UNIVERSIDADE REGIONAL DE BLUMENAU

CENTRO DE CIÊNCIAS EXATAS E NATURAIS

CURSO DE CIÊNCIA DA COMPUTAÇÃO – BACHARELADO

RECONHECIMENTO FACIAL 2D PARA SISTEMAS DE AUTENTICAÇÃO EM DISPOSITIVOS MÓVEIS

LUCIANO PAMPLONA SOBRINHO

BLUMENAU
2010

2010/1-15

LUCIANO PAMPLONA SOBRINHO

RECONHECIMENTO FACIAL 2D PARA SISTEMAS DE AUTENTICAÇÃO EM DISPOSITIVOS MÓVEIS

Trabalho de Conclusão de Curso submetido à Universidade Regional de Blumenau para a obtenção dos créditos na disciplina Trabalho de Conclusão de Curso II do curso de Ciência da Computação — Bacharelado.

Prof. Paulo Cesar Rodacki Gomes, Doutor - Orientador

BLUMENAU
2010

2010/1-15

RECONHECIMENTO FACIAL 2D PARA SISTEMAS DE AUTENTICAÇÃO EM DISPOSITIVOS MÓVEIS

Por

LUCIANO PAMPLONA SOBRINHO

Trabalho aprovado para obtenção dos créditos na disciplina de Trabalho de Conclusão de Curso II, pela banca examinadora formada por:

Presidente:	Prof. Paulo Cesar Rodacki Gomes, Doutor – Orientador, FURB
Membro:	Prof. Dalton Solano dos Reis, Mestre – FURB
Membro:	Prof. Francisco Adell Péricas, Mestre – FURB

Blumenau, 01 de julho de 2010

Dedico este trabalho ao meu pai, irmãos namorada e a todos os amigos, especialmente aqueles que me ajudaram diretamente na realização deste.

AGRADECIMENTOS

Ao meu pai, por sempre acreditar na minha capacidade e investir no meu potencial.

Ao meu irmão, Maurício, por todo o constante apoio e esclarecimento de dúvidas em relação ao desenvolvimento da aplicação.

À minha namorada, pela motivação e compreensão nas horas difíceis.

Ao meu orientador, Paulo César Rodacki Gomes, por abraçar a idéia deste trabalho e pela orientação durante o desenvolvimento.

Aos voluntários, por disponibilizarem seu tempo e imagens para os testes presentes neste projeto.

Defeitos não fazem mal, quando há vontade e poder de os corrigir.

Machado de Assis

RESUMO

Neste trabalho são apresentados meios para apresentar uma forma de efetuar todo o processamento de um reconhecimento facial em um dispositivo móvel. Utilizou-se um iPhone para representar os dispositivos móveis de modo que em tempo real seja possível detectar e reconhecer um indivíduo. É utilizada uma compilação específica da biblioteca OpenCV para o sistema operacional do aparelho para atender ao objetivo proposto no presente trabalho. São utilizadas também técnicas de processamento de imagens e visão computacional para detecção, normalização e reconhecimento biométrico da face. A utilização desta biblioteca e técnicas possibilitaram desenvolver um estudo para verificar a taxa de acerto em relação à variação do ambiente de captura, por exemplo, diferentes rostos, diferentes poses e diferentes condições de iluminação. Observou-se que mesmo variando situações de iluminação e poses na captura das fotos dos rostos é possível atingir uma boa taxa de acerto no reconhecimento facial.

Palavras-chave: Reconhecimento facial. Dispositivos móveis. iPhone. Processamento de imagens. Visão computacional.

ABSTRACT

This work shows ways to present a way to make all the processing of a facial recognition on a mobile device. It was used iPhone to represent mobile devices so that real time is possible to detect and recognize an individual. It use a specific build of the library OpenCV for the operating system of the device to meet the proposed goal of this work. They are also used techniques of image processing and computer vision for detection, biometric recognition and normalization of the face. Using this library and develop techniques allowed a study to verify the accuracy rate on the variation of the capture environment, for example, different faces, different poses and different lighting conditions. It was observed that even varying lighting situations and poses to capture photos of faces is possible to achieve a good hit rate in face recognition.

Key-words: Face recognition. Mobile devices. iPhone. Image processing. Computer vision.

LISTA DE ILUSTRAÇÕES

LISTA DE SIGLAS

3GS – 3G *Speed*

DARPA – *Defense Advanced Research Projects Agency*

NIST – *National Institute of Standarts and Technology*

OpenCV – *Open Computer Vision library*

SUMÁRIO

1 INTRODUÇÃO

Reconhecimento facial é um dos processos mais utilizados pelos seres humanos, pois permite identificar rapidamente qualquer pessoa e assim definir o tipo apropriado de interação com a mesma. Embora seja uma tarefa simples para qualquer pessoa, é um processo de extrema complexidade de implementação em uma máquina.

É notável o crescimento da telefonia móvel atualmente, o qual se percebe verificando a enorme quantidade de aparelhos e as tantas ferramentas disponibilizadas a cada novo modelo lançado. Segundo Agência Nacional de Telecomunicações (2009), a base de usuários no Brasil atualmente chega a aproximadamente 150 milhões.

Com este avanço, dos modelos mais simples aos mais sofisticados, os celulares adquiriram uma capacidade de processamento cada vez mais próxima a de um computador, e deste modo, executam diversas tarefas de um computador, como envio e recebimento de *e-mails* e leitura de arquivos de diversos tipos, passando a ser uma forma simples e cômoda de carregar informações, possivelmente confidenciais e sigilosas.

De acordo com Diário de São Paulo (2008), os aparelhos foram o principal alvo de ladrões durante roubos na capital paulista em 2007. Os usuários lesados, além do furto, correm o risco de ter sua identidade roubada. Conforme Moreira (2009), um estudo no Reino Unido comprovou que 4,2 milhões de britânicos guardam dados em seus celulares que poderiam ser usados para roubo de identidade, e apenas 6 em cada 10 usuários usam senha para barrar o acesso às informações.

Decorrente destes fatos, medidas de segurança tornam-se necessárias para proteger os dados armazenados, permitindo acesso apenas ao proprietário. Uma solução seria a utilização de um programa de autenticação com análise de imagens faciais.

Diante do exposto, propõe-se desenvolver um sistema capaz de reconhecer faces em imagens capturadas a partir de uma câmera de celular, e compará-las com outras previamente inseridas em uma base existente, possibilitando a criação de uma ferramenta de autenticação de usuários em dispositivos móveis para permissão de acesso as suas funções.

Atualmente, existem diversas técnicas para resolver esta questão, porém, pouco se vê aplicado aos dispositivos móveis. Portanto, a ferramenta deve aplicar métodos de forma específica ainda pouco explorada, permitindo ser utilizada como base para outros estudos, com aplicações em diferentes cenários além do proposto.

1.1 OBJETIVOS DO TRABALHO

O objetivo deste trabalho é desenvolver um protótipo capaz de autenticar indivíduos através da face em dispositivos móveis, especificamente o iPhone, utilizando técnicas de processamento de imagens e visão computacional.

Os objetivos específicos do trabalho são:

a) utilizar técnicas de processamento de imagens para melhorar a qualidade das imagens obtidas (i.e. filtragem de ruído e normalização da iluminação);
b) localizar a face nas imagens obtidas;
c) normalizar a pose das faces encontradas (i.e. padronização do alinhamento das
d) faces);
e) extrair as características das faces utilizando a Análise de Componentes Principais (PCA);
f) utilizar métricas de similaridade para comparar as faces de entrada com as faces conhecidas.

1.2 ESTRUTURA DO TRABALHO

O presente trabalho está estruturado em quatro capítulos. O segundo capítulo contém a fundamentação teórica necessária para a compreensão deste trabalho. Nele são abordados tópicos relacionados sobre o reconhecimento biométrico e sistemas de reconhecimento facial 2D, processamento de imagens, visão computacional e sobre o dispositivo móvel iPhone. Alguns trabalhos correlatos à aplicação também são destacados.

O terceiro capítulo trás comentários a respeito do desenvolvimento da ferramenta, apontando os requisitos principais do problema trabalhado, a especificação contendo diagramas de casos de uso, seqüência e classes. No mesmo capítulo são comentados detalhes da implementação apresentando técnicas e ferramentas utilizadas, logo a seguir a operacionalidade da aplicação e por fim apresentados os resultados e discussão.

O quarto capítulo refere-se às conclusões e extensões do trabalho.

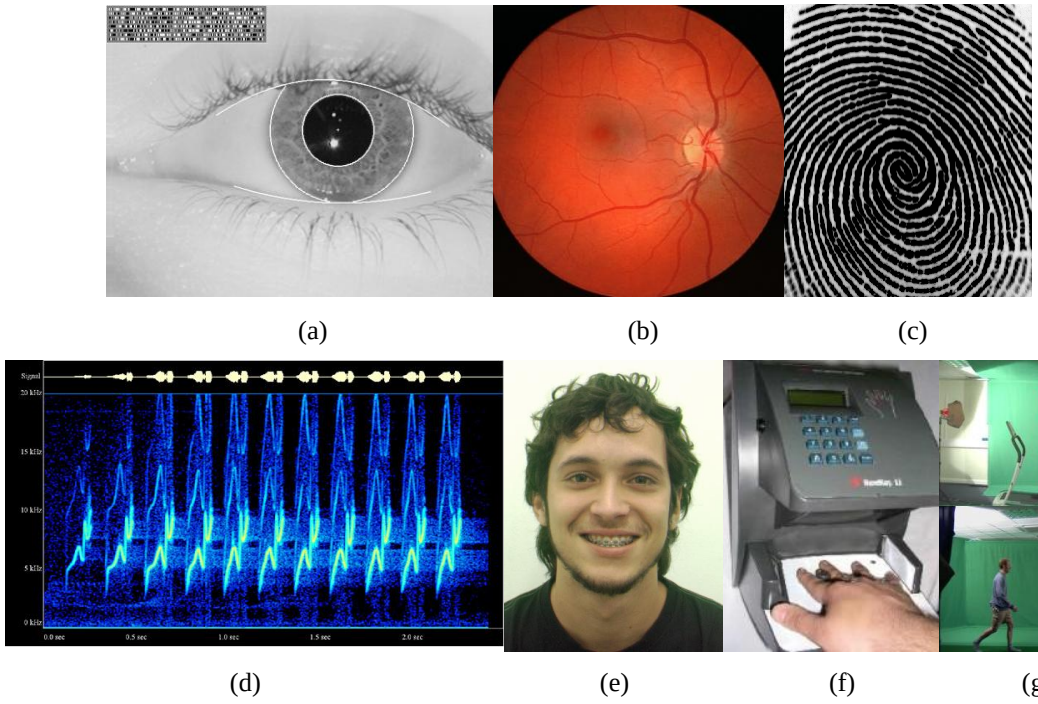
2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, aspectos teóricos relacionados ao desenvolvimento e entendimento do trabalho são apresentados. São abordados tópicos sobre o reconhecimento biométrico e sistemas de reconhecimento facial 2D, processamento de imagens, visão computacional e sobre o dispositivo móvel iPhone. Por fim, alguns trabalhos correlatos são comentados.

2.1 RECONHECIMENTO BIOMÉTRICO

O reconhecimento biométrico corresponde ao uso de características físicas ou comportamentais para a identificação de indivíduos (e.g. íris (DAUGMAN, 1993; DAUGMAN, 2004), retina, impressão digital (LEMES, 2007), voz, face, geometria da mão (BEYOND IF SOLUTIONS, 2009), forma de andar (ISIS RESEARCH GROUP, 2003) e DNA, ver Figura 1). Seres humanos utilizam esta forma de reconhecimento, mesmo que inconscientemente, através de características como a voz, a face e a forma de andar (NATIONAL SCIENCE & TECHNOLOGY COUNCIL SUBCOMMITTEE ON BIOMETRICS, 2010).

Atualmente, a identificação de indivíduos é realizada através de números, senhas ou cartões. Por exemplo, o Cadastro de Pessoas Físicas (CPF) é um número identifica indivíduos da população brasileira, e um caixa eletrônico de um banco identifica um cliente através de um cartão magnético e uma senha. O problema é que estes tipos de informação/objeto podem ser esquecidos, roubados, duplicados ou perdidos, enquanto sistemas biométricos não têm estes problemas e ainda assim são capazes de garantir altos níveis segurança, variando de acordo com a característica utilizada (JAIN et al., 2004). De acordo com Campos (2001), métodos de identificação de pessoas sempre foram muito importantes para a sociedade, e, assumindo-se que não existem pessoas completamente idênticas, extingue-se a necessidade da utilização de documentos de identificação quando se dispõe de métodos capazes de diferenciar pessoas sem confundi-las com seus semelhantes.



