UNIVERSIDADE ESTADUAL PAULISTA
"Júlio de Mesquita Filho"
Pós-Graduação em Ciência da Computação

André Marcelo Farina

BioMobile: Sistema de Identificação de Usuários em Dispositivos Móveis na Plataforma Android Utilizando Reconhecimento de Faces a Partir de Vídeo

UNESP

2012

André Marcelo Farina

BioMobile: Sistema de Identificação de Usuários em Dispositivos Móveis na Plataforma Android Utilizando Reconhecimento de Faces a Partir de Vídeo

Orientador: Prof. Dr. Aparecido Nilceu Marana

Dissertação de Mestrado elaborada junto ao Programa de Pós-Graduação em Ciência da Computação – Área de Concentração em Sistemas de Computação, linha de Processamento de Imagens e Visão Computacional, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

UNESP

2012

ANDRÉ MARCELO FARINA

BioMobile: Sistema de Identificação de Usuários em Dispositivos Móveis na Plataforma Android Utilizando Reconhecimento de Faces a Partir de Vídeo

Dissertação de Mestrado elaborada junto ao Programa de Pós-Graduação em Ciência da Computação – Área de Concentração em Sistemas de Computação, linha de Processamento de Imagens e Visão Computacional, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

BANCA EXAMINADORA

Prof. Dr. Aparecido Nilceu Marana
Professor Adjunto
UNESP/FC – Bauru
Orientador

Prof. Dr. João Eduardo Machado Perea Martins
Professor Doutor
UNESP/FC – Bauru

Profa. Dra. Fátima de Lourdes dos Santos Nunes Marques
Professora Associada
USP/EACH – São Paulo

São José do Rio Preto, 3 de Fevereiro de 2012

Dedico essa dissertação de mestrado
Aos meus pais Edno e Geny, dos quais recebi as mais importantes lições,
A minha amada esposa Giovana, de quem obtive todo apoio e incentivo e
Aos meus filhos Gustavo e Rafael, profundamente amados.

AGRADECIMENTOS

Venho agradecer a todos que de alguma maneira, direta ou indiretamente, contribuíram para a realização deste trabalho:

Agradeço primeiramente a Deus, por ter concedido condições para a realização deste projeto.

À minha família, principalmente à minha esposa, pelo incentivo e apoio que foram fundamentais durante todo o período de estudo.

Ao meu orientador, Prof. Dr. Aparecido Nilceu Marana, pela paciência, incentivo, apoio e grande dedicação sem os quais este trabalho não poderia ser concebido.

Agradeço ao Prof. Dr. João Perea Martins, a quem várias vezes recorri para obter uma palavra de ânimo e incentivo.

À Lecom Tecnologia S/A, empresa onde trabalho, especialmente ao Tiago Amôr, Diretor de Operações, pela compreensão e valorização deste trabalho fazendo concessões e apoiando sempre que solicitado.

E por fim, à Unesp, pela infraestrutura oferecida e pela qualidade do programa.

RESUMO

Na era da informação em que vivemos a demanda por conectividade e acesso a dados é crescente. Neste cenário os dispositivos móveis estão se tornando cada vez mais populares. Com o aumento da demanda por este tipo de dispositivo aumenta também o volume de informações neles armazenadas. Com isso, os dispositivos móveis estão se tornando uma importante fonte de informação que precisam de mecanismos adequados de proteção. O objetivo desta dissertação de mestrado foi promover um estudo sobre a viabilidade da utilização de Biometria para a proteção de informações dos usuários em dispositivos móveis, em particular nos *smartphones*, via reconhecimento de faces. A opção pelo reconhecimento facial deveu-se ao fato dos *smartphones* proverem naturalmente recursos de *hardware* para a aquisição de vídeos. Para a realização do estudo proposto, foi desenvolvido um sistema de reconhecimento facial na plataforma Android, denominado BioMobile, cuja arquitetura foi projetada para permitir a execução integralmente no domínio dos dispositivos móveis. Para a detecção das faces, foi adotado o algoritmo Viola-Jones, enquanto que para o reconhecimento foram usados os descritores PCA e LBP. O BioMobile foi desenvolvido para operar nos modos de autenticação e de reconhecimento. Além disso, ele trabalha com vídeos, o que possibilita a adoção da técnica de maioria de votos, tornando-o mais tolerante a falhas e mais preciso. Os resultados experimentais mostraram que, ao contrário do algoritmo baseado em LBP, o algoritmo baseado em PCA se torna inviável quando usado em dispositivos com restrições de memória e de processamento. Os resultados obtidos também indicam que o modo de operação de reconhecimento quando aplicado totalmente no domínio dos dispositivos móveis não é conveniente em termos de tempo de processamento. Por outro lado, no modo de autenticação os resultados apontam para um desempenho bem melhor neste quesito. O sistema também apresentou baixas taxas de erro (em torno de 2%) dependendo da configuração adotada.

Palavras-chave: Biometria, reconhecimento facial, dispositivos móveis, Android.

ABSTRACT

In the information age in which we live the demand for connectivity and data access is growing. In this scenario the mobile devices are becoming increasingly popular, especially the smart phones. The increasing demand for this type of device leads to high personal and professional information storage. Therefore, mobile devices are becoming important sources of information that need proper protection mechanisms. The purpose of this master dissertation was to carry out a study on the feasibility of using biometrics as a means of safe, efficient and appropriate information protection on mobile devices, especially in smart phones, via user identification through face recognition. The choice for facial recognition was due to the fact that the smart phones provide embedded video cameras. In order to conduct the proposed study, it was developed a face recognition system on the Android platform, called BioMobile, whose architecture was designed to allow its full execution without the need of external servers. For face detection it was adopted the Viola-Jones algorithm, while for face recognition it were assessed the PCA and LBP descriptors. BioMobile was also designed to operate in authentication and recognition modes. Besides, it works with videos instead of still images that allow the adoption of the technique of majority voting becoming it more fault-tolerant and precise. The experimental results showed that the PCA algorithm is not feasible when used in devices with memory and processing power limitations. LBP, in contrast, showed to be appropriated. The results also indicated that the recognition operation mode fully executed in the mobile devices is not so convenient due to the processing time required. On the other hand, the authentication operation mode showed good performance regarding this issue. The system also showed few error rates (around 2%) depending on the adopted configuration.

Key-words: Biometry, facial recognition, mobile devices, Android.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

CAPÍTULO 1. **Introdução**

Os dispositivos móveis estão se tornando cada vez mais populares, em parte devido à necessidade de conectividade, característica da sociedade contemporânea, e em parte devido à nova gama de funcionalidades e serviços oferecidos por estes dispositivos. No mesmo ritmo em que ganham funcionalidades, os dispositivos móveis armazenam cada vez mais informações, tanto de cunho pessoal como profissional, se tornando importantes fontes de informação que precisam de mecanismos eficientes e adequados de proteção (IJIRI; SAKURAGI; LAO, 2006).

Porém, o tamanho cada vez mais reduzido destes dispositivos móveis, as limitações das memórias e a menor capacidade computacional destes dispositivos, se comparados aos *desktops* e *notebooks*, representam um fator limitador na implementação de mecanismos de segurança. Nesses dispositivos de tamanhos reduzidos, a utilização de senhas geraria grande desconforto ao usuário ao impor a necessidade de digitação de vários caracteres alfanuméricos, incluindo muitas vezes letras maiúsculas e minúsculas, além de caracteres especiais.

Assim, com vistas a prover um modo eficiente e conveniente de autenticação para os dispositivos móveis, que têm nos *smartphones* seu melhor representante, a autenticação biométrica tem sido cada vez mais cogitada, principalmente quando é baseada em algoritmos leves, que requerem poucos recursos de processamento (HADID ET AL, 2007).

O objetivo desta dissertação de mestrado foi promover um estudo sobre a viabilidade da utilização de Biometria como meio seguro, eficiente e adequado para proteção de informações nos dispositivos móveis, em particular nos *smartphones*, via identificação de usuários por meio do reconhecimento de faces. A opção pelo reconhecimento facial deveu-se ao fato dos *smartphones* proverem naturalmente recursos de hardware para a aquisição de vídeos. Para a realização do estudo proposto, foi desenvolvido um sistema de reconhecimento facial na plataforma Android denominado BioMobile. A arquitetura deste sistema foi projetada para permitir sua execução integral no domínio dos dispositivos móveis, sem a necessidade de conexão com servidores ou outros sistemas durante o processo de identificação dos usuários.

O BioMobile foi implementado na plataforma Android, mantida por uma aliança de empresas liderada pela Google, e apresenta duas versões: uma que utiliza descritores baseados

na técnica de análise das componentes principais, PCA (*Principal Component Analysis*), e outra baseada em padrões binários locais, LBP (*Local Binary Patterns*).

1.1 Objetivos

O objetivo desta dissertação de mestrado foi promover um estudo sobre a viabilidade da utilização de Biometria como meio seguro, eficiente e adequado para proteção de informações nos dispositivos móveis, em particular nos *smartphones*, via identificação de usuários por meio do reconhecimento de faces. Para tanto, foi projetado e implementado um sistema de autenticação biométrica usando o reconhecimento de faces para dispositivos móveis, com ênfase para a plataforma Android da Google, denominado BioMobile. A implementação desse sistema permitiu comparar o desempenho, nos quesitos tempo de processamento e acurácia, das técnicas baseadas em PCA e LBP, com diferentes configurações, para a identificação dos usuários por meio de suas características faciais, nos modos de autenticação e reconhecimento.

1.2 Organização da Dissertação

Além deste capítulo introdutório, esta dissertação de mestrado possui os seguintes capítulos:

- Capítulo 2: apresenta uma revisão bibliográfica sobre Biometria e suas principais características, incluindo os sistemas biométricos e as formas de analisar seus desempenho, e descreve técnicas para a detecção de faces em imagens por meio do algoritmo Viola-Jones e para reconhecimento facial por meio das técnicas PCA e LBP;
- Capítulo 3: aborda as principais plataformas dos dispositivos móveis, bem como os ambientes de desenvolvimento, com ênfase para a plataforma Android adotada neste trabalho;
- Capítulo 4: apresenta um estudo sobre o reconhecimento facial em dispositivos móveis e alguns trabalhos correlatos;
- Capítulo 5: descreve o BioMobile, sistema biométrico projetado e implementado neste trabalho;

- Capítulo 6: apresenta os resultados obtidos com a aplicação do sistema biométrico desenvolvido utilizando uma base de dados de vídeos contendo faces capturadas via câmeras de vídeos embutidas em telefones celulares;
- Capítulo 7: apresenta as conclusões do trabalho e indica trabalhos futuros.

CAPÍTULO 2. **Biometria**

Este capítulo é dedicado à revisão bibliográfica sobre Biometria incluindo seus principais aspectos, as características dos sistemas biométricos, o reconhecimento facial abordando os processos de detecção e de reconhecimento facial considerando os algoritmos Viola-Jones, PCA e LBP, além de apresentar a API OpenCV, composta por vários algoritmos que apoiam o processamento de imagens em geral e o processamento biométrico em particular.

2.1 Introdução

Biometria pode ser definida como o modo de identificar pessoas baseadas em suas características físicas ou comportamentais. Tais características devem ser únicas de modo que possam ser usadas para determinar a identidade de uma pessoa.

O processo de identificação ocorre via uma credencial que pode ser definida como uma evidência fornecida por alguém, ao requisitar algum tipo de acesso. As credenciais são classificadas em três tipos (MILLER, 1994):

- Posse: o requisito para acesso é deter a posse de um objeto. Por exemplo, a posse da chave de um veículo identifica o usuário autorizado;
- Conhecimento: o requisito para acesso é deter certo conhecimento. Trata-se do método mais comum de controle de acesso atualmente, no qual a autenticação do usuário é baseada em uma informação secreta, uma senha, compartilhada entre o usuário e a aplicação;
- Biometria: o requisito para acesso é apresentar características físicas ou comportamentais computadas na forma de um identificador biométrico único.

Dentre os três tipos de credenciais, é nítido que as credenciais biométricas são as mais robustas com relação à prevenção de fraudes, pois são mais difíceis de serem compartilhadas, roubadas, forjadas ou alteradas.

O termo “reconhecimento biométrico” é bastante usado em sistemas de informação e trata-se do reconhecimento automático de uma pessoa com base em uma ou mais de suas características físicas ou comportamentais. A palavra "biometria" também é adotada para

designar métodos de reconhecimento biométrico.

Como vantagens, a biometria apresenta as seguintes características (JAIN, 2009):

- não pode ser facilmente esquecida, transferida, perdida, copiada ou utilizada por outra pessoa visto que é algo pessoal e intransferível;
- desencoraja a prática de fraudes e aumenta a segurança uma vez que aumenta significativamente a dificuldade de fraudar uma identidade;
- propicia grande conveniência, pois não exige a memorização de senhas e, tampouco, o porte de algum dispositivo para autenticação;
- elimina as falsas repudiações, pois não há argumentos para o repúdio quando o processo de autenticação está baseado em características biométricas.

A identificação biométrica não apresenta apenas vantagens. Em geral, são apontadas as seguintes desvantagens (JAIN, 2009):

- as características biométricas de uma pessoa não pode ser reestabelecida ou substituída como as senhas. Uma vez copiada, uma identidade biométrica será sempre passível de ser usada em fraudes;
- os sistemas biométricos podem ser atacados e dados biométricos podem ser roubados;
- as informações biométricas podem revelar muito mais do que a identidade da pessoa. Algumas doenças, hábitos ou outras particularidades do indivíduo podem ser reveladas pelas características biométricas;
- a identidade da pessoa depende de uma pontuação, o que pode levar a erros de decisão.

2.2 Sistemas Biométricos

2.2.1 Conceitos fundamentais

A função básica de um sistema biométrico é o reconhecimento de padrões. Em biometria padrões estão associados às características físicas e comportamentais humanas.

Para realizar esta função, o sistema biométrico passa por três etapas básicas: i) a aquisição de dados biométricos de uma amostra, ii) a extração do conjunto de característica dos dados obtidos, e iii) a comparação das características com outras previamente cadastradas (JAIN; ROSS; PRABHAKAR, 2004).

Quanto ao modo de operação, os sistemas biométricos podem atuar tanto na autenticação (verificação) como na identificação (reconhecimento) de pessoas (BOLLE *et al*. 2004). Enquanto o primeiro modo consiste no processo de verificar a veracidade de uma declaração de identidade, o segundo busca relacionar um indivíduo amostrado com a identidade de uma pessoa cadastrada previamente em um banco de dados.

Na Figura 2.1 estão ilustrados os dois modos de operação dos sistemas biométricos.

2.2.2 Desempenho dos Sistemas Biométricos

Apesar dos muitos esforços da comunidade científica e da grande variedade de propostas que têm surgido com o objetivo de lidar com as limitações dos sistemas biométricos, eles ainda continuam sujeitos a erros, muitos deles devidos às más condições nas quais a amostra é capturada. A qualidade da característica biométrica é um fator importante no desempenho do sistema.

Outra característica que pode comprometer o desempenho dos sistemas biométricos são as altas variabilidades intraclasse e as altas similaridades interclasses que podem estar presentes nas informações biométricas.

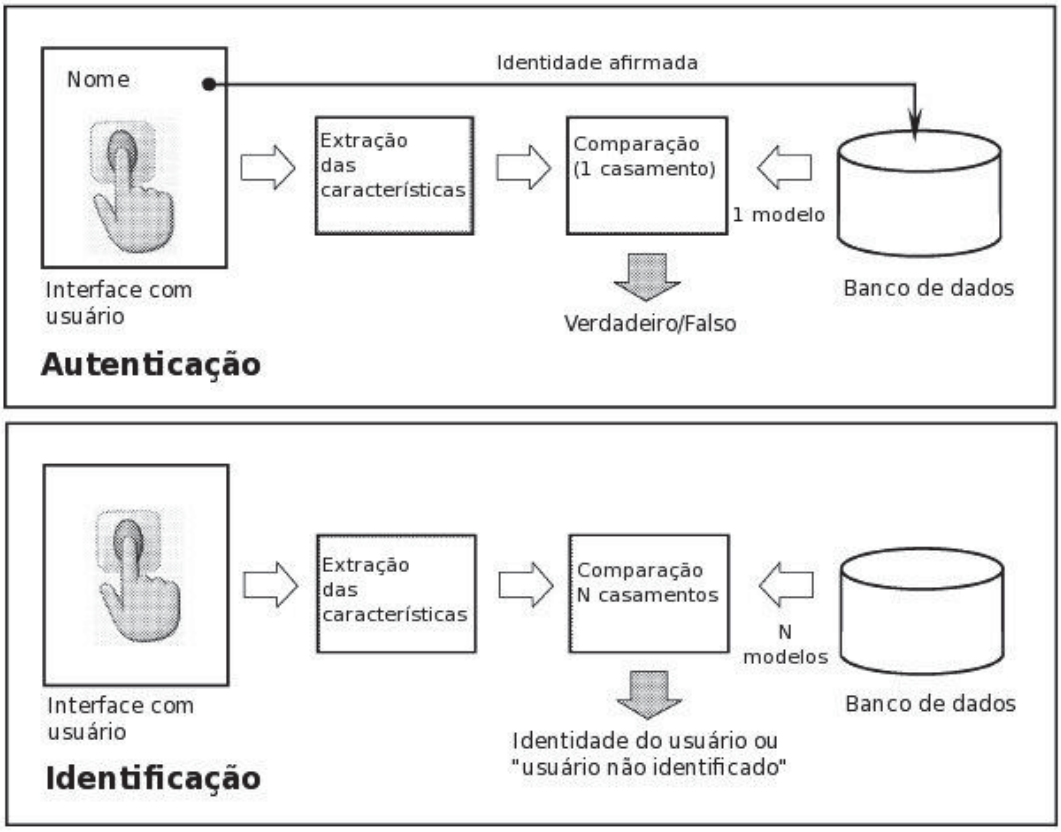

Figura 2.1. Diagrama de blocos dos dois principais modos de operação de um sistema biométrico: autenticação (verificação) e identificação (reconhecimento). (JAIN; ROSS; PRABHAKAR, 2004).

A variabilidade intraclasse ocorre quando amostras biométricas de uma mesma pessoa são capturadas em diferentes condições ambientais ou emocionais, ou mesmo utilizando-se diferentes sensores. Estas variações podem levar a diferentes dados biométricos associados a uma mesma pessoa.

A similaridade interclasses, por outro lado, ocorre quando amostras biométricas de diferentes pessoas possuem uma variação muito pequena, como no caso das características faciais de gêmeos univitelinos.

Deste modo, os sistemas biométricos podem ter seus desempenhos limitados pelas condições da captura da amostra ou pelas suas características intrínsecas.

Outro fator que pode limitar o desempenho dos sistemas biométricos é o fato de que o resultado pode assumir apenas dois estados: verdadeiro ou falso. Este fator aliado ao uso de um sistema de pontuação na tomada de decisão pode levar a dois tipos de erros: falsa aceitação (FA) que é o reconhecimento de uma amostra falsa como sendo verdadeira e falsa rejeição (FR) que é o reconhecimento de uma amostra verdadeira como sendo falsa.

A contribuição destes erros na limitação da eficácia dos sistemas biométricos é tão

grande que suas taxas de ocorrências são usadas na avaliação de desempenho de sistemas biométricos. Essas taxas podem ser calculadas da seguinte forma:

- taxa de falsa aceitação (*False Acceptance Rate* - FAR) ou taxa de falsos positivos:
  FAR = quantidade de falsa aceitação / quantidade de tentativas de impostores
- taxa de falsa rejeição (*False Rejection Rate* – FRR) ou taxa de falsos negativos:
  FRR = quantidade de falsa rejeição / quantidade de tentativas de genuínos

O mecanismo de pontuação dos sistemas biométricos depende de um limiar (*threshold*) que é usado como uma linha divisória entre os estados verdadeiro e falso. Isto se faz necessário porque os dados biométricos de uma amostra nunca são idênticos aos dados de outra, mesmo que elas sejam do mesmo usuário e capturadas sob as mesmas condições e pelo mesmo sensor (JAIN; ROSS; PRABHAKAR, 2004).

Na Figura 2.2 são apresentados exemplos de distribuições de pontuações obtidas com comparações impostoras e genuínas, além das taxas FRR e FAR determinadas por um determinado valor de limiar.

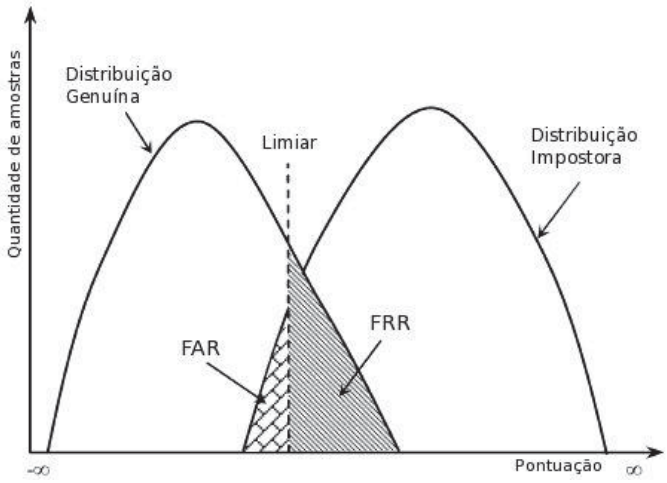

Figura 2.2. Um limiar estabelecendo regiões de falsa aceitação e falsa rejeição em um exemplo de distribuições de pontuações genuínas e impostoras (JAIN; ROSS; PRABHAKAR, 2004).

Observando a Figura 2.2 nota-se que quanto mais à esquerda estiver posicionado o limiar, mais seguro será o sistema, pois a taxa de falsa aceitação diminuirá, enquanto que ao ser deslocado para a direita mais conveniente será o sistema, pois reduzirá a taxa de falsa rejeição. A Figura 2.3 também mostra este comportamento em um gráfico onde as curvas FAR e FRR estão traçadas em função do valor do limiar.

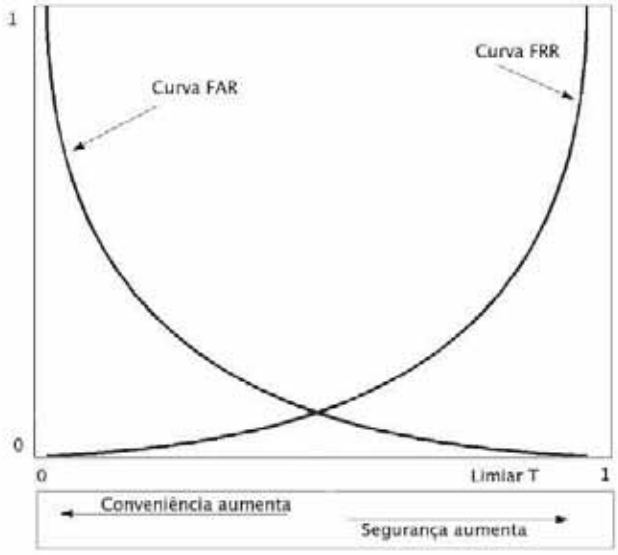

Figura 2.3. Curvas FAR e FRR em função do Limiar T (COSTA; OBELHEIRO; FRAGA, 2006).

Segundo JAIN, ROSS & PRABHAKAR (2004) os erros de falha de captura (*Failure To Capture* – FTC) e falha de cadastro (*Failure To Enroll* - FTE) também interferem no desempenho dos sistemas biométricos.

A falha de captura está associada à quantidade de vezes que o dispositivo falha na tarefa de capturar a amostra biométrica, enquanto que a falha de cadastro representa a porcentagem de vezes que o usuário tenta sem sucesso se cadastrar no sistema.

2.3 Reconhecimento Facial

O reconhecimento facial é uma das formas de identificação biométrica que apresentam maior aceitabilidade, uma vez que os seres humanos utilizam naturalmente as características faciais para identificarem-se mutuamente. Esta propriedade particular, aliada ao fato de possibilitar o reconhecimento à distância e de forma sigilosa, torna o reconhecimento facial uma das principais técnicas de identificação biométrica de pessoas.

Antes da etapa de extração das características da face é necessária uma etapa para a sua detecção em uma imagem particular.

A etapa de detecção, ou segmentação como também é conhecida, é um processo crítico para o sucesso do reconhecimento facial. Metodologias baseadas em funções de distâncias e em redes neurais podem alcançar taxas de detecção correta de aproximadamente 85% (ZHAO *et al.*, 2003).

Na etapa de extração das características das imagens da face são adotadas, em geral, duas abordagens (ZHAO *et al.*, 2003):

1. **Abordagem global (Aparência da Face):** consiste em obter um conjunto de descritores numéricos a partir dos pixels da imagem. Mesmo com a presença de ruído produzido pelas variações de luminosidade, reflexos e outros fatores, pode-se distinguir uma face de outra adotando-se funções básicas de processamento de imagens para transformar a imagem da face.
2. **Abordagem local (Geometria da Face)**: tem como fundamento a modelagem da face, considerando informações geométricas dos elementos que a compõem, tais como: nariz, boca, olhos, bochechas, etc. Desta forma, o reconhecimento da face se resume na comparação dos valores das informações geométricas da face detectada na imagem de entrada do sistema biométrico com aquelas previamente cadastradas no banco de dados.

Segundo ZHAO et al. (2003), considerando as formas de extração das características da face, o processo de comparação está baseado em três tipos:

1. **Métodos holísticos:** relacionados à abordagem global de extração das características da face, consideram toda a região da face. Destaca-se a técnica PCA, baseada nas autofaces (*eigenfaces*), dentre as muitas técnicas existentes;
2. **Métodos estruturais:** relacionados à abordagem local de extração das características da face, apresentam as técnicas mais recentes que utilizam medidas geométricas como ângulos e distâncias relativas entre os elementos componentes da face como, por exemplo, a boca e o nariz;
3. **Métodos híbridos**: buscam misturar o melhor dos dois métodos, na tentativa de se aproximarem do sistema humano de percepção, que utiliza tanto a aparência global da face quanto suas características locais.

Independentemente do método de comparação adotado, a eficácia do processo de reconhecimento de faces está condicionada às condições de obtenção da imagem. Se alguma das características mudar de forma significativa, como por exemplo, cerramento dos olhos, uso de óculos, alterações de expressões faciais, ou se as condições do ambiente não forem adequadas apresentando excesso ou falta de luminosidade, o resultado pode ser bastante comprometido. Mesmo em condições controladas, os algoritmos de reconhecimento de face

podem apresentar altas taxas de erros. Portanto, o desempenho dos sistemas de reconhecimento de face depende muito da aplicação, e os bons resultados obtidos em laboratório ou experiências em ambientes controlados não garantem necessariamente os mesmos resultados na prática (ZHAO *et al.*, 2003).

A tecnologia de autenticação biométrica baseada no reconhecimento facial apresenta os seguintes pontos favoráveis (LIN, 2000):

- alto nível de aceitabilidade, visto que faz parte do processo natural de reconhecimento de faces entre os humanos;
- sistema de autenticação menos intrusivo, pois os sistemas de reconhecimento de face não exigem contato com o sensor e podem até mesmo ser executados sem a colaboração do usuário;
- baixo custo dos dispositivos de aquisição de imagens 2D, pois as câmeras já são integradas a notebooks e dispositivos móveis;

Como pontos desfavoráveis, podemos citar:

- necessidade de um ambiente controlado, principalmente quanto às condições de iluminação;
- apesar de ser eficiente em aplicações de identificação de pequena escala, em aplicações de identificação de larga escala ainda é considerada uma biometria pobre;
- disfarces podem ser usados para fraudar o sistema.

2.3.1 Detecção da Face

A localização e segmentação da face presente em uma imagem é o passo inicial do processo de reconhecimento automático de faces. Esta etapa é considerada crítica para o processo de reconhecimento facial visto que uma falha nesta fase pode comprometer todo o processo.

Um dos algoritmos mais eficazes para a detecção de faces em imagens estáticas é o algoritmo Viola-Jones (VIOLA; JONES, 2001), que se encontra implementado na biblioteca

Intel OpenCV (*Intel Open Source Computer Library*) e que visa localizar em uma imagem características que codifiquem alguma informação do padrão sendo detectado. Esses padrões são baseados nas características de Haar (Figura 2.4) que codificam informações sobre a existência de contrastes orientados entre regiões da imagem (BRADSKI; PISAREVSKY, 2000). A Figura 2.4 apresenta algumas características de Haar propostas para a detecção de faces em imagens estáticas.

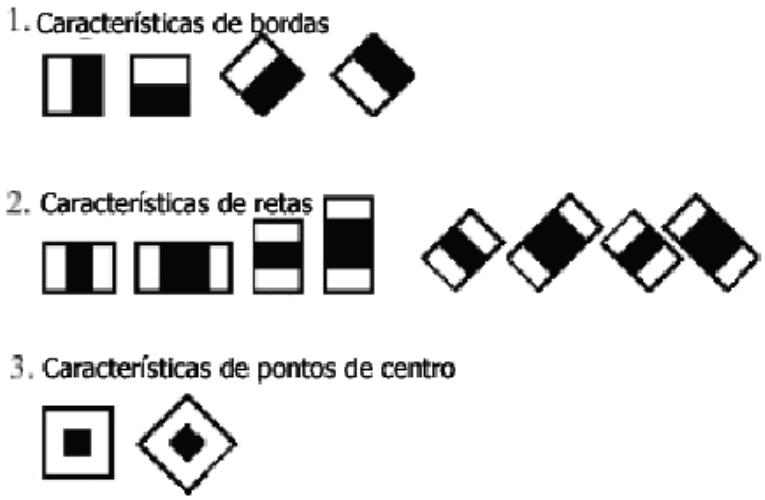

Figura 2.4. Características de Haar mais usadas em detecção de faces (VIOLA; JONES, 2001).

Quando as características de Haar são aplicadas em uma imagem, são examinados os contrastes naturais proporcionados pelas características da face, considerando suas relações de espaço, como ilustrado na Figura 2.5.

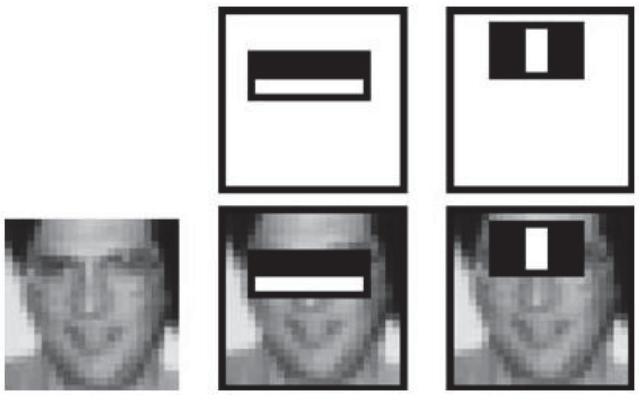

Figura 2.5. Relação entre as características de Haar e os contrastes naturais da face (VIOLA; JONES, 2004).

A fim de computar as características de Haar de forma eficiente, uma representação

intermediária para a imagem original é gerada. Esta imagem é conhecida como imagem integral e é gerada de tal modo que o valor armazenado no pixel (x, y) da imagem integral corresponde à soma dos valores de todos os pixels acima e à esquerda de (x,y), inclusive.

Para se gerar a imagem integral em um único passo sobre a imagem original, usa-se recorrentemente a equação 2.1:

$$s(x, y) = s(x, y - 1) + i(x, y)$$
$$ii(x, y) = ii(x - 1, y) + s(x, y) \quad (2.1)$$

onde ii é a imagem integral, s(x,y) é a soma cumulativa da linha, s(x, -1) = 0 e ii(-1, y) = 0.

A representação da imagem integral pode ser observada na Figura 2.6. O ponto ii(x,y) na imagem armazena o somatório de todos os pixels desde a origem (0,0) até o ponto (x,y), inclusive.

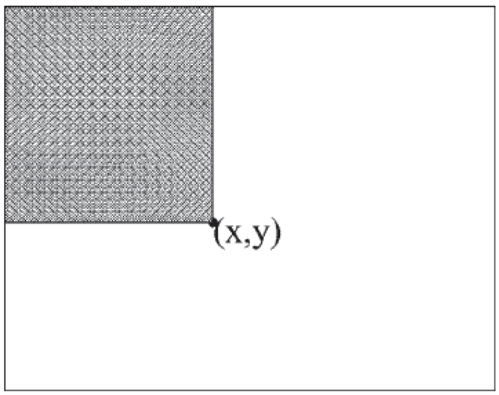

Figura 2.6. Representação da imagem integral (VIOLA; JONES, 2004).

A partir da imagem integral, a soma dos valores dos pixels de uma região retangular qualquer da imagem pode ser calculada de modo eficiente usando apenas os valores dos quatro pontos da imagem integral que delimitam o retângulo, e operações aritméticas simples de soma e subtração, conforme ilustrado pela Figura 2.7.

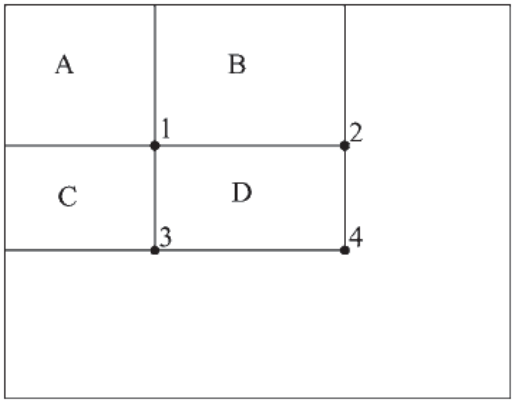

Figura 2.7. A partir da imagem integral, a soma da região D pode ser eficientemente calculada, utilizando-se apenas os valores da imagem integral nos vértices identificados por 1, 2, 3 e 4 (VIOLA; JONES, 2004).

O cálculo da região D ilustrada na Figura 2.7 pode ser representado pela fórmula:

$$D = ii(4) - ii(2) - ii(3) + ii(1) \qquad (2.2)$$

Apesar da facilidade de encontrar as características de Haar em uma imagem integral, como ilustrado na Figura 2.7, uma subjanela de qualquer tamanho, a quantidade possível de combinações das características de Haar é muito grande, de modo que, a fim de agilizar o processo de classificação, obtém-se um pequeno subconjunto composto das características mais representativas excluindo a maioria das características disponíveis. O método escolhido para esta etapa do processo é o método AdaBoost (VIOLA; JONES, 2004), cujo funcionamento visa construir um classificador “forte” baseado na combinação de subclassificadores “fracos” que dependem de uma única característica.

Então, uma estrutura em cascata, resultado da combinação dos classificadores fracos, é utilizada a fim de obter uma redução no tempo de processamento deste algoritmo. O princípio de funcionamento deste procedimento em cascata ajusta os classificadores para conseguirem altas taxas de detecção e, então, determina que a avaliação de um segundo classificador só será invocada caso a avaliação do primeiro seja positivo. Caso contrário, o procedimento é interrompido e a sub-janela rejeitada. Portanto, é necessário um resultado positivo em todos os classificadores para que a detecção do padrão em uma subjanela tenha êxito (VIOLA; JONES, 2004). Este procedimento é ilustrado na Figura 2.8.

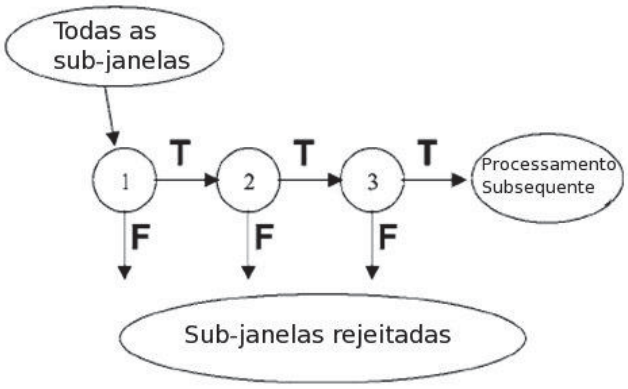

Figura 2.8. Funcionamento do algoritmo em cascata do algoritmo Viola-Jones (VIOLA; JONES, 2004).

A justificativa para a escolha desta técnica de detecção de faces no sistema que foi proposto e implementado neste trabalho está justamente em sua característica principal: o bom desempenho, essencial para equipamentos que apresentam restrições no que tange à capacidade computacional, como os dispositivos móveis. Este bom desempenho é propiciado pelo conjunto de passos do algoritmo permitindo que ele apresente uma taxa de detecção tão boa quanto outras apresentadas na literatura, porém com um tempo de processamento substancialmente menor (VIOLA; JONES, 2001).

2.3.2 Extração das Características Faciais com PCA

Após a etapa de detecção e segmentação da face presente na imagem, a próxima fase do reconhecimento é a extração das características.

Neste trabalho, uma das técnicas avaliadas para a descrição das faces é baseada na Análise das Componentes Principais (PCA – *Principal Component Analysis*) (KIRBY; SIROVICH, 1990; TURK; PENTLAND, 1991), cuja característica mais marcante é a capacidade de compressão por meio da descorrelação dos dados presentes na imagem, que pode ser realizada analisando-se a variância em cada uma de suas dimensões. A técnica baseada em PCA reduz a dimensionalidade dos dados iniciais, conservando as dimensões de maior variância. Como resultado, os dados são comprimidos sem perdas significantes de informação promovendo a minimização do esforço computacional.

A grande correlação existente entre os valores dos pixels de uma imagem de face faz com que os dados de uma face correspondam a um subespaço do espaço vetorial de uma

imagem. O método da análise das componentes principais identifica e representa eficientemente este subespaço. Este cenário pode ser observado na Figura 2.9.

Em sua fase de treinamento, o algoritmo PCA busca encontrar o conjunto de autovetores (bases) que formam o subespaço de faces.

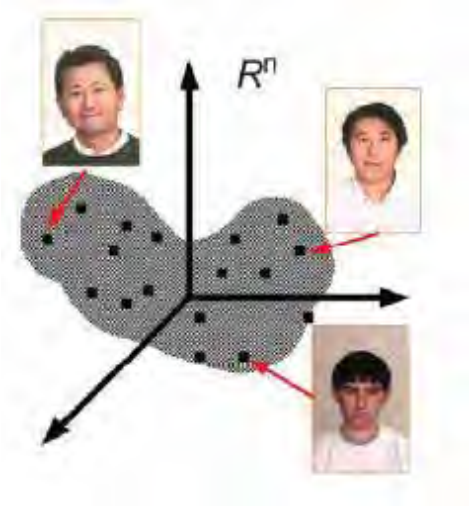

Figura 2.9. Representação do subespaço das faces no espaço da imagem de entrada (JAIN, 2004).

A equação 2.2 representa o problema cuja solução é a base para o subespaço de faces:

$$\lambda = \Phi^T \Sigma \Phi \qquad (2.3)$$

onde Σ é a matriz de covariância entre as *n* dimensões, Φ é a matriz de autovetores de Σ e λ é matriz diagonal contendo os autovalores λ_i.

O processo de reconhecimento facial baseado na análise das componentes principais (PCA) é composto das seguintes etapas:

- um conjunto de imagens selecionadas é usado para a criação do espaço de faces;
- são coletadas imagens de faces de usuários que são projetadas neste espaço gerando vetores de características que representam as faces os quais são armazenados;
- na comparação de uma face amostrada esta é projetada no espaço de faces para obter o vetor de características que é comparado com os vetores das faces previamente cadastradas;
- vários métodos podem ser adotados para se calcular a distância da face

amostrada com as faces cadastradas como, por exemplo, a distância euclidiana;

- o valor da distância da face mais próxima com relação à amostrada é comparado a um limiar e caso seja maior ou igual o usuário é então autenticado.

2.3.3 Extração das Características com LBP

Outra técnica de descrição de faces avaliada neste trabalho é baseada em padrões binários locais (LBP - *Local Binary Pattern*), também conhecida como Transfomada Census.

O operador LBP, elemento principal desta técnica de reconhecimento facial, é conhecido pelo seu ótimo desempenho como descritor de texturas e por isso tem sido amplamente adotado em vários tipos de aplicações e, em particular, para a descrição de faces (AHONEN; HADID; PIETIKÄINEN, 2006).

Se entendermos que uma face é uma composição de micropadrões podemos admitir que o operador LBP pode ser usado para descrevê-la (AHONEN; HADID; PIETIKÄINEN, 2006).

O operador LBP atribui um rótulo a cada pixel da imagem comparando o valor do tom de cinza deste pixel com os da vizinhança. Em um movimento circular no sentido horário, a iniciar do pixel acima e à esquerda do pixel central, o valor do tom de cinza desse pixel central é comparado com cada um dos seus vizinhos. Em cada comparação, se o valor do tom de cinza do pixel vizinho for maior ou igual ao do pixel em questão este vizinho é rotulado com valor 1, caso contrário com valor 0. Ao alinharmos estes resultados na mesma sequência em que foram calculados, teremos uma representação numérica de base 2. Este valor é então convertido para a base decimal e depois associado ao pixel central. Este processo é ilustrado na Figura 2.10.

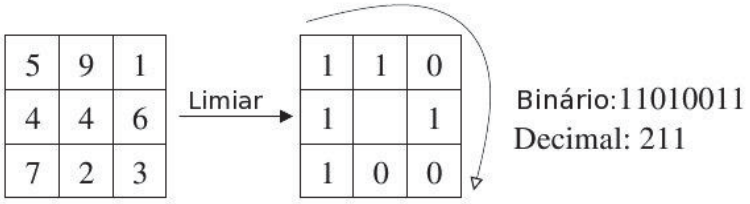

Figura 2.10. Representação do modo de atuação do operador LBP (AHONEN; HADID; PIETIKÄINEN, 2004).

Uma vez calculado o operador LBP em cada pixel da imagem o passo seguinte é a geração do histograma. Esta distribuição de frequências dos valores dos operadores LBP é usada como a descrição facial e é conhecida como descritor LBP (AHONEN; HADID; PIETIKÄINEN, 2006).

As configurações do raio de atuação e a quantidade de amostras dentro deste raio a serem consideradas podem alterar o desempenho do algoritmo. Na literatura a notação desta configuração é (P,R), onde P representa a quantidade de vizinhos considerados no raio R. Nos primeiros trabalhos, a configuração de raio e vizinhos amostrados era (8,1), ou seja, eram considerados oito vizinhos com um raio de atuação igual a um. Porém, logo surgiram novas propostas com outras configurações como, por exemplo, (8,2) (AHONEN; HADID; PIETIKÄINEN, 2006).

Em alguns casos, o ponto de amostragem não coincide com a posição de um pixel, tornando necessário o uso de interpolação bilinear para se determinar o valor do tom de cinza naquele ponto (AHONEN; HADID; PIETIKÄINEN, 2006). A Figura 2.11 ilustra uma configuração de vizinhança (8,2) circular, que utiliza a interpolação bilinear para calcular os valores nos pontos que não coincidem com os centros dos pixels da imagem.

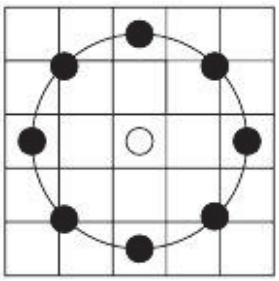

Figura 2.11. Vizinhança (8,2) circular. Os pontos que não coincidem com o centro de um pixel tem seu valor bilinearmente interpolado (AHONEN; HADID; PIETIKÄINEN, 2004).

Outra técnica proposta para melhorar o desempenho do algoritmo com relação a sua eficácia consiste na divisão da imagem da face em regiões de igual tamanho, a partir das quais são calculados os histogramas de operadores LBP, que são concatenados para serem usados como um descritor único da face (AHONEN; HADID; PIETIKÄINEN, 2004).

A Figura 2.12 ilustra a representação da face por meio de operadores LBP. Nota-se que a imagem foi dividida em janelas onde cada janela resulta em um histograma que é concatenado para compor o descritor da face.

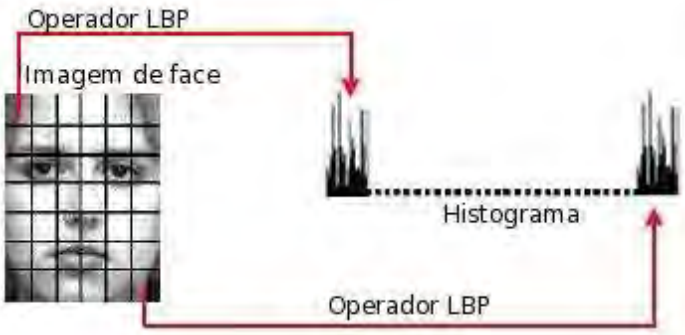

Figura 2.12. Representação da face usando operadores LBP com janelamento (CHANG-YEON, 2008).

2.4 OpenCV

OpenCV (*Intel Open Source Computer Vision Library*) é o nome dado a uma biblioteca de código aberto, criada em 2000 por um grupo de desenvolvimento da Intel, que possui uma coleção de poderosas ferramentas destinadas ao desenvolvimento de aplicativos na área de Visão Computacional.

A biblioteca OpenCV é composta de funções para processamento de imagens e vídeos, entrada e saída de dados, estrutura de dados, álgebra linear, interface gráfica básica do usuário com sistema independente de janelas, controle de teclado e mouse, além de centenas de algoritmos destinados a área de visão computacional.

A biblioteca foi escrita usando a linguagem C/C++, é portável para Windows, Linux e MacOS, além de ser totalmente livre para uso acadêmico e comercial. Seu código fonte está disponível para que seus usuários possam alterá-lo, adequando-o a uma necessidade particular.

Dentre suas muitas funções, encontra-se a implementação da técnica de Viola-Jones, para detecção de faces, e uma implementação da técnica PCA para reconhecimento de faces, ambas usadas neste trabalho.

Os principais módulos da biblioteca OpenCV são:

- cv: contempla as funcionalidades e algoritmos destinados a visão computacional;
- cvaux: contempla os algoritmos destinados a área de visão computacional que ainda estão em fase experimental;
- cxcore: módulo de estruturas de dados e álgebra linear;

- highgui: módulo de controle de interface e dispositivos de entrada;
- ml: módulo de *Machine Learning*. Trata-se de um conjunto de classes e funções para a classificação estatística, *clustering*, etc;
- Cvcam: módulo portável para processamento de vídeo digital de câmeras;
- ed: trata-se de um manual de estrutura de dados e operações.

Uma visão geral da biblioteca OpenCV e seus principais módulos é apresentada na Figura 2.13.

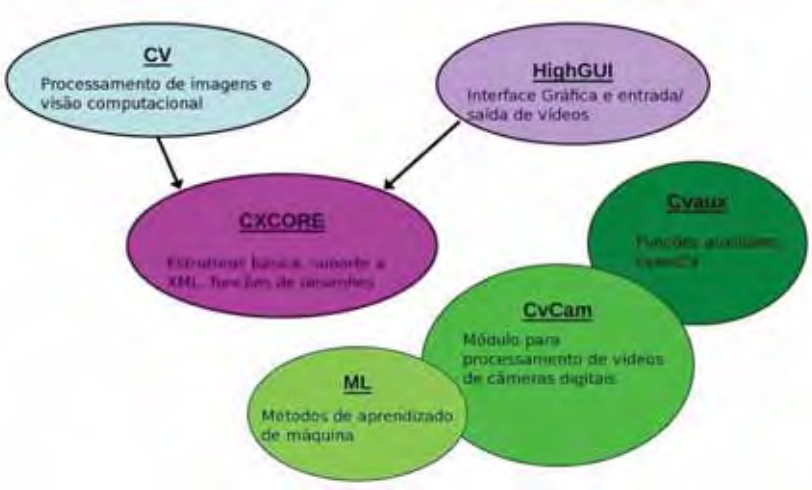

Figura 2.13. Os principais módulos da biblioteca OpenCV (KHALILI, 2007).

2.5 Considerações Finais

Neste capítulo foi realizada uma revisão da literatura sobre biometria e seus principais aspectos. Foram destacadas as principais características dos sistemas biométricos, assim como as técnicas para reconhecimento facial em suas fases de segmentação e descrição. Foram citados os algoritmos Viola-Jones, para detecção de faces, e as técnicas de descrição baseadas em PCA e LBP, usadas na fase de reconhecimento facial. Por fim, a última seção tratou da OpenCV, uma biblioteca de funções composta de centenas de algoritmos que apoiam o processamento de imagens em geral e o processamento biométrico baseado em reconhecimento de faces, alvo desta dissertação de mestrado.

CAPÍTULO 3. **Plataformas de Desenvolvimento para Dispositivos Móveis**

O objetivo deste capítulo é apresentar a revisão bibliográfica realizada sobre os dispositivos móveis abordando as principais plataformas de software disponíveis, bem como os ambientes de desenvolvimento oferecidos por estas plataformas.

3.1 Introdução

A partir dos anos 90 houve um substancial crescimento no desenvolvimento e popularização dos dispositivos computacionais móveis, como são chamados os dispositivos de pequeno porte (de mão) que possuem capacidade computacional e funcionalidades semelhantes a um microcomputador. Esse crescimento levou ao surgimento de várias plataformas de execução e desenvolvimento de software (FIGUEIREDO; NAKAMURA, 2003).

Segundo ROGERS et al. (2009), essas plataformas são também conhecidas como pilhas de software (*software stacks*) devido à combinação de softwares de propósitos variados como, por exemplo, sistema operacional, ambiente de execução e compilador, com o objetivo de criar um ambiente de execução e desenvolvimento de software para uma classe de dispositivo.

Dentre as plataformas de execução e desenvolvimento de software para dispositivos móveis destacam-se: Symbian OS, Java Micro Edition (JME), Microsoft e Android, descritas nas próximas seções.

3.2 A Plataforma Symbian OS

O sistema operacional Symbian (SYMBIAN, 2009) se destaca entre os sistemas operacionais destinados a dispositivos móveis, sendo atualmente um dos mais populares em sua categoria.

Desde o início, a idéia que veio a se concretizar no Symbian foi fundamentada em colaboração, começando com os projetos de jogos e software de produtividade de escritório para computadores pessoais Sinclair, de David Potter, no início dos anos 80, uma parceria que foi lançada com o nome de "Psion". Esses programas foram fundamentais para o surgimento, em 1984, do Psion Organizer, o primeiro computador portátil do mundo, que logