próximo quadro e os passa para o algoritmo de rastreamento para que o algoritmo faça o reconhecimento.

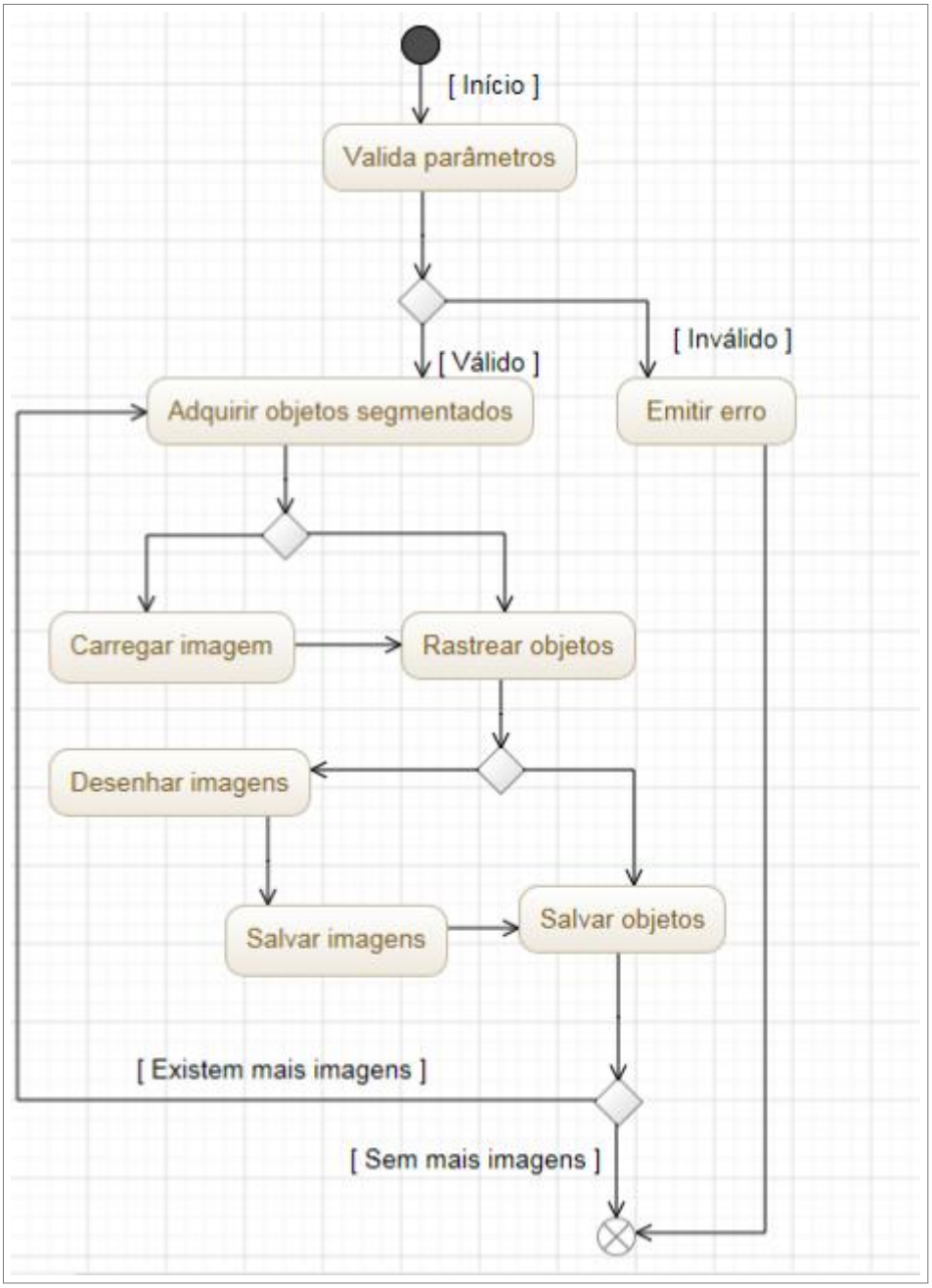

Figura 4.4: Diagrama de atividades para o rastreador de objetos.
Fonte: Adaptada de Costa (2008).

4.4 Considerações do capítulo

Neste Capítulo 4 foi feita a apresentação da metodologia de desenvolvimento adotada para este trabalho e que foi segmentada em três fases. Em seguida foram detalhados os requisitos dos sistema, expostos na forma de requisitos funcionais e não funcionais. Por fim, foi mostrado o modelo conceitual proposto.

Capítulo 5

Resultados Experimentais

Neste capítulo será descrito detalhadamente a validação do modelo proposto, os exemplos aplicados para a detecção de movimento e a análise dos resultados obtidos.

5.1 Introdução

O principal objetivo da presente validação é a detecção de movimento em um cômodo de uma residência e o posterior rastreamento da movimentação desse objeto. O experimento se iniciou com a adaptação de funções do OpenCV.

Um quesito desafiador foi a manipulação dos arquivos de vídeo. Apesar da biblioteca OpenCV possuir funções para diversos aspectos da visão computacional, o ajuste adequado das imagens para o processamento dos quadros não é um assunto trivial.

5.2 Implementação

Para a implementação do modelo foi utilizada a linguagem de programação C++ combinada com a biblioteca de visão computacional OpenCV. A escolha dessa linguagem se deu pelo fato de ser multiplataforma e compatível com as funcionalidades do OpenCV. Como ambiente de desenvolvimento foi utilizada a ferramenta Eclipse com o plugin CDT (do

inglês C/C++ *Development Tools*) instalado em um computador portátil Dell Vostro 1320, com processador Intel *Core 2 Duo* e 3GB de memória. Apesar de também ser compatível com o sistema operacinal Linux, o ambiente foi configurado em uma instalação do sistema operacional *Windows* 7 *Home Basic, Service Pack* 1, de 32 *bits*. Para a gravação dos arquivos de vídeo foi uma utilizada uma câmera Samsung DV2014F com resolução de 16 MP.

O começo da implementação é baseado no módulo de detecção de movimento. O primeiro passo foi testar a captura do vídeo para análise, para isso foi necessário informar ao algoritmo o caminho completo de localização do arquivo no computador. Se este parâmetro estiver correto, o programa executa um laço de repetição que só é finalizado quando não existem novos quadros para serem analisados. A saída da leitura do vídeo é adicionada em uma matriz. Como representa o Algoritmo 5.1, são selecionados dois quadros, ordenados e subsequentes, que tem as cores das imagens tranformadas em escala de cinza, visto que o algoritmo desse projeto não aplica o rastreamento através de cor.

Algoritmo 5.1 Leitura dos quadros e conversão em escala de cinza.

```
1:  capture.read(frame1);
2:
3:  cv::cvtColor(frame1,grayImage1,COLOR_BGR2GRAY);
4:
5:  capture.read(frame2);
6:
7:  cv::cvtColor(frame2,grayImage2,COLOR_BGR2GRAY);
8:
9:  cv::absdiff(grayImage1,grayImage2,differenceImage);
```

A função *cvtColor* possui três parâmetros. O primeiro é a imagem de entrada. O segundo é a imagem de saída, que deve ter o mesmo tamanho e profundidade da imagem de entrada. E o último é o código de conversão de espaço de cor. Em seguida, é aplicada a diferença absoluta elemento a elemento entre duas matrizes. Como o sistema implementado é adaptativo, a subtração de fundo é realizada dinamicamente. Como mostra o Algoritmo 5.1, a função *absdiff* utiliza os quadros lidos, e já editados na função *cvtColor*, como os parâmetros de entrada e o terceiro é o de saída.

Após isso é realizada a limiarização da imagem usando a função *threshold*. No Algoritmo 5.2 é possível visualizar que a imagem resultante da função *absdiff* é usada como entrada na função *threshold*. É então aplicado um limiar de nível fixo para cada elemento da matriz

Algoritmo 5.2 Limiarização, remoção de ruídos e binarização da imagem.

```
cv::threshold(differenceImage,thresholdImage,SENSITIVITY_VALUE,255,
THRESH_BINARY);
cv::blur(thresholdImage,thresholdImage,cv::Size(BLUR_SIZE,BLUR_SIZE
));
cv::threshold(thresholdImage,thresholdImage,SENSITIVITY_VALUE,255,
THRESH_BINARY);
```

que é definido como um valor de sensibilidade para ser aplicado na binarização da imagem. Quanto menor este valor, mais sensível será. Depois é feito um desfoque da imagem para remoção de ruído utilizando a função *blur*. Esta função borra uma imagem usando o filtro de quadro normalizado. É necessário usar a função *threshold* novamente para que a imagem de saída esteja binarizada mais uma vez, ou seja, em condições favoráveis para ser usada para a próxima etapa, que é integrante do módulo de rastreamento.

Como mostra o Algoritmo 5.3, neste momento já é possível usar a função *findContours* para selecionar os objetos de interesse. Esse procedimento permite encontrar contornos selecionando apenas os contornos mais externos em uma imagem binária. A função *approxPolyDP* é importante pois, faz a aproximação dos contornos usando o algoritmo de Douglas-Peucker. O algoritmo DP simplifica o número de pontos exitentes em um polígono.

Algoritmo 5.3 Busca dos contornos dos objetos de interesse.

```
vector<vector<Point> > contours;
vector<Vec4i> hierarchy;

findContours( thresholdImage, contours, hierarchy, CV_RETR_EXTERNAL,
CV_CHAIN_APPROX_SIMPLE, Point(0, 0) );

vector<vector<Point> > contours_poly( contours.size() );
vector<Rect> boundRect( contours.size() );

double area_maior=0;
int j = 0;
for ( int i = 0; i < contours.size(); i++ ) {
    approxPolyDP( Mat(contours[i]), contours_poly[i],
    3, true );
    double area = contourArea(contours_poly[i]);
    if (area > area_maior) {
       area_maior = area;
       j = i;
    }
    boundRect[i] = boundingRect( Mat(contours_poly[i]) );
}
```

Ainda como etapa do rastreamento do objeto, a função *drawContours* desenha uma linha nos contornos mais externos do objeto. Como mostrado no Algoritmo 5.4, os contornos podem ser desenhados a partir de formas geométricas diversas. Para o presente trabalho, foram usados os contornos poligonais e em forma de retângulo, desenhados na cor branca na tela de acompanhamento e na cor vermelha no vídeo final.

Algoritmo 5.4 Desenho dos contornos dos objetos de interesse.

```
 1:  Mat drawing = Mat::zeros( thresholdImage.size(),  CV_8UC3);
 2:
 3:  for ( int i = 0; i< contours.size(); i++ ) {
 4:     drawContours( drawing, contours_poly, i, Scalar (255,255,255), 1,
 5:     8, vector<Vec4i>(), 0, Point() );
 6:
 7:     rectangle( drawing, boundRect[i].tl(),
 8:     boundRect[i].br(), Scalar(255,255,255), 2, 8, 0 );
 9:
10:     double area = contourArea(contours_poly[i]);
11:     if (area > 15000 && area < 900000) {
12:        rectangle( frame1, boundRect[j].tl(),
13:        boundRect[j].br(), Scalar(0,0,255), 2, 8, 0 );
14:     }
15:  }
16:
```

5.3 Análise dos resultados

Como descrito no Capítulo 1, este projeto foi desenvolvido com a finalidade de propor um algoritmo que fosse capaz de monitorar padrões anômalos de movimentação em uma residência. A implementação desse sistema dependeria da aplicações do conceitos de visão computacional, direcionados para a detecção de movimento e rastreamento de objetos em um vídeo. Para avaliar os resultados obtidos nesse trabalho foram coletadas as imagens após a aplicação dos processos citados na Seção 5.2.

A Figura 5.1 mostra a imagem resultante após a aplicação da subtração absoluta de fundo. A saída é composta apenas pela saturação dos *pixels* diferentes entre a imagem atual e a seguinte. Como as imagens de entrada estavam em escala de cinza, a imagem de saída também será assim. Porém, se o movimento for executado de maneira lenta, a função de diferença absoluta de fundo pode apresentar uma falso negativo.

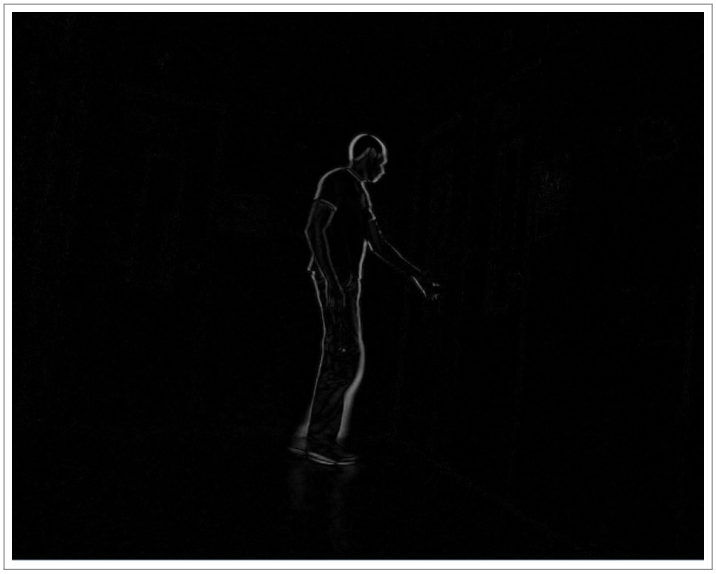

Figura 5.1: Resultado da diferença absoluta de fundo.
Fonte: Própria autora.

A função *absdiff* é representada pela Equação 5.1, onde $src1(I)$ é a imagem atual e $src2(I)$ é a imagem seguinte.

$$dst(I) = cor(|src1(I) - src2(I)|) \tag{5.1}$$

A Figura 5.2 mostra a imagem resultante após a limiarização e binarização.

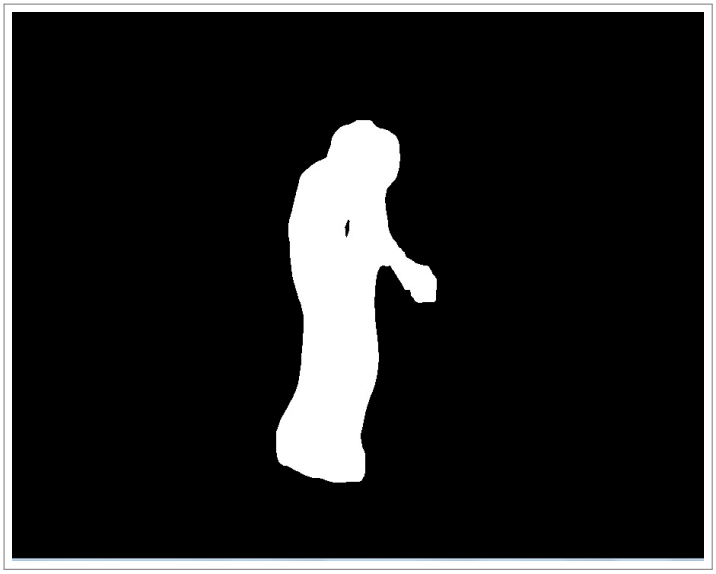

Figura 5.2: Resultado da limiarização e binarização da imagem.
Fonte: Própria autora.

A função *threshold* é representada pela Equação 5.2, onde $maxval$ é a valor que será aplicado como cor do pixel caso a cor atual seja maior que o limiar e $thresh$ é o valor do limiar fixo. Esta é a equação para o parâmetro *THRESH_BINARY*. O limiar fixo aplicado neste projeto foi 15, pois era necessário uma sensibilidade alta.

$$dst(x,y) = \begin{cases} maxval & : if src(x,y) > thresh \\ 0 & : outra forma \end{cases} \tag{5.2}$$

A Figura 5.3 mostra o resultado do processo de segmentação da região de interesse. Nela, o desenho poligonal ao redor do objeto encontrado e o retângulo limitando foram recuperados após o uso da função *findContours*. A função não considera o primeiro pixel da borda da imagem, por este motivo, imagens que tocam a borda são cortadas.

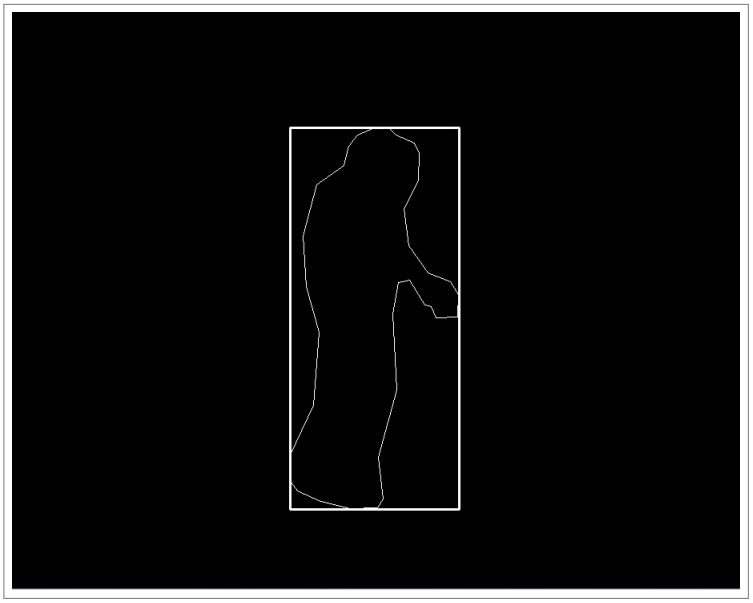

Figura 5.3: Desenho dos contornos na imagem de acompanhamento.
Fonte: Própria autora.

A Figura 5.4 apresenta a saída no vídeo após serem aplicadas as técnicas de detecção de movimento e rastreamento de objetos. Se o movimento não for detectado, o algoritmo continua operando normalmente. O sistema se encerra quando a alimentação das imagens é interrompida.

Na saída do vídeo é mostrada uma caixa de texto no canto inferior esquerdo contendo a data

Figura 5.4: Resultado final.
Fonte: Própria autora.

e a hora que ocorreu a detecção do movimento. Também é mostrada a mensagem "Humano entrou na sala!"no canto superior direito.

Capítulo 6

Considerações Finais

Neste capítulo serão descritas as conclusões obtidas a partir dos experimentos realizados com este trabalho, bem como propostas de trabalhos futuros.

6.1 Conclusões

O cuidado com as pessoas idosas está aumentando cada dia mais. Com o auxílio do videomonitoramento esta atividade tem se tornado menos incômoda. Para que as câmeras não sejam apenas coletoras de dados brutos, a conexão com sistemas computacionais torna possível a análise e manipulação desses dados.

Neste cenário, o presente projeto apresentou um estudo que buscou propor e avaliar um algoritmo que fosse capaz de reconhecer os padrões de movimentação em residências de idosos que moram ou permanecem sozinhos. Para isso foi essencial a abordagem dos conceitos de visão computacional implementados através da interface da biblioteca OpenCV.

Após a realização dos experimentos e execução dos exemplos de uso em um ambiente simulado, foi possível concluir que a detecção de movimento pôde ser percebida em 90% dos vídeos analisados. Nos momentos identificados como falso negativo, a falta de detecção deve-se ao fato que, para movimentos muito lentos, a diferença absoluta de fundo encontra poucos *pixels* entre a imagem atual e a seguinte, sinalizando-os como apenas ruído.

Assim que o movimento foi detectado, o rastreamento do objeto pôde ser observado em 100% das imagens coletadas. A representação se deu pelo desenho de um retângulo vermelho na saída de vídeo final. Esse retângulo acompanhou a pessoa durante a sua passagem no vídeo.

O uso da biblioteca OpenCV permitiu grande auxílio no desenvolvimento do projeto, ainda assim, características pertinentes ao ambiente externo podem interferir nos resultados observados. Fatores como iluminação do ambiente, móveis em uma residência podem atribuir, erroneamente, resultados positivos ou negativos.

Como limitações deste projeto foi identificado que, com o uso de apenas uma câmera, podem ocorrer áreas de oclusão de imagens. Segundo Delai e Coelho (2010), este efeito poderia ser minimizado caso fosse usado algum algoritmo de fluxo óptico, como o de Lucas-Kanade, por exemplo. Outra limitação identificada está relacionada à câmera, que deve estar parada e em um local estável.

6.2 Trabalhos futuros

Como proposta para trabalhos futuros sugere-se a adaptação do sistema para o Raspeberry pi que tem se mostrado robusto para aplicações de pequeno porte.

O uso de mais de uma câmera poderá contribuir para mitigar os danos causados em casos de oclusão dos objetos rastreados.

Também poderá ser projetado o monitoramento de outros padrões da rotina diária.

Referências Bibliográficas

CAMARGOS, M. C. S.; RODRIGUES, R. N.; MACHADO, C. J. Idoso, família e domicílio: uma revisão narrativa sobre a decisão de morar sozinho. *Rev. bras. estud. popul.*, 2011.

COSTA, B. A. D. da. *Segmentação, Rastreamento de Objetos e Detecção de Eventos Primitivos com Aplicação no Monitoramento Automático de Ações Humanas em Vídeo*. Tese (Mestrado) — Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, 2008. Disponível em: <http://docs.computacao.ufcg.edu.br/posgraduacao/dissertacoes/2008/Dissertacao_BrunoAlexandreDiasdaCosta.pdf>. Acesso em: 07 de Dezembro de 2014.

DELAI, R. L.; COELHO, A. D. *Visão Computacional com a OpenCV – Material Apostilado e Veículo Seguidor Autônomo*. 2010. Disponível em: <http://www.maua.br/pesquisas/visao-computacional-com-opencv-material-apostilado-veiculo-seguidor-autonomo>. Acesso em: 20 de Dezembro de 2014.

IBGE, I. B. d. G. e. E. *Sinópse dos Resultados de Censo 2010*. Rio de Janeiro, RJ: [s.n.], 2010. Disponível em: <http://www.censo2010.ibge.gov.br/sinopse/webservice/>. Acesso em: 03 de Novembro de 2014.

IBGE, I. B. d. G. e. E. *Síntese de Indicadores Sociais*. Rio de Janeiro, RJ: [s.n.], 2010. Disponível em: <http://www.ibge.gov.br/home/estatistica/populacao/condicaodevida/indicadoresminimos/sinteseindicsociais2010/SIS_2010.pdf>. Acesso em: 19 de Outubro de 2014.

IBGE, I. B. d. G. e. E. *Censo Demográfico 2010: Características da população e dos domicílios*. Rio de Janeiro, RJ: [s.n.], 2011. Disponível em: <http://biblioteca.ibge.gov.br/visualizacao/periodicos/93/cd_2010_caracteristicas_populacao_domicilios.pdf>. Acesso em: 19 de Outubro de 2014.

IBGE, I. B. d. G. e. E. *Tábua completa de mortalidade para o Brasil – 2013*. Rio de Janeiro, RJ: [s.n.], 2014. Disponível em: <ftp://ftp.ibge.gov.br/Tabuas_Completas_de_Mortalidade/Tabuas_Completas_de_Mortalidade_2013/notastecnicas.pdf>. Acesso em: 03 de Dezembro de 2014.

KASTEREN, T. L.; ENGLEBIENNE, G.; KRÖSE, B. J. An activity monitoring system for elderly care using generative and discriminative models. *Personal Ubiquitous Comput.*, Springer-Verlag, London, UK, UK, v. 14, n. 6, p. 489–498, set. 2010. ISSN 1617-4909. Disponível em: <http://dx.doi.org/10.1007/s00779-009-0277-9>.

KIM, J.-T. et al. *Emergency Situation Alarm System Motion Using Tracking of People like Elderly Live Alone*. Suwon, South Korea's: IEEE, 2013. Disponível em: <http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6579321>. Acesso em: 12 de dezembro de 2014.

KREKOVIć, M. et al. *A method for real-time detection of human fall from video*. Opatija, Croácia: IEEE, 2012. Disponível em: <http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=6240925>. Acesso em: 06 de outubro de 2014.

LIN, Z. et al. Precise process definitions for activities of daily living: A basis for real-time monitoring and hazard detection. In: *Proceedings of the 3rd Workshop on Software Engineering in Health Care*. New York, NY, USA: ACM, 2011. (SEHC '11), p. 13–16. ISBN 978-1-4503-0585-3. Disponível em: <http://doi.acm.org/10.1145/1987993.1987998>.

MACHADO, A.; OLIVEIRA, J. P. M. de. Adaptação apoiada por composição de serviços em ambientes ubíquos sensíveis ao contexto. *Cadernos de Informática*, 2013.

MACHADO, B. B. et al. Implementação de um algoritmo de reconhecimento facial usando eigenface. *Revista e-Xacta*, 2009. Disponível em: <http://revistas.unibh.br/index.php/dcet/article/viewFile/247/137>. Acesso em: 15 de Dezembro de 2014.

MARENGONI, M.; STRINGHINI, D. Adaptação apoiada por composição de serviços em ambientes ubíquos sensíveis ao contexto. *Revista de Informática Teórica e Aplicada*, 2009. Disponível em: <http://seer.ufrgs.br/rita/article/viewFile/rita_v16_n1_p125/7289>. Acesso em: 11 de Dezembro de 2014.

OPENCV, O. S. C. V. L. 2014. Disponível em: <http://opencv.org>. Acesso em: 20 de Dezembro de 2014.

OUCHI, K.; DOI, M. Smartphone-based monitoring system for activities of daily living for elderly people and their relatives etc. In: *Proceedings of the 2013 ACM Conference on Pervasive and Ubiquitous Computing Adjunct Publication*. New York, NY, USA: ACM, 2013. (UbiComp '13 Adjunct), p. 103–106. ISBN 978-1-4503-2215-7. Disponível em: <http://doi.acm.org/10.1145/2494091.2494120>.

SEGUIN, C.; LAMOTTE, F. D.; PHILIPPE, J.-L. System services partitioning in ambient assisted living environment. In: *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*. New York, NY, USA: ACM, 2012. (UbiComp '12), p. 778–781. ISBN 978-1-4503-1224-0. Disponível em: <http://doi.acm.org/10.1145/2370216.2370390>.

SOUSA, K. A. O. *Uso de Visão Computacional em Dispositivos Móveis para Auxílio à Travessia de Pedestres com Deficiência Visual*. Tese (Mestrado) — Universidade Presbiteriana Mackenzie, 2013. Disponível em: <http://www.mackenzie.br/fileadmin/PUBLIC/UP_MACKENZIE/servicos_educacionais/stricto_sensu/Engenharia_Eletrica/Kelly_Aparecida_Oliveira_Sousa.pdf>. Acesso em: 11 de Dezembro de 2014.

SOUZA, L. R. de. *Algoritmo para Reconhecimento e Acompanhamento de Trajetória de Padrões em Vídeos*. Tese (Trabalho de Conclusão de Curso) — Universidade Federal do Vale do São Francisco, 2011. Disponível em: <http://www.univasf.edu.br/~ccomp/monografias/monografia_1.pdf>. Acesso em: 09 de Dezembro de 2014.

Apêndice A

Configuração do Ambiente para Desenvolvimento

Esse apêndice tem como principal objetivo auxiliar outros pesquisadores que estejam iniciando o desenvolvimento de aplicativos usando a biblioteca OpenCV.

A.1 Instalação e Configuração

Este projeto foi desenvolvido em um computador com o sistema operacional *Windows* 7. Porém o OpenCV também pode ser utilizado em computadores com o *Windows* 8, o MAC OS ou versões do Linux instalados. Tanto arquiteturas de 32 *bits* como as arquiteturas de 64 *bits* são suportadas.

Para a execução deste projeto, a versão pré-compilada do OpenCV não se apresentou satisfatória, sendo necessário o uso da extensão CMake. A primeira etapa consiste em descarregar a ferramenta Eclipse com o plugin CDT. Para este projeto foi usado o Eclipse Luna, versão para 32 *bits*. É necessário ajustar a variável de sistema PATH e adicionar o caminho da pasta do Eclipse.

Em seguida, é preciso descarregar e instalar as ferramentas: MinGW, CMake e OpenCV. Como feito anteriormente com o Eclipse, é necessário adicionar o caminho das pastas

dos programas instalados na variável de sistema PATH. A configuração do CMake é feita seguindo os procedimentos abaixo:

- Executar a ferramenta CMake-GUI.
- Escolher a pasta do C:/opencv como *source* (origem) e a pasta C:/opencv/build/x86/mingw como destino.
- Pressionar o botão configurar para escolher o *makefile* do MinGW como gerador. Escolher as opções necessárias e clicar em configurar novamente.
- Pressionar o botão gerar e, quando terminar a execução, sair do programa.
- Abrir o modo de linha de comando (cmd.exe) e navegar até a pasta C:/opencv/build/x86/mingw.
- Digitar mingw32-make. Acompanhar o progresso de construção dos binários. Ao fim do processo, reiniciar o computador.

O Eclipse requer a edição das propriedades do projeto para usar as bibliotecas do OpenCV. É necessário adicionar a pasta de origem do OpenCV na opção: *Project* > *Properties* > *C/C++ Build* > *Settings* > *GCC C++ Compiler* > *Includes*. Na opção *Project* > *Properties* > *C/C++ Build* > *Settings* > *MinGW C++ Linker* > *Libraries* é preciso adicionar as bibliotecas uma a uma. O caminho de pesquisa *Library* (-L) deve ser preenchido com a pasta de construção do OpenCV.