Fonte: http://www.google.com.br/imghp?hl=pt-BR&tab=wi.

Figura 1 – Exemplos de (a) íris, (b) retina, (c) impressão digital, (d) voz, (e) face, (f) geometria da mão, e (g) forma de andar

Cada característica biométrica pode ser avaliada de acordo com o seu comportamento e os recursos computacionais exigidos (BOLLE, 2003; JAIN, ROSS e PRABHAKAR, 2004a). As formas de avaliação são as seguintes:

a) universalidade, que avalia se todas as pessoas possuem a característica em questão;
b) singularidade, que avalia se a característica é suficientemente diferente para várias pessoas;
c) permanência, que avalia se a característica permanece invariante com o passar do tempo;
d) coletabilidade, que indica a facilidade em representar a característica computacionalmente;
e) desempenho, que avalia a precisão da decisão e o tempo e recursos computacionais requeridos;
f) aceitabilidade, que avalia a disposição das pessoas em aceitar a identificação por determinada característica no dia-a-dia; e
g) contornabilidade, que avalia a facilidade em enganar o sistema.

A classificação de algumas características biométricas é mostrada no Quadro 1 (JAIN;

ROSS; PRABHAKAR, 2004). Cada coluna representa uma das formas de avaliação, seguindo a numeração realizada anteriormente. Como podem ser observadas, as características mais seguras costumam apresentar baixa aceitabilidade, por serem invasivas ou precisarem de alguma cooperação do usuário. A face e a voz apresentam alta aceitabilidade, mas o desempenho é inferior ao das demais características.

Para o iPhone, as características que podem ser coletadas são a face e a voz. Considerando que os recursos computacionais exigidos para o reconhecimento por voz são maiores do que para o reconhecimento facial, e o desempenho das duas características é semelhante, a face é a característica mais adequada para um aplicativo de reconhecimento biométrico para o iPhone.

Quadro 1 – Avaliação de características biométricas

Característica	**a**	**b**	**d**	**d**	**e**	**f**	**g**
Face	Alta	Baixa	Média	Alta	Baixa	Alta	Alta
Impressão Digital	Média	Alta	Alta	Média	Alta	Média	Média
Geometria da mão	Média	Média	Média	Alta	Média	Média	Média
Íris	Alta	Alta	Alta	Média	Alta	Baixa	Baixa
Retina	Alta	Alta	Média	Baixa	Alta	Baixa	Baixa
Voz	Média	Baixa	Baixa	Média	Baixa	Alta	Alta
DNA	Alta	Alta	Alta	Baixa	Alta	Baixa	Baixa

2.1.1 Sistemas de Reconhecimento Facial 2D

A crescente necessidade de um sistema de reconhecimento automático e confiável de indivíduos tem alimentando uma atenção especial da comunidade científica pelo reconhecimento através de características biométricas, entre elas a face. Graças aos avanços computacionais nas últimas décadas, diferentes sistemas para desempenhar esta tarefa têm sido propostos para diferentes aplicações.

Atualmente, existem poucas soluções de reconhecimento facial no mercado e raras exceções aplicadas em dispositivos móveis. Chellappa (CHELLAPA, WILSON e SIROHEY, 1995) revelam algumas possíveis aplicações na área, como:

a) identificação pessoal para banco, passaporte, fichas criminais;
b) monitoramento de multidões;
c) criação de retrato falado;

d) busca em fichas criminais;
e) envelhecimento computadorizado para auxiliar busca por desaparecidos;
f) interfaces perceptuais homem-máquina com reconhecimento de expressões faciais;
g) sistemas de segurança e controle de acesso.

Um sistema de autenticação através de reconhecimento da face requer processos eficientes, que tragam resultados precisos. Desta forma, visando a melhora da possibilidade de êxito em um sistema de reconhecimento, Campos (2001) expõe a necessidade de primeiramente segmentar as imagens faciais, para que somente essas sejam tratadas, permitindo que objetos ao redor do sujeito a ser reconhecido possam ser descartados. Em geral, elementos de processamento de imagens e de reconhecimento de padrões são utilizados para extrair e interpretar tais informações.

2.2 PROCESSAMENTO DE IMAGENS

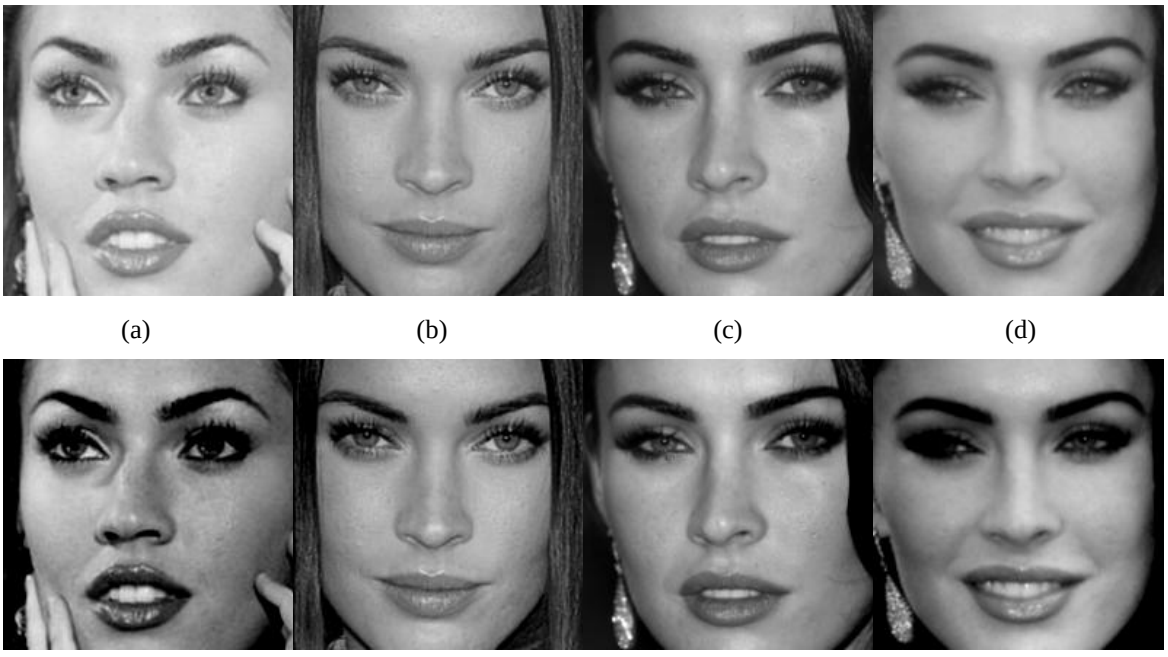
Processar uma imagem consiste em transformá-la sucessivamente com o objetivo de extrair mais facilmente a informação desejada nela contida, melhorando o aspecto visual de certas feições estruturais para o analista humano e fornecendo subsídios para sua interpretação. A evolução da tecnologia de computação digital, bem como o desenvolvimento de novos algoritmos para lidar com sinais bidimensionais está permitindo uma gama de aplicações cada vez maior.

O primeiro passo do processo é a aquisição da imagem. Segundo Gonzalez e Woods (2002), as principais fontes de ruídos surgem durante a aquisição e/ou transmissão da imagem, sendo afetada por uma série de fatores, tais como qualidade dos equipamentos e situação do ambiente.

Após a obtenção da imagem digital, a próxima tarefa é pré-processar a mesma, visando um aumento de chances de sucesso nos processos seguintes. O pré-processamento envolve técnicas de realce de contrastes, remoção de ruídos, entre outras.

Para o reconhecimento facial, duas técnicas de processamento de imagens são necessárias para melhorar a qualidade da imagem de entrada: filtragem de ruído e normalização da iluminação. O objetivo da filtragem de ruído é eliminar variações na imagem de entrada obtidas no momento da aquisição, causadas principalmente pela

imprecisão do sensor de captura. Por sua vez, a normalização da iluminação é responsável por amenizar as variações na imagem causadas por diferentes fontes de iluminação no momento da aquisição, como mostrado na Figura 2.

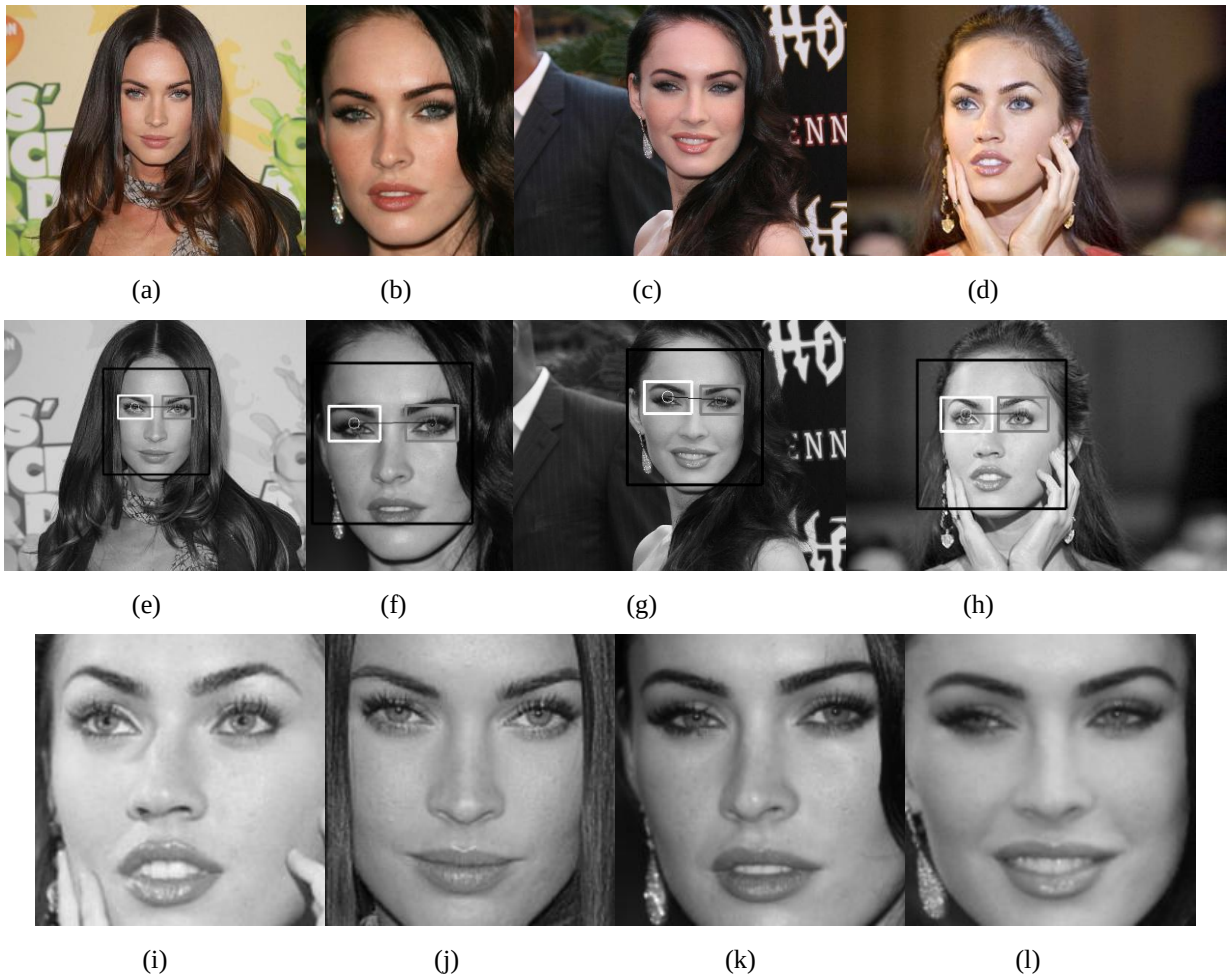


Fonte: http://www.google.com.br/imghp?hl=pt-BR&tab=wi.

Figura 2 – Exemplos de normalização de iluminação: faces originas (a)-(d) e seus respectivos resultados da normalização de iluminação (e)-(h)

2.3 VISÃO COMPUTACIONAL

De acordo com Shapiro e Stockman (2001), o objetivo da visão computacional é analisar uma imagem de entrada visando extrair informações relevantes a um determinado problema. Em um sistema de reconhecimento facial, várias etapas requerem este tipo de processamento, entre elas a localização da face (Figuras 3(e)-3(h)), a normalização da mesma (Figuras 3(i)-3(l)), e a extração de características da face.



Fonte: http://www.google.com.br/imghp?hl=pt-BR&tab=wi.

Figura 3 – Exemplos de detecção e normalização de pose: imagens originais (a)-(d), suas respectivas detecções facial (e)-(h) e seus respectivos resultados para a normalização de pose e escala (i)-(l).

Diferentes técnicas para a localização da face foram propostas na literatura (HJELMAS e LOW, 2001; HSU, ABDEL-MOTTALEB e JAIN, 2002; YANG, KRIEGMAN e AHUJA, 2002), e a abordagem mais bem sucedida para a realização desta tarefa foi proposta por Viola e Jones (2004). Este sistema possui atualmente o melhor custo benefício entre velocidade e taxa de acerto de localização da face, e uma implementação desta técnica está disponível na biblioteca OpenCV (OPENCV, 2010).

A normalização da face consiste em obter o ângulo de inclinação da face a partir da imagem de entrada, e retificar a mesma de maneira que a pose da face seja a mesma das imagens previamente cadastradas.

A extração de características da face é a etapa de obtenção de dados relevantes para o

reconhecimento. Kirby e Sirovich (KIRBY e SIROVICH, 1990; SIROVICH e KIRBY, 1987) empregaram a Análise de Componentes Principais (PCA) para extrair estas características e vários trabalhos surgiram na literatura baseados nesta idéia (HESHER, SRIVASTAVA e ERLEBACHER, 2002; MOON e PHILLIPS, 2001; TURK e PENTLAND, 1991). Nesta abordagem, faces canônicas são geradas, e as faces a serem reconhecidas são representadas como uma combinação linear destas faces canônicas. Nesta combinação, cada face canônica recebe um peso e as faces de entrada são representadas como um conjunto de pesos. Como a quantidade de faces canônicas geralmente utilizada é bem menor que a quantidade de *pixels* das faces de entrada, o reconhecimento pode ser realizado de maneira mais rápida e eficiente.

2.4 RECONHECIMENTO FACIAL 2D

Por padrão, um sistema de reconhecimento biométrico possui os seguintes estágios:

a) aquisição de imagens em formato digital;
b) pré-processamento para melhoramento e padronização da imagem obtida;
c) correspondência para comparar a imagem adquirida com uma ou mais imagens da base de dados;
d) avaliação dos resultados da correspondência para realizar o reconhecimento.

2.4.1 Aquisição

A aquisição de imagens é a etapa do reconhecimento facial. Para a execução desta etapa do processo é utilizada uma câmera como fonte de aquisição. A Figura 4 mostra alguns exemplos de imagens adquiridas pelo iPhone.

Figura 4 – Exemplos de imagens adquiridas com o iPhone

2.4.2 Pré-processamento

A etapa de pré-processamento é responsável pela padronização das imagens de face de entrada. Para isto, esta etapa é dividida em três estágios:

a) detecção facial, em que a face é localizada na imagem;
a) normalização de pose e escala, em que inclinações e variações de tamanho nas faces detectadas são corrigidas;
b) normalização de iluminação, onde variações na imagem causadas por diferentes ambientes e diferentes fontes de iluminação são amenizadas.

2.4.3 Correspondência e Avaliação

Vários métodos foram propostos para o reconhecimento facial (CHELLAPA, WILSON e SIROHEY, 1995). O método escolhido é baseado em autofaces (TURK e

PENTLAND, 1991), que permite a realização de grandes quantidades de correspondências em um curto espaço de tempo com um mínimo de recursos computacionais.

2.5 IPHONE

O iPhone é mais que um simples telefone (APPLE, 2009a) (ver Figura 5). É um aparelho que combina três dispositivos em um: celular, iPod *widescreen* e dispositivo para internet. Na versão mais recente, 3GS possui diversas funcionalidades: controle por voz, até 32 Gb para armazenamento, câmera de 3 megapixels e gravação de vídeo. Dotado de um processador de 600 megahertz e 256 megabytes de memória RAM, disponibiliza aos desenvolvedores uma série de ferramentas para criação de aplicativos (APPLE, 2009b).

O pacote de desenvolvimento do SDK do iPhone possui várias ferramentas, entre elas:

a) xCode: ambiente de desenvolvimento, editor de código e depurador gráfico;
b) iPhone Simulator: ferramenta que simula o iPhone no computador *desktop* para executar, testar e depurar a aplicação;
c) Interface Builder: permite projetar interfaces utilizando o recurso arrastar e soltar.

Além de recursos técnicos para o desenvolvimento de aplicações, o iPhone *Developer Program* (APPLE, 2009b) fornece ferramentas sofisticadas para efetuar testes e depurações de código no iPhone. Com a aplicação finalizada, ainda é possível distribuí-la na *App Store*, loja virtual de aplicativos para esta plataforma, assim disponibilizando o aplicativo desenvolvido para milhares de usuários do aparelho.

Figura 5 – iPhone

2.6 TRABALHOS CORRELATOS

Poucos sistemas disponíveis no mercado propõem-se a mesma tarefa que o presente trabalho. A maior parte do material encontrado não é aplicada a dispositivos móveis. Foram escolhidos três os quais mais se enquadram nos principais objetivos deste trabalho. Os seguintes projetos foram selecionados: “Face Match” (POLAR BEAR FARM, 2009), “iFace” (ANIMETRICS, 2009) e “Detecção e Reconhecimento Facial em Seqüências de Vídeo” (FIOR, 2008).

2.6.1 Face Match

Trata-se de uma aplicação que traz a detecção facial e reconhecimento para o iPhone (POLAR BEAR FARM, 2009). O Face Match (FM) tenta detectar e reconhecer pessoas em fotografias tiradas ou armazenadas pelo aparelho.

As pessoas reconhecidas podem ser classificadas pelos nomes (Figura 6) e, desta

forma, o sistema utiliza estas marcações para melhorar a capacidade de reconhecimento. Cada vez que uma imagem é capturada, o usuário pode verificar se os dados demonstrados correspondem à realidade, ou não. Em caso negativo, basta corrigir, tornando o algoritmo mais eficiente em novos reconhecimentos.

FM não realiza o processamento para detecção e reconhecimento no próprio aparelho. As imagens escolhidas são enviadas para um servidor, onde são processadas. Esta aplicação encontra-se disponível na *App Store*.

Figura 6 – Face Match

2.6.2 iFace

Da mesma forma que o FM, o iFace é um aplicativo de identificação facial que aproveita o iPhone e sua interface inovadora (ANIMETRICS, 2009), porém não executa o processamento no aparelho, e sim em servidores próprios, como ilustrado na Figura 7. É uma aplicação nativa do aparelho que utiliza protocolos padrão da Internet para enviar imagens faciais para o servidor da Animetrics, Face Identity Management System (FIMS), para enfrentar o processamento biométrico.

O aplicativo tem duas funções principais:

a) interface para um sistema de identificação biométrica para uso em segurança, inclusive policiais, militares e privadas;

b) jogo biométrico que avalia a similaridade entre o usuário e pessoas famosas.

Quando utilizada como uma aplicação de segurança, FM pode ser usada para identificar suspeitos que fazem parte de uma lista de procurados ou pessoas que estão em uma lista autorizada para a confirmação do acesso. Com a simplicidade da interface do iPhone, os operadores são capazes de capturar imagens de alta qualidade, de forma rápida e fácil, para fins de identificação.

A rede integrada do iPhone permite o uso de protocolos padrão e um processo simplificado para a realização de identificação facial. O iFace está atualmente em fase de avaliação pelo NIST e pelo DARPA para uso militar.

Figura 7 – iFace

2.6.3 Detecção e Reconhecimento Facial em Seqüências de Vídeo

O principal objetivo deste projeto foi a criação de um sistema que utilize não apenas uma única imagem para o reconhecimento facial, mas uma combinação de imagens da mesma face obtida através de uma seqüência de vídeo (FIOR, 2008), como mostrado na Figura 8.

Este trabalho utilizou como base para o reconhecimento facial o método baseado em autofaces (TURK e PENTLAND, 1991) a partir da Análise de Componentes Principais (PCA), técnica mais utilizada para reduzir o espaço de representação de bases de dados multidimensionais.

Neste projeto, o problema foi restringido para vídeos contendo apenas uma pessoa e, futuramente, pretende-se integrar um sistema de rastreamento para permitir o reconhecimento simultâneo de vários indivíduos.

Os resultados finais apresentados mostram que a utilização do método proposto melhorou em até 20% a taxa de reconhecimento se comparado a um sistema de identificação clássico, que avalia as faces individualmente.

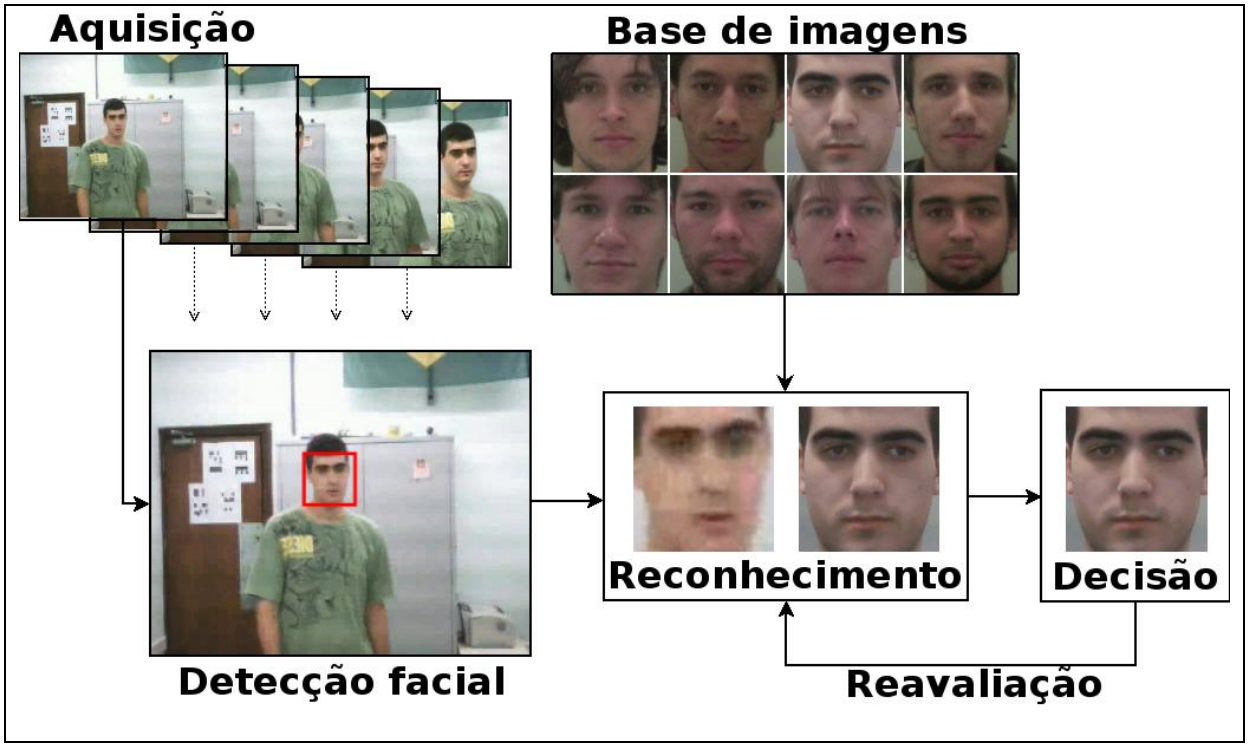


Figura 8 – Detecção e Reconhecimento Facial em Seqüências de Vídeo

3 DESENVOLVIMENTO

Neste capítulo são detalhadas as etapas do desenvolvimento da ferramenta. São apresentados os principais requisitos, a especificação, a implementação e por fim são listados resultados e discussão.

3.1 REQUISITOS PRINCIPAIS DO PROBLEMA A SER TRABALHADO

O protótipo foi desenvolvido tendo em vista os seguintes requisitos:

a) disponibilizar uma interface para permitir o cadastro da face do usuário como usuário do sistema na base de faces (Requisito Funcional - RF);
b) disponibilizar uma interface para permitir a autenticação do usuário no sistema (RF);
c) efetuar tratamento de luminosidade, foco e pose nas imagens, tanto para a base de faces, quanto para a autenticação (RF);
d) gerar alerta caso a imagem para cadastro na base de faces não seja adequada (RF);
e) informar se a autenticação teve, ou não, sucesso (RF);
f) utilizar linguagem de programação Objective-C (Requisito Não-Funcional - RNF);
g) utilizar ambiente xCode (RNF).

3.2 ESPECIFICAÇÃO

A especificação do sistema utiliza alguns dos diagramas da *Unified Modeling Language* (UML) em conjunto com a ferramenta Enterprise Architect 8.0.856 para a elaboração dos diagramas de casos de uso, de classes e de seqüência. Alguns diagramas estão em sua forma resumida para melhor visualização.

3.2.1 Diagrama de casos de uso

A Figura 9 apresenta o diagrama de casos de uso com as principais interações do usuário com o sistema.

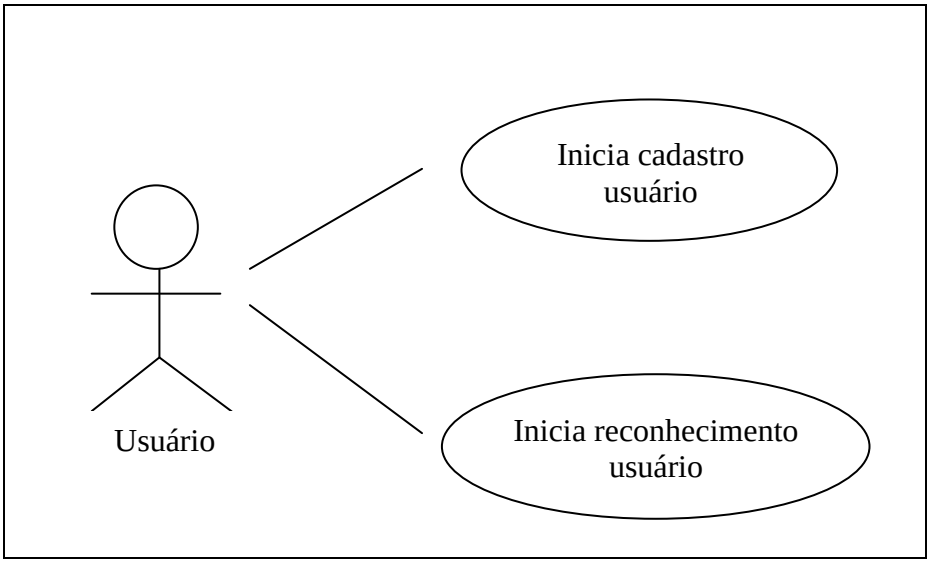


Figura 9 - Diagrama de casos de uso

O caso de uso `Inicia cadastro usuário` (Quadro 2) descreve como o usuário pode selecionar o inicio do processo de cadastramento de um usuário. Este caso de uso possui um cenário principal e um alternativo.

Inicia cadastro usuário: possibilita ao usuário iniciar o processo de cadastramento de um usuário.	
Pré-condição	O menu inicial deve ser exibido pelo sistema.
Cenário principal	1) O usuário seleciona o botão CADASTRAR USUÁRIO. 2) O sistema exibe uma nova tela com opções para captura. 3) O usuário seleciona o meio de captura clicando no botão BUSCAR ou CAPTURAR. 4) O usuário captura a imagem. 5) O sistema exibe o resultado junto na imagem capturada pelo meio de captura selecionado.
Fluxo Alternativo 01	No passo 3, caso o meio de captura seleciona esteja indisponível, o outro meio é selecionado automaticamente.
Pós-condição	O sistema exibe os resultados com sucesso.

Quadro 2 – Caso de uso inicia cadastro usuário

O segundo caso de uso `Inicia reconhecimento usuário` (Quadro 3) trata do

momento em que o usuário pretende selecionar o inicio do processo de reconhecimento facial de um usuário. Este caso de uso possui um cenário principal e um alternativo.

Inicia reconhecimento usuário: possibilita ao usuário iniciar o processo de reconhecimento facial de um usuário.	
Pré-condição	O menu inicial deve ser exibido pelo sistema.
Cenário principal	1) O usuário seleciona o botão RECONHECER USUÁRIO. 2) O sistema exibe uma nova tela com opções para captura. 3) O usuário seleciona o meio de captura clicando no botão BUSCAR ou CAPTURAR. 4) O usuário captura a imagem. 5) O sistema exibe o resultado junto na imagem capturada pelo meio de captura selecionado.
Fluxo Alternativo 01	No passo 3, caso o meio de captura seleciona esteja indisponível, o outro meio é selecionado automaticamente.
Pós-condição	O sistema exibe os resultados com sucesso.

Quadro 3 – Caso de uso inicia reconhecimento usuário

3.2.2 Diagrama de classes

O diagrama de classes apresentado na Figura 10 exibe as principais classes utilizadas no sistema com os atributos. Algumas classes, responsáveis pela declaração dos protótipos das funções, foram abstraídas para uma melhor visualização do digrama, não comprometendo o entendimento do mesmo.

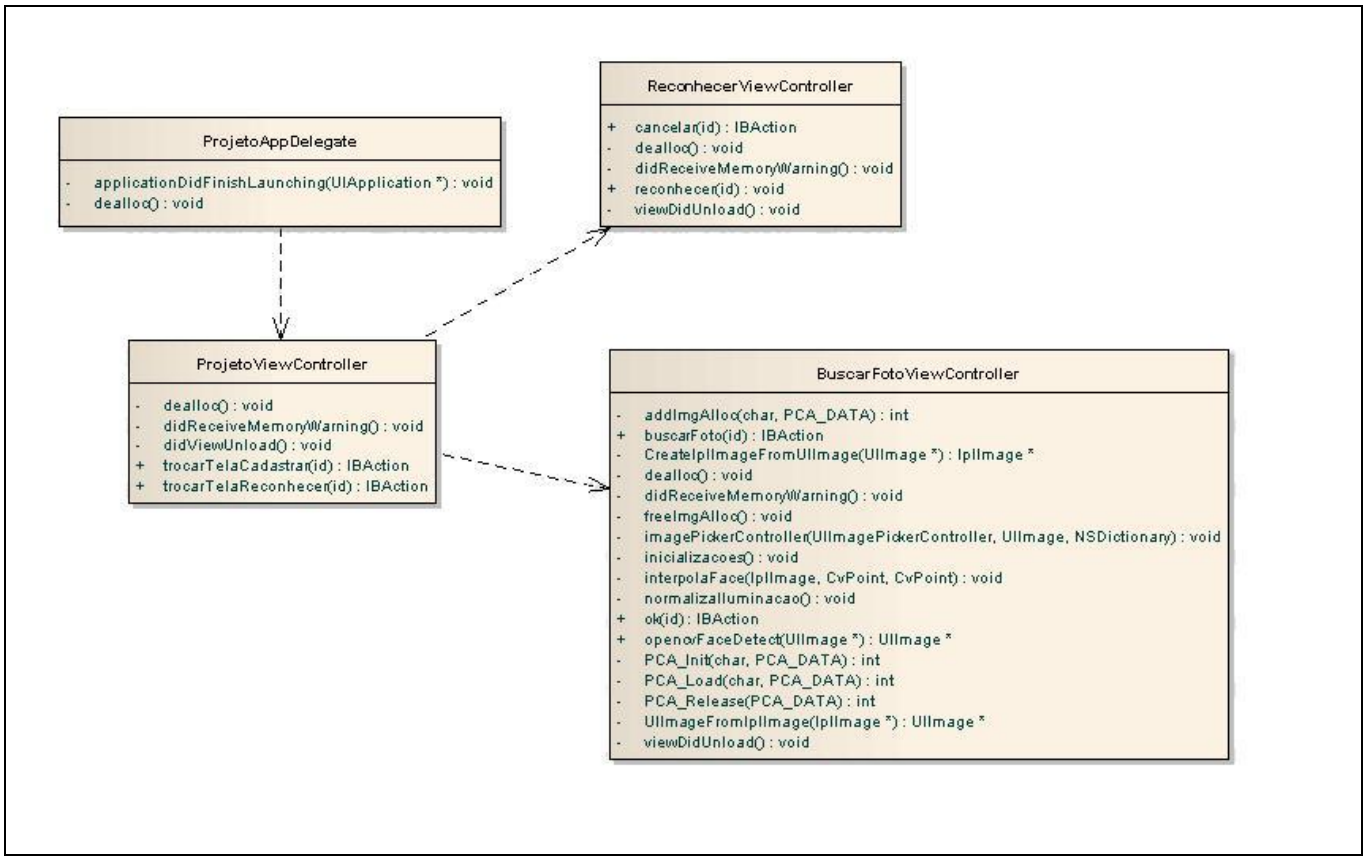


Figura 10 – Diagrama de Classes

3.2.2.1 Classe `ProjetoAppDelegates`

A classe `ProjetoAppDelegate` é responsável por responder a muitas das mensagens da aplicação. Mais especificamente, a classe deve atender as seguintes necessidades:

a) criar um objeto `Window` e exibi-lo para o usuário;

b) inicializar os dados do objeto;

c) responder ao pedido de `quit`;

d) lidar com advertências de baixa memória.

Entre estes itens, o primeiro é de extrema importância, pois é quem informa que o programa teve início. Também é responsável por adicionar na tela uma visão da classe `ProjetoViewController`, que corresponde a tela inicial do sistema.

3.2.2.2 Classe `ProjetoViewController`

A classe `ProjetoViewController` tem como função principal alternar entre as duas opções básicas do sistema, cadastro e reconhecimento. Essa classe implementa e gerencia os eventos dos botões, fazendo a conexão com a janela seguinte, conforme seleção do usuário. Também possui métodos para gerenciar advertências de memória e desalocar variáveis.

3.2.2.3 Classe `ReconhecerViewController`

A classe `ReconhecerViewController` tem como finalidade básica apresentar a interface e gerenciar o processo de reconhecimento facial. Nesta classe está presente o método responsável por iniciar o processo de reconhecimento, `reconhecer`. Este método, em conjunto com as demais operações efetuadas na classe `BuscarFotoViewController`, apresenta o resultado final proposto pela ferramenta, reconhecer faces. A classe também controla a seleção de imagem de um indivíduo para o início do processo.

3.2.2.4 Classe `BuscarFotoViewController`

A classe `BuscarFotoViewController` contém a maior parte das funções específicas do sistema. Nela são encontrados os métodos de conversão de imagem entre os formatos `IplImage` e `UIImage`, utilizados pela biblioteca OpenCV e pelo Objective-C, respectivamente como também os métodos para detecção facial, normalização de pose e iluminação, e criação do arquivo de treino com as imagens adquiridas.

3.2.3 Diagrama de seqüência

Figura 11 mostra o diagrama de seqüência, que permite visualizar os casos de uso `Inicia cadastro de usuário` e `Inicia reconhecimento usuário` de forma resumida para uma melhor visualização. A partir do momento em que o usuário inicia a aplicação, o sistema passa a apresentar a tela inicial onde o usuário alterna entres as

funcionalidades. Quando o usuário seleciona o botão `Cadastrar Usuário`, o processo de aquisição de imagens é iniciado, e o método `trocarTelaCadastrar` é executado fazendo a troca entre as telas. Em seguida o usuário entra no *loop* de iterações do cadastro de faces até ser retornado para a tela inicial com o fim do processo.

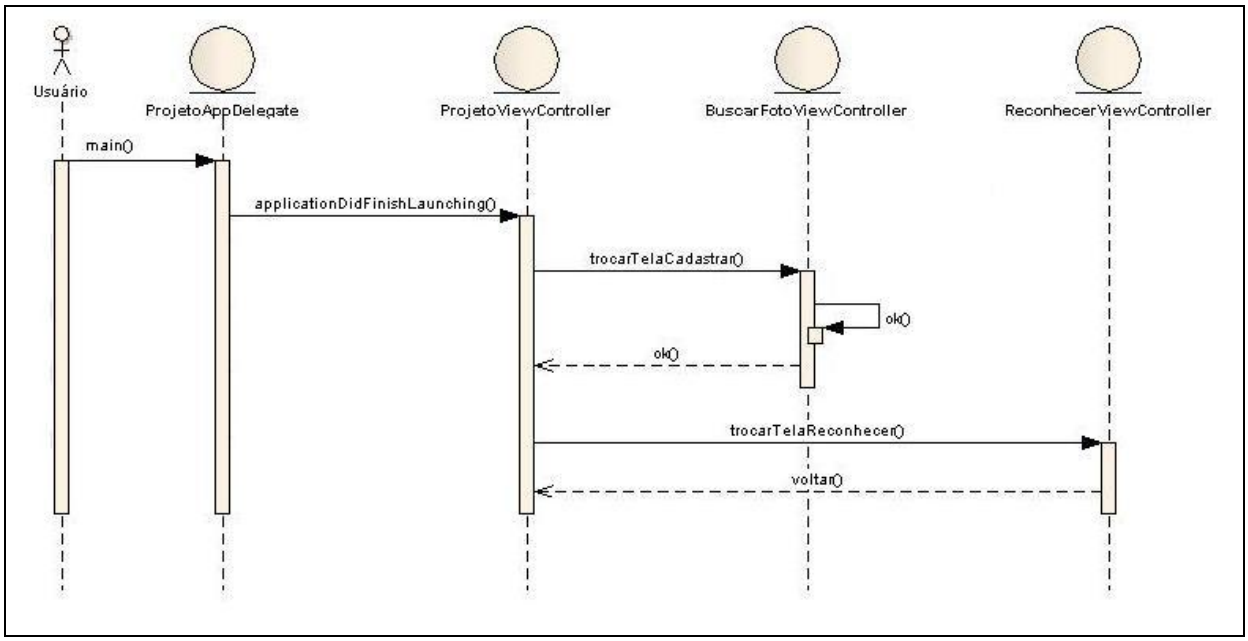


Figura 11 – Diagrama de Seqüência.

Tendo novamente a tela inicial como ponto de partida, o usuário pode selecionar o botão `Reconhecer Usuário` e, desta forma, ativar os procedimentos para o inicio do reconhecimento facial, e retornar a tela inicial quando finalizar o mesmo.

3.3 IMPLEMENTAÇÃO

Nas subseções seguintes são ilustrados detalhes das técnicas e ferramentas utilizadas e a operacionalidade da implementação.

3.3.1 Técnicas e ferramentas utilizadas

Para a implementação da ferramenta foi utilizada a linguagem de programação Objective-C juntamente com iPhone SDK 3.1. O ambiente de desenvolvimento usado foi o Xcode Version: 3.1.4. A biblioteca OpenCV V2.0.0 foi obtida e compilada a partir do

trabalho de (NIWA, 2009).

3.3.1.1 Detecção Facial

O objetivo da detecção facial é encontrar a posição da face em uma imagem de entrada e, para esta etapa, foi utilizada a câmera disponibilizada no iPhone, com resolução de 3 megapixels e foco automático (APPLE, 2009a). Algumas restrições aplicadas durante a aquisição são necessárias para garantir um bom funcionamento do sistema. A face deve:

e) estar completamente visível (i.e. sem oclusões);
f) estar posicionada frontalmente;
g) apresentar expressão neutra (i.e. sem variações na face causadas por expressões faciais).

As variações causadas na imagem de entrada quando estas restrições não são aplicadas afetam consideravelmente o resultado dos estágios seguintes de pré-processamento e reconhecimento.

Entre os vários métodos propostos para realizar esta tarefa (HJELMAS e LOW, 2001; HSU, ABDEL-MOTTALEB e JAIN, 2002; VIOLA e JONES, 1991; YANG e KRIEGMAN, 2002), neste trabalho utilizamos a abordagem proposta por Viola e Jones (VIOLA e JONES, 1991), cuja implementação está disponível na biblioteca OpenCV. Esta abordagem foi escolhida porque apresenta os melhores resultados em termos de localização e velocidade quando comparada com as outras abordagens presentes na literatura. Além disso, a quantidade de processamento computacional necessária é menor do que em outras abordagens, o que torna essa abordagem ideal para a utilização em aplicações para dispositivos móveis. A Figura 12 mostra alguns exemplos do método de Viola e Jones (2004) aplicado em imagens adquiridas pelo iPhone.

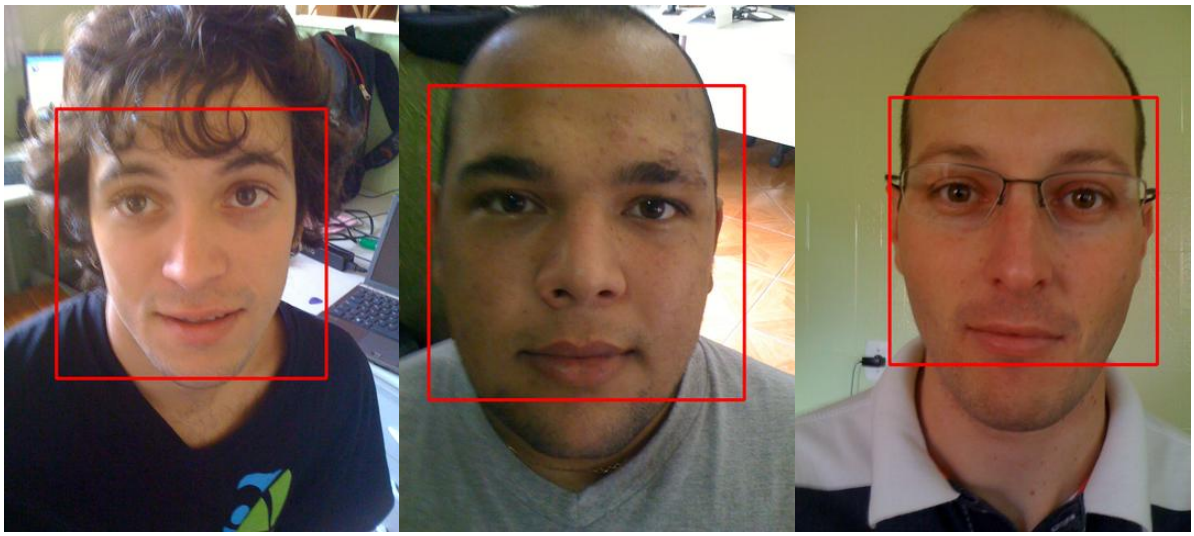


Figura 12 – Exemplos do resultado da detecção facial em imagens do iPhone

Para realizar a detecção, uma subjanela percorre a imagem em busca de faces. Para cada subjanela, um conjunto de classificadores é aplicado para determinar se a região sob a subjanela é uma face ou não. Para que uma subjanela seja considerada face, ela deve ser aceita por todos os classificadores. Se apenas um classificador rejeitar a subjanela, os demais classificadores não são aplicados para poupar tempo de processamento, uma vez que a maioria das subjanelas analisadas é de regiões que não são faces.

3.3.1.2 Normalização de Pose e de Escala

Nesse estágio, a mesma técnica utilizada anteriormente para localizar a face na imagem de entrada é utilizada agora para localizar os olhos na região da face. Com a posição dos olhos obtida, o ângulo formado pela reta que liga os centros dos olhos é utilizado para corrigir uma eventual inclinação apresentada pela face detectada, como mostrada na Figura 13.

Figura 13 – Exemplos da correção da inclinação de uma face detectada

O quadro 4 apresenta o código fonte referente ao método de detecção facial.

```
...
// Encontra seqüências de faces
CvSeq *faces = cvHaarDetectObjects(img, cascade_face, storage, 1.1, 2,
CV_HAAR_DO_CANNY_PRUNING, cvSize(TAM_MIN_FACE, TAM_MIN_FACE));
        // Verifica se alguma face foi encontrada
        if(!faces || faces->total == 0){
                [labelStatus setTag:@"FACE NAO DETECTADA"];
                detectada=false;
                return [self UIImageFromIplImage:img];
        }
        else {
                // Obtém as informações da primeira face encontrada
                CvRect *r = (CvRect *) cvGetSeqElem(faces, 0);
                // Salva coordenadas da face encontrada
                pt_f.x = r->x;
                pt_f.y = r->y;
                s_f.x = r->width;
                s_f.y = r->height;

                // Encontra maior face detectada da imagem
                for(i=1; i < faces->total; i++) {
                        // Obtém as informaçõees da i-ésima face encontrada
                        CvRect *r = (CvRect *) cvGetSeqElem(faces, i);
                        // Caso maior, salva coordenadas como face atual
                        if(r->width > s_f.x && r->height > s_f.y) {
                                pt_f.x = r->x;
                                pt_f.y = r->y;
                                s_f.x = r->width;
                                s_f.y = r->height;
                        }
                }

                // Recorta a metada superior da face para procurar por olhos
                IplImage *temp = cvCreateImage(cvSize(s_f.x,s_f.y/2), 8, 1/*3*/);
```

```
                for(i=0; i < s_f.y/2; i++)
                        for(j=0; j < s_f.x; j++)
                                CV_IMAGE_ELEM(temp, uchar, i, j) = CV_IMAGE_ELEM(img, uchar,
                                                                        pt_f.y+i, pt_f.x+j);

...
}
```

Quadro 4 – Trecho do código fonte do método `opencvFaceDetect para detectar face`

Como pode ser observada na Figura 12, a região selecionada como face pelo detector varia tanto em tamanho como no posicionamento sobre a face. Para padronizar isto, a face é redimensionada para que a distância entre os centros dos olhos seja igual para todas as faces, e a face é posicionada de maneira que os centros dos olhos de todas as faces estejam localizados no mesmo ponto. No Quadro 5 pode-se observar o trecho do código fonte referente ao método de utilizado para normalização da pose.

```
void interpolaFace(IplImage* img, CvPoint p1, CvPoint p2) {
        int x1, y1, x2, y2, xtmp, ytmp, i, j;
        float d, l, a, x[4], y[4], xI[3], yI[3], X, Y, R1, R2, R3, R4, S1, S2, S3, S4, B1, B2,
B3, B4, B;
        x1 = p1.x;
        y1 = p1.y;
        x2 = p2.x;
        y2 = p2.y;
        if(x1 > x2) {
                xtmp = x1; x1 = x2; x2 = xtmp;
                ytmp = y1; y1 = y2; y2 = ytmp;
        }

        d = sqrtf(powf(x2-x1, 2.0f)+powf(y2-y2, 2.0f));
        l = (float)(x2-x1);
        a = (float)(y2-y1);

        X = (float)x1 - l/2.0f;
        Y = (float)y1 - a/2.0f;
        l *= 2.0f;
        a *= 2.0f;

        x[0] = X + a/4.0f;
        y[0] = Y - l/4.0f;
        x[1] = x[0] + l;
        y[1] = y[0] + a;

        x[2] = X - (3.0f*a)/4.0f;
        y[2] = Y + (3.0f*l)/4.0f;
        x[3] = x[2] + l;
        y[3] = y[2] + a;
        for(i=0; i < S; i++) {
                xI[0] = (((float)(S-1)-(float)i)*x[0] + (float)i*x[2])/(float)(S-1);
                yI[0] = (((float)(S-1)-(float)i)*y[0] + (float)i*y[2])/(float)(S-1);
                xI[1] = (((float)(S-1)-(float)i)*x[1] + (float)i*x[3])/(float)(S-1);
                yI[1] = (((float)(S-1)-(float)i)*y[1] + (float)i*y[3])/(float)(S-1);
                for(j=0; j < S; j++) {
                        xI[2] = (((float)(S-1)-(float)j)*xI[0] + (float)j*xI[1])/(float)(S-1);
                        yI[2] = (((float)(S-1)-(float)j)*yI[0] + (float)j*yI[1])/(float)(S-1);
                        xtmp = (int)xI[2];
                        ytmp = (int)yI[2];
                        X = xI[2]-(int)xI[2];
                        Y = yI[2]-(int)yI[2];

                        R1 = powf(3.0f+X, 3.0f) - 4.0f*powf(2.0f+X, 3.0f) + 6.0f*powf(1.0f+X,
                                                                3.0f) - 4.0f*powf(X, 3.0f);
                        R2 = powf(2.0f+X, 3.0f) - 4.0f*powf(1.0f+X, 3.0f) + 6.0f*powf(X, 3.0f);
                        R3 = powf(1.0f+X, 3.0f) - 4.0f*powf(X, 3.0f);
                        R4 = powf(X, 3.0f);

                        S1 = powf(3.0f+Y, 3.0f) - 4.0f*powf(2.0f+Y, 3.0f) + 6.0f*powf(1.0f+Y,
                                                                3.0f) - 4.0f*powf(Y, 3.0f);
```

```
                        S2 = powf(2.0f+Y, 3.0f) - 4.0f*powf(1.0f+Y, 3.0f) + 6.0f*powf(Y, 3.0f);
                        S3 = powf(1.0f+Y, 3.0f) - 4.0f*powf(Y, 3.0f);
                        S4 = powf(Y, 3.0f);

                        B1 = (R1*(float)CV_IMAGE_ELEM(img, uchar, ytmp-1, xtmp-1) +
      R2*(float)CV_IMAGE_ELEM(img, uchar, ytmp-1, xtmp) + R3*(float)CV_IMAGE_ELEM(img, uchar,
ytmp-1, xtmp+1) + R4*(float)CV_IMAGE_ELEM(img, uchar, ytmp-1, xtmp+2))/6.0f;
                        B2 = (R1*(float)CV_IMAGE_ELEM(img, uchar, ytmp, xtmp-1) +
R2*(float)CV_IMAGE_ELEM(img, uchar, ytmp, xtmp) + R3*(float)CV_IMAGE_ELEM(img, uchar, ytmp,
xtmp+1) + R4*(float)CV_IMAGE_ELEM(img, uchar, ytmp, xtmp+2))/6.0f;
                        B3 = (R1*(float)CV_IMAGE_ELEM(img, uchar, ytmp+1, xtmp-1) +
R2*(float)CV_IMAGE_ELEM(img, uchar, ytmp+1, xtmp) + R3*(float)CV_IMAGE_ELEM(img, uchar,
ytmp+1, xtmp+1) + R4*(float)CV_IMAGE_ELEM(img, uchar, ytmp+1, xtmp+2))/6.0f;
                        B4 = (R1*(float)CV_IMAGE_ELEM(img, uchar, ytmp+2, xtmp-1) +
R2*(float)CV_IMAGE_ELEM(img, uchar, ytmp+2, xtmp) + R3*(float)CV_IMAGE_ELEM(img, uchar,
ytmp+2, xtmp+1) + R4*(float)CV_IMAGE_ELEM(img, uchar, ytmp+2, xtmp+2))/6.0f;

                        B = (S1*B1+S2*B2+S3*B3+S4*B4)/6.0;
                        CV_IMAGE_ELEM(face, uchar, i, j) = (int)(B+0.5f);
                }
        }
        normalizaIluminacao();
}
```

Quadro 5 – Código fonte do método `interpolaFace`

A Figura 14 mostra o resultado da normalização após este ajuste de escala e posicionamento.

Figura 14 – Resultados da normalização da escala e da posição da face

3.3.1.3 Normalização de Iluminação

Neste último estágio de pré-processamento, o objetivo é padronizar a variação dos valores dos pixels das imagens. Para isto, a simetria da face é considerada para corrigir variações de iluminação direcionais, e então uma equalização da face inteira é realizada para igualar a faixa de valores dos pixels de todas as faces para corrigir variações de iluminação do ambiente. O trecho de código apresentado no Quadro 6 refere-se ao método responsável pela normalização da iluminação.

```
void normalizaIluminacao() {
        float a, b, c;
        int i, j;
        float mp, mediana, desvio_mediano;
        int mi, mj, mc, minit, mend;

        a = b = c = 0.0f;
        for(i=0; i < S; i++)
                for(j=0; j < S; j++) {
                        a += CV_MAT_ELEM(*C, float, 0, i*S+j)*(float)CV_IMAGE_ELEM(face, uchar, i, j);
                        b += CV_MAT_ELEM(*C, float, 1, i*S+j)*(float)CV_IMAGE_ELEM(face, uchar, i, j);
                        c += CV_MAT_ELEM(*C, float, 2, i*S+j)*(float)CV_IMAGE_ELEM(face, uchar, i, j);
                }

        for(i=0; i < S; i++)
                for(j=0; j < S; j++)
                        CV_MAT_ELEM(*ilumBuf, float, i*S+j, 0) = (float)CV_IMAGE_ELEM(face, uchar, i, j) -
(float)i * a - (float)j * b - c;

        mc = (S*S)/2;
        minit = 0;
        mend = S*S-1;

        while(1) {
                mi = minit;
                mj = mend;
                mp = CV_MAT_ELEM(*ilumBuf, float, mi, 0);

                while(mi < mj) {
                        while(CV_MAT_ELEM(*ilumBuf, float, mj, 0) >= mp && (mi < mj))
                                mj--;
                        if(mi != mj) {
                                CV_MAT_ELEM(*ilumBuf, float, mi, 0) = CV_MAT_ELEM(*ilumBuf, float, mj, 0);
                                mi++;
                        }
                        while(CV_MAT_ELEM(*ilumBuf, float, mi, 0) <= mp && (mi < mj))
                                mi++;
                        if(mi != mj) {
                                CV_MAT_ELEM(*ilumBuf, float, mj, 0) = CV_MAT_ELEM(*ilumBuf, float, mi, 0);
                                mj--;
                        }
                }
                CV_MAT_ELEM(*ilumBuf, float, mi, 0) = mp;

                if(mi == mc) {
                        mediana = CV_MAT_ELEM(*ilumBuf, float, mi, 0);
                        break;
                }
                else if(mi < mc) {
                        minit = mi+1;
                }
                else {
                        mend = mi-1;
                }
        }

        for(i=0; i < S*S; i++)
                CV_MAT_ELEM(*ilumBuf, float, i, 0) = fabsf(CV_MAT_ELEM(*ilumBuf, float, i, 0)-mediana);

        mc = (S*S)/2;
        minit = 0;
        mend = S*S-1;

        while(1) {
                mi = minit;
                mj = mend;
                mp = CV_MAT_ELEM(*ilumBuf, float, mi, 0);

                while(mi < mj) {
                        while(CV_MAT_ELEM(*ilumBuf, float, mj, 0) >= mp && (mi < mj))
                                mj--;
                        if(mi != mj) {
                                CV_MAT_ELEM(*ilumBuf, float, mi, 0) = CV_MAT_ELEM(*ilumBuf, float, mj, 0);
                                mi++;
                        }
                        while(CV_MAT_ELEM(*ilumBuf, float, mi, 0) <= mp && (mi < mj))
                                mi++;
                        if(mi != mj) {
                                CV_MAT_ELEM(*ilumBuf, float, mj, 0) = CV_MAT_ELEM(*ilumBuf, float, mi, 0);
                                mj--;
                        }
                }
                CV_MAT_ELEM(*ilumBuf, float, mi, 0) = mp;

                if(mi == mc) {
                        desvio_mediano = CV_MAT_ELEM(*ilumBuf, float, mi, 0);
                        break;
                }
```

```
                else if(mi < mc) {
                        minit = mi+1;
                }
                else {
                        mend = mi-1;
                }
        }

        for(i=0; i < S; i++)
                for(j=0; j < S; j++)
                        CV_MAT_ELEM(*ilumBuf, float, i*S+j, 0) = (float)CV_IMAGE_ELEM(face, uchar, i, j) -
(float)i * a - (float)j * b - c;

        for(i=0; i < S; i++)
                for(j=0; j < S; j++)
                        CV_MAT_ELEM(*normFace, float, i, j) = 127.5f + ((CV_MAT_ELEM(*ilumBuf, float, i*S+j,
0)-mediana)/desvio_mediano)*42.5;

        for(i=0; i < S; i++)
                for(j=0; j < S; j++)
                        CV_IMAGE_ELEM(face, uchar, i, j) = (uchar)(CV_MAT_ELEM(*normFace, float, i, j) >
255.0f ? 255.0f : (CV_MAT_ELEM(*normFace, float, i, j) < 0.0f ? 0.0f : CV_MAT_ELEM(*normFace, float, i,
j)));
}
```

Quadro 6 – Código fonte do método `normalizaImagem`

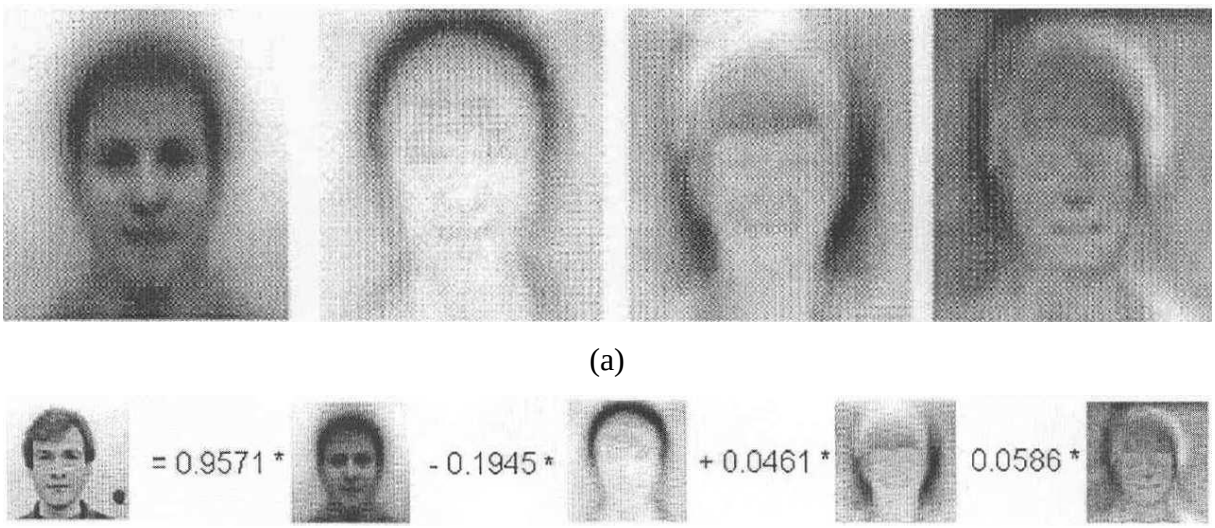
A Figura 15 mostra alguns resultados deste estágio.

Figura 15 – Resultados da normalização de iluminação

3.3.1.4 Correspondência e Avaliação

O método `Autofaces`, escolhido como mais eficiente para essa ferramenta, são autovetores utilizados para o reconhecimento facial, obtidos através da Análise de Componentes Principais (PCA, *Principal Component Analysis*) e, assim como para a detecção facial, o critério de escolha do método foi o melhor custo-benefício entre eficiência e velocidade. Autovetor, em álgebra linear, representa uma direção que é preservada por uma transformação linear. Esta técnica é responsável pela reorganização das imagens utilizadas de maneira que seja possível reduzir o espaço de representação, e assim diminuir o tempo de processamento do reconhecimento. Ela pode ser utilizada quando as imagens utilizadas dados

apresentam alguma similaridade, pois é a partir da avaliação das semelhanças presentes no conjunto de imagens que a dimensão dos dados é reduzida. Uma vez reorganizada, uma face passa a ser representada através da soma ponderada de autofaces, em vez de ser representada por uma matriz de pixels. A Figura 16 mostra um exemplo de um conjunto de autofaces (Figura 16(a)) e da representação de uma face através da combinação destas autofaces (Figura 16(b)).



(a)

(b)

Figura 15 – (a) Conjunto de autofaces e (b) representação de uma face através das autofaces em (a)

Para criar as autofaces, precisamos de um conjunto de imagens da face de um indivíduo que será reconhecido posteriormente, chamado *F*. Cada imagem $f \in F$ é representada por um ponto *N*-dimensional $[p_1,p_2,...,p_N]$, sendo que *N* representa a quantidade de pixels e p_i é o *i*-ésimo pixel da imagem. A face média $\bar{f}$ precisa ser calculada para a obtenção das autofaces, e cada pixel $\bar{p}_i \in \bar{f}$ é a média do *i*-ésimo pixel de todas as imagens em *F*.

Após a aplicação da PCA, tem-se como resultado um conjunto de *M* autofaces, chamado *A*. Cada autoface em *A* também é representada por um ponto *N*-dimensional. Cada imagem de face, que antes era representada por um vetor de pixels *N*-dimensional, agora é representado por um vetor de pesos *M*-dimensional $[w_1,w_2,...,w_M]$, sendo w_i o peso da *i*-ésima autoface $a_i \in A$. Cada peso w_i é obtido através da equação representada no Quadro 7.

$$w_i = (f - \bar{f}) \cdot a_i$$

Quadro 7 – Equação para definição de pesos

Assim como o vetor de pesos para uma face $f \in F$ pode ser obtido através do vetor de pixels, o inverso também é válido, e a face pode ser reconstruída através da equação representada no Quadro 8.

$$\hat{f} = \bar{f} + \sum_{i=1}^{M} w_i a_i$$

Quadro 8 – Equação para reconstrução da face

Para o reconhecimento, uma nova imagem *g* pode ser identificada ou não como sendo do mesmo indivíduo representado em *F* através do seguinte procedimento:

a) calcular o vetor de pesos de *f* através da equação representada no Quadro 7;
b) obter $\hat{f}$ através da equação do Quadro 8;
c) calcular o erro de reconstrução ε através da equação apresentada no Quadro 9.

Se o erro de reconstrução for menor que um valor de erro máximo *E*, então a face *f* é considerada como sendo do mesmo indivíduo representado em *F*.

$$\varepsilon = \frac{1}{N} \sum_{i=1}^{N} |p_i^g - p_i^{\hat{g}}|$$

Quadro 9 – Equação para calcular erro da reconstrução

3.3.2 Operacionalidade da implementação

Esta seção tem por finalidade mostrar a operacionalidade da implementação em nível de usuário. Nas próximas seções serão abordadas as principais funcionalidades da ferramenta.

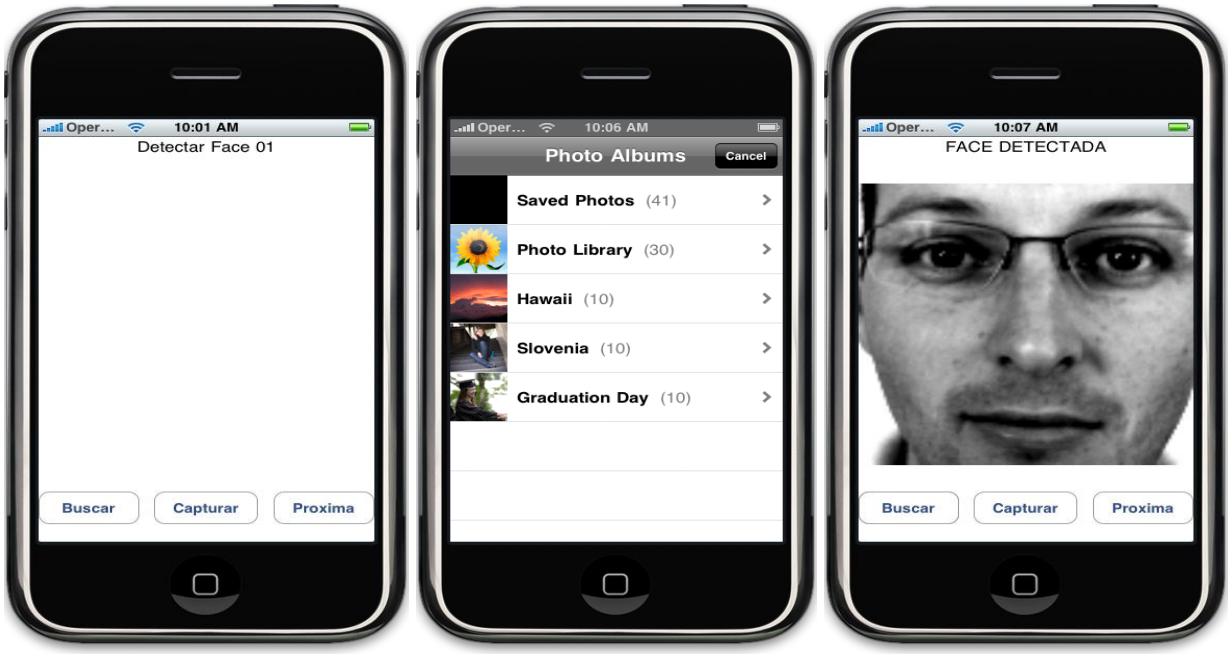
3.3.2.1 Escolhendo a funcionalidade inicial

Ao iniciar a aplicação, será apresentada a tela `Menu inicial` ao usuário disponibilizando duas opções, as quais correspondem ao cadastro ou reconhecimento de um indivíduo. A Figura 17 demonstra a tela mostrada pelo sistema nessa primeira etapa.

Figura 17 – Tela menu inicial

3.3.2.2 Cadastrando um novo usuário

A partir da seleção da opção de `Cadastrar Usuário` na tela inicial, o sistema apresentará ao usuário uma nova tela, conforme Figura 18, responsável pelo processo de cadastramento das imagens adquiridas do usuário a ser cadastrado. Nessa etapa, optou-se pela obtenção de 10 imagens aceitas pela ferramenta como adequadas para finalização do cadastro, e pode ser feita através de duas formas. A primeira seria através de uma busca de fotos já armazenadas no dispositivo utilizando o botão `Buscar` e, a segunda, através de captura a partir da câmera do aparelho, caso disponível, através do botão `Capturar`.



(a) (b) (c)

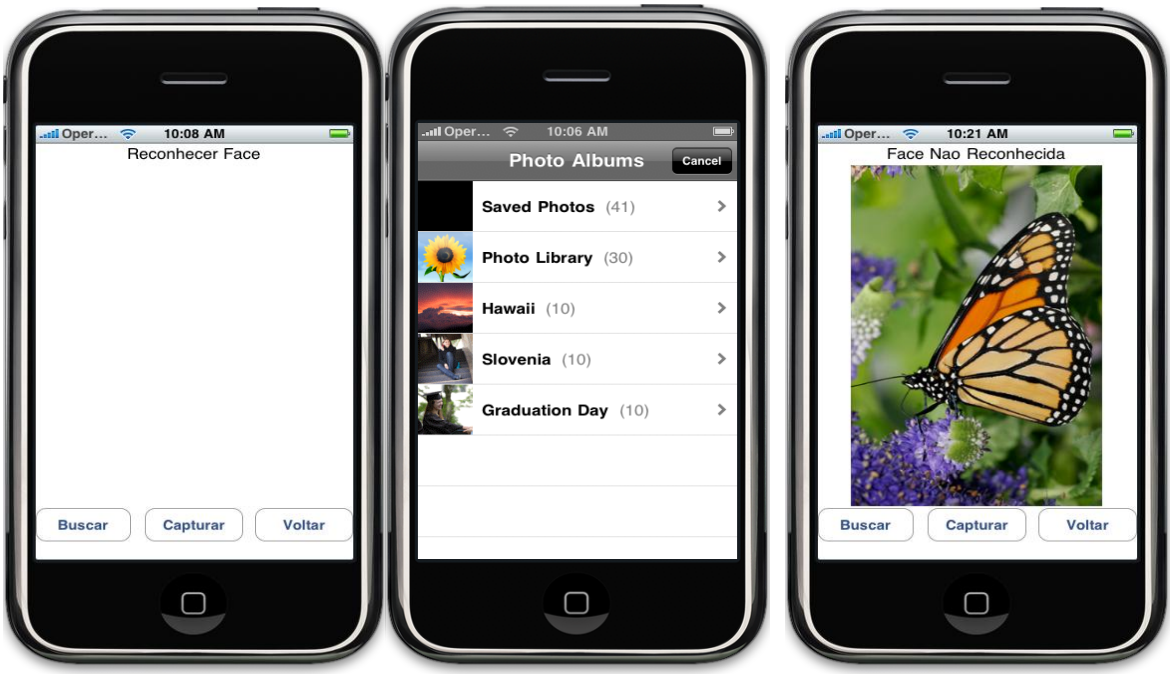
Figura 18 – (a) Tela cadastrar usuário (b) Tela Selecionar Foto (c) Foto selecionada

Após a seleção da foto e aprovação do sistema, o usuário poderá partir para a próxima aquisição selecionando o botão `Próxima`. O sistema apresenta mensagens de sucesso, `FACE DETECTADA`, ou fracasso, `FACE NÃO DETECTADA`, conforme as imagens são selecionadas, para facilitar a passagem para a próxima foto.

Por fim, obtidas as 10 faces, o botão `Treinar` será apresentado substituindo o anterior `Proxima` para encerrar o processo de cadastramento, e criar o arquivo de treino utilizado como base para o reconhecimento facial. O arquivo de treino contém o modelo da face a ser reconhecida.

3.3.2.3 Reconhecendo um usuário cadastrado

Selecionando a segundo opção apresentada no início da aplicação, através do botão `Reconhecer Usuário`, pode-se iniciar o processo de reconhecimento facial de um indivídio. A Figura 19 apresenta a tela referente a escolha dessa função.



(a) (b) (c)

Figura 19 – (a) Tela reconhecer usuário (b) Tela selecionar foto (c) Tela Foto selecionada

Os botões `Buscar` e `Capturar` tem praticamente a mesma funcionalidade da tela de cadastro. Porém, nesta etapa, servem para selecionar uma imagem do indivíduo que se quer reconhecer. Tendo feita a seleção, o sistema apresenta o resultado do processo através de mensagens de sucesso ou fracasso.

3.4 RESULTADOS E DISCUSSÃO

Nesta seção são apresentados os resultados dos experimentos realizados utilizando-se a abordagem proposta. Na subseção 3.4.1 a base de dados utilizada é mostrada. Na subseção 3.4.2 os resultados da detecção facial são apresentados. Na subseção 3.4.3 são apresentados os resultados da etapa de normalização da face. Por fim, a subseção 3.4.4 apresenta os resultados do reconhecimento facial.

3.4.1 Base de Dados

Para a realização dos experimentos, foi criada uma base de imagens, adquiridas com o iPhone, contendo 73 imagens de 8 indivíduos. Algumas imagens são mostradas na Figura 20.

Figura 20 – Base de imagens de face do iPhone

3.4.2 Detecção Facial

O estágio de detecção facial localizou corretamente a face em todas as 73 imagens da base. Alguns exemplos do resultado da detecção são apresentados na Figura 21, mostrando a robustez da abordagem a diferentes variações presentes nas imagens, entre elas escala, iluminação, pose e barba.

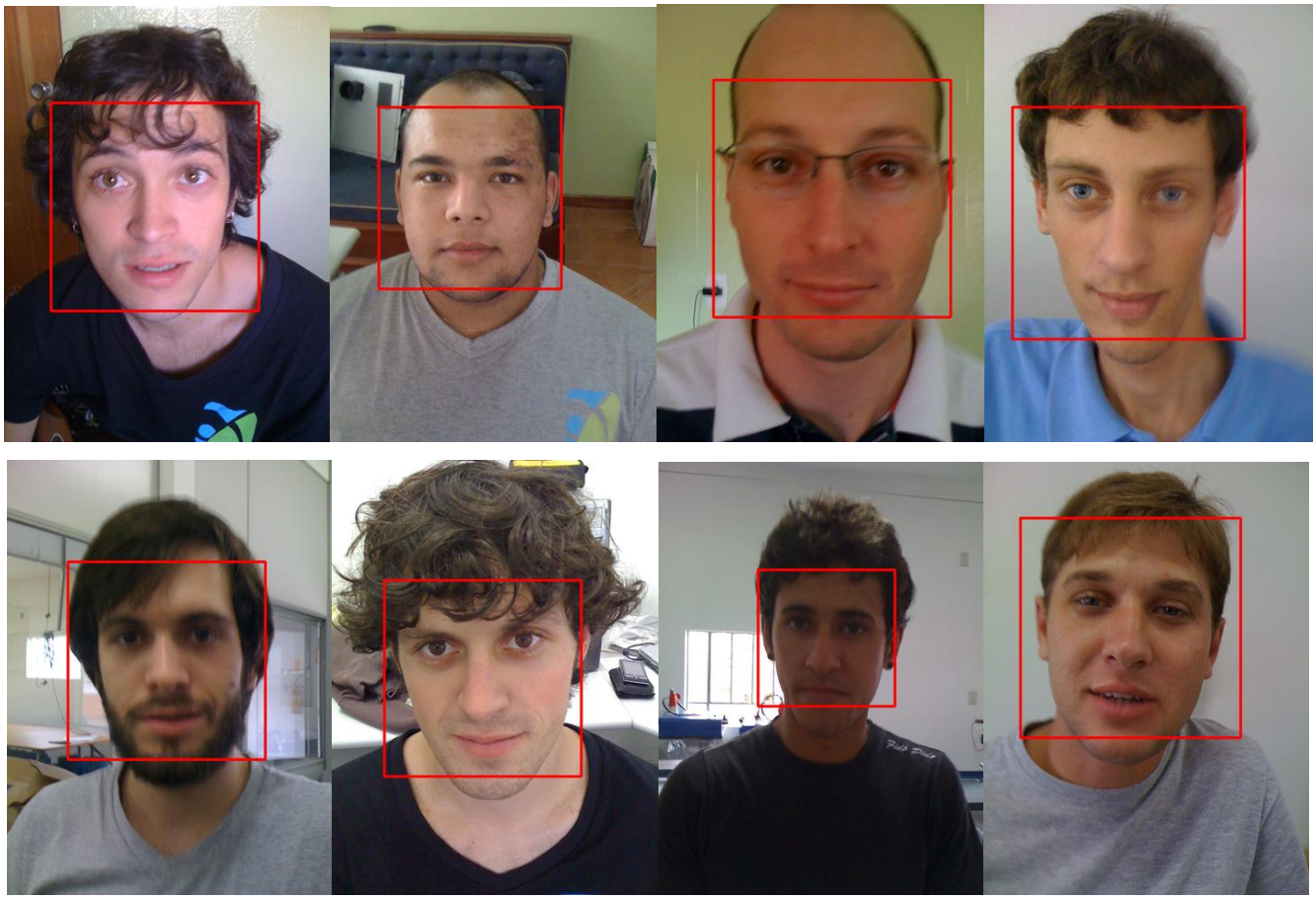


Figura 21 – Resultados da detecção facial

3.4.3 Normalização Facial

A etapa de normalização facial foi bem sucedida em 97% das imagens contidas na base de dados (71 de 73). A Figura 22 mostra alguns resultados da etapa de normalização facial. As normalizações incorretas foram causadas pela imprecisão na localização dos olhos nas imagens, como mostrado na Figura 23. Mesmo para as imagens normalizadas corretamente, uma pequena variação de escala ainda permanece, como pode ser observado na Figura 22, devido a pequenos deslocamentos na detecção dos olhos.

Figura 22 – Resultados da normalização facial

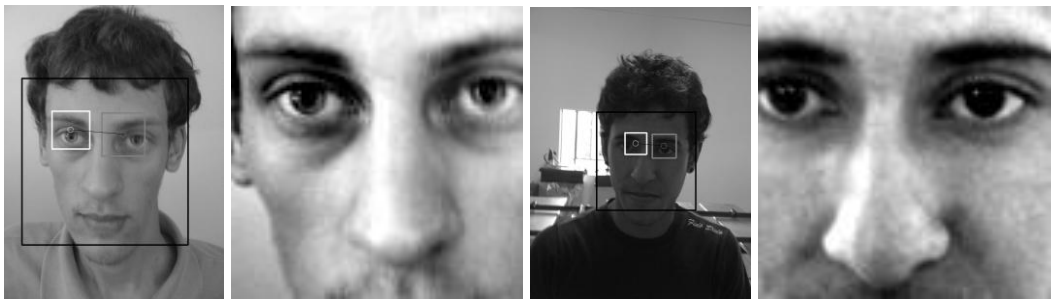


Figura 23 – Exemplos de normalização facial incorreta

3.4.4 Reconhecimento Facial

Para o reconhecimento facial, foi realizado um teste para cada indivíduo da base, considerando em cada teste que o indivíduo em questão é o dono do dispositivo móvel. Nesse experimento, sete imagens do indivíduo são utilizadas para gerar as autofaces, e então as demais faces do indivíduo são utilizadas como tentativas de autenticação, e as imagens dos demais indivíduos são utilizadas para tentar burlar o sistema. A Tabela 1 mostra o erro máximo de reconstrução obtido para imagens do indivíduo no treinamento, e também o erro

mínimo de reconstrução obtido para imagens dos demais indivíduos não representados no conjunto de imagens de treino. E1 representa o erro máximo de reconstrução de uma imagem do mesmo indivíduo utilizado no treino e E2 representa o erro mínimo de reconstrução de uma imagem que não pertence ao individuo utilizado no treino.

Tabela 1 – Erros de reconstrução obtidos para imagens do indivíduo utilizado no treino e demais indivíduos.

Indivíduo	**E1**	**E2**
01	35.510231	40.199074
02	31.284386	34.785160
03	15.387489	37.137314
04	32.508778	34.202297
05	22.002182	32.547684
06	23.343987	34.602760
07	19.305302	31.361706
08	13.701664	32.415688

Assumindo que valor de erro máximo *E* é estabelecido para cada treinamento, como o valor de E2 é maior que o valor de E1 para todos os indivíduos, o sistema proposto é capaz de reconhecer 100% das imagens do mesmo indivíduo utilizado no treino e também capaz de identificar 100% dos casos de tentativas de fraude. Para isto, basta determinar que o valor de *E* esteja entre os valores de E1 e E2 para cada indivíduo.

Se o objetivo for uma solução mais genérica, e apenas um valor para *E* é estabelecido, então duas soluções são possíveis:

a) *E* recebe o valor máximo entre os valores calculados para E1;

b) *E* recebe o valor mínimo entre os valores calculados para E2.

No primeiro caso, todas as imagens do mesmo indivíduo utilizado no treino continuam sendo reconhecidas, porém algumas tentativas de fraude serão bem sucedidas. Nos experimentos realizados, 5% das tentativas de fraude (27 de 511) foram bem sucedidas. No segundo caso, nenhuma tentativa de burlar o sistema foi bem sucedida, mas algumas imagens que deveriam ser reconhecidas podem ser classificadas como tentativa de fraude. Nos experimentos realizados, apenas 82% das imagens foram corretamente autenticadas (14 de 17).

Por fim, para um sistema mais confiável, é necessário que um estágio de treino seja realizado para determinar um valor específico de *E* para cada indivíduo, e assim tornar possível a garantia de melhores resultados para o reconhecimento e detecção de fraude. Em uma solução mais genérica, que pode ser utilizada para coibir as tentativas de fraude, um valor de *E* é determinado para todos os indivíduos, e ainda é possível garantir boas taxas de verificação com um mínimo de autenticações incorretas.

4 CONCLUSÕES

Com a evolução das técnicas de visão computacional no decorrer dos anos, tornou-se possível sua utilização em sistemas de reconhecimento facial em diversas áreas. A presente proposta propõe utilizar tais técnicas aplicadas em dispositivos de menor poder computacional.

Para tanto, foram estudados algoritmos e métodos, buscando a implementação de um sistema de reconhecimento facial para dispositivos móveis, utilizando especificamente o iPhone. Desta forma, baseando-se nos resultados do protótipo, foi possível avaliar a viabilidade de aplicação de tais métodos para esse aparelho e criar possibilidades de diferentes aplicações na área.

Se comparado aos trabalhos correlatos, o presente projeto difere-se por visar o uso da capacidade do dispositivo móvel para o total processamento, desde a detecção até o reconhecimento. Assim como sugere o iFace, propõe-se ser utilizado como ferramenta de segurança, porém, poderá ser utilizado até mesmo sem uma conexão externa, já que não dependerá do processamento em um servidor para obtenção de um resultado final.

O terceiro trabalho correlato citado serve como base para futuras extensões, pois trata da melhora no desempenho do método utilizado para o reconhecimento facial neste trabalho.

4.1 EXTENSÕES

Assim como a comparação em uma seqüência de vídeo, outras possibilidades podem ser analisadas e discutidas no aspecto prático do sistema, como:

a) as chances de burlar o aplicativo utilizando uma foto do usuário cadastrado;
b) quantidade de fotos cadastradas na base e/ou captadas para que a taxa de reconhecimento seja alta, evitando falsos negativos e positivos;
c) fatores como mudança de escala e orientação, para restringir o mínimo e máximo da face durante o reconhecimento;
d) testes com pequenas alterações na face (bronzeamento, iluminação, ambientes internos ou externos), acessórios (óculos, brincos, batom), expressões e desvio de orientação;

e) utilização de uma seqüência de imagens com o movimento de rotação da cabeça, substituindo o uso de apenas a face frontal.

REFERÊNCIAS BIBLIOGRÁFICAS

AGÊNCIA NACIONAL DE TELECOMUNICAÇÕES. **Portal Anatel**: números do setor. [S.l.], 2009. Disponível em: http://www.anatel.gov.br. Acesso em: 2 set. 2009.

ANIMETRICS. **iFace:** facial identification application. [Conway], [2009?]. Disponível em: <http://www.animetrics.com/products/iface.php>. Acesso em: 7 set. 2009.

APPLE. **iPhone:** celular, iPod e dispositivo para internet. [S.l.], 2009a. Disponível em: <http://www.apple.com/br/iphone/>. Acesso em: 3 set. 2009.

_____. **iPhone developer program:** apple developer connection. [S.l.], 2009b. Disponível em: <http://developer.apple.com/iphone/program/>. Acesso em: 3 set. 2009.

BEYOND IF SOLUTIONS. **HandKey II**. Palos Park, [2009?]. Disponível em: <http://www.beyondifsolutions.com/>. Acesso em: 20 mai. 2010.

BOLLE, Raul M. et al. **A. Guide to biometrics**. [S.l], Springer-Verlag, 2003.

CAMPOS, Teófilo E. de. **Técnicas de seleção de características com aplicações em reconhecimento de faces.** 2001. 160 f. Dissertação (Mestrado em Ciência da Computação) – Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo.

CHELLAPA, Rama; WILSON, Charles L.; SIROHEY, Saad. Human and machine recognition of faces: a survey. **Proceedings of the IEEE**, [s.l.], v. 83, n. 5, p. 703-740. 1995.

DAUGMAN, John. High confidence visual recognition of persons by a test of statistical independece. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, [s.l.], v. 15, n. 11, p. 1148–1161, 1993.

_____. How iris recognition works. **IEEE Transactions on Circuits and Systems for Video Technology**, [s.l.], v. 14, n. 1, p. 21–30, 2004.

DIÁRIO DE SÃO PAULO. **Celular lidera estatísticas de roubo em SP durante 2007**. São Paulo, 2008. Disponível em: <http://oglobo.globo.com/sp/mat/2008/02/01/celular_lidera_estatisticas_de_roubo_em_sp_durante_2007-379761881.asp>. Acesso em: 2 set. 2009.

FIOR, Alessandro. G. F. et al. Detecção e reconhecimento facial em seqüências de vídeo. **Revista Eletrônica de Iniciação Científica**, [s.l.], v. 8, n. 2, p. 33-43, Jun. 2008.

GONZALEZ, Rafael C.; WOODS, Richard E. **Digital image processing**. 2nd ed. New Jersey: Prentice-Hall, 2002.

HESHER, Curt; SRIVASTAVA, Anuj; ERLEBACHER, Gordon. Principal component analysis of range images for facial recognition. In: INTERNATIONAL CONFERENCE ON IMAGING SCIENCE, SYSTEMS, AND TECHNOLOGY, [1]., 2002, Las Vegas. **Proceedings…** Las Vegas: CSREA Press, 2002, não paginado.

HJELMAS, Erik; LOW, Boon K. Face detection: a survey. **Computer Vision and Image Understanding**, [s.l.], v. 83, n. 3, p. 236-274, Set. 2001.

HSU, Rein L; ABDEL-MOTTALEB, Mohamed; JAIN, Anil K. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, [s.l.], v. 24, n. 5, p. 696–706, 2002.

ISIS RESEARCH GROUP. **Automatic gait recognition for human ID at a distance**. Southampton, [2003?]. Disponível em: <http://www.gait.ecs.soton.ac.uk/>. Acesso em: 20 de maio 2010.

JAIN, Anil. K. et al. Biometrics: a grand challenge. In: INTERNATIONAL CONFERENCE ON PATTERN RECOGNITION, 4., 2004, Cambridge. **Proceedings…** Cambridge: [S.n.], 2004, não paginado.

JAIN, Anil. K.; ROSS, Arun A.; PRABHAKAR, Sunil. An introduction to biometric recognition. **IEEE Transactions on Circuits and Systems for Video Technology**, [s.l], v. 14, n. 1, p.4–20, 2004.

KIRBY, Michael; SIROVICH, Lawrence. Application of the karhunen-loeve procedure for the characterization of human faces. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, [s.l.], v. 12, n. 1, p. 103-108, Jan. 1990.

LEMES, Rubisley P. et al. Registro de imagens no reconhecimento de impressões digitais. In: 20TH BRAZILIAN SYMPOSIUM ON COMPUTER GRAPHICS AND IMAGE PROCESSING, 20., 2007, Belo Horizonte. **Proceedings…** Belo Horizonte: [S.n.], 2007, não paginado.

MOON, Hyeonjoon; PHILLIPS, Patrick. J. Computational and performance aspects of PCA-based face-recognition algorithms. **Perception**, [s.l.], v. 30, n. 1, p. 303-321. [Jan?]. 2001.

MOREIRA, Daniela. **Roubo de identidades ameaça celulares**. São Paulo, 2009. Disponível em: <http://info.abril.uol.com.br/aberto/infonews/032009/20032009-33.shl>. Acesso em: 2 set. 2009.

NATIONAL SCIENCE & TECHNOLOGY COUNCIL SUBCOMMITTEE ON BIOMETRICS. **Biometrics catalog**. Disponível em: <http://www.biometricscatalog.org/>. Acesso em: 20 maio 2010.

NIWA, Yoshimasa. **Using OpenCV on iPhone**. [S.l.], 2009. Disponível em: <http:// niw.at/articles/2009/03/14/using-opencv-on-iphone/em>. Acesso em: 7 set. 2009.

OPENCV. **Open computer vision library**. Disponível em: <http://sourceforge.net/projects/opencvlibrary/>. Acesso em: 20 maio 2010.

POLAR BEAR FARM. **Face match:** face recognition in your pocket. [Christchurch], [2009?]. Disponível em: <http://www.polarbearfarm.com/facematch/index.html>. Acesso em: 7 set. 2009.

SHAPIRO, Linda. G.; STOCKMAN, George. C. **Computer vision**. Washington: Prentice-Hall, 2001.

SIROVICH, Lawrence; KIRBY, Michael. Low-dimensional procedure for the characterization of human faces. **Journal of the Optical Society of America A: Optics, Image Science, and Vision**, Washington, v. 4, n. 3, p. 519–524, mar. 1987.

TURK, Matthew; PENTLAND, Alex. Eigenfaces for recognition. **Journal of Cognitive Neuroscience**, Cambridge, v. 3, n. 1, p. 71-86. jan. 1991.

VIOLA, Paul; JONES, Michael. J. Robust real-time face detection. **International Journal of Computer Vision**, Springer, v. 57, n. 2, p. 137-154, Maio 2004.

YANG, Ming-Hsuan; KRIEGMAN, David J.; AHUJA, Narendra. Detecting faces in images: a survey. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, [s.l.], v. 24, n. 1, p. 34-58, jan. 2002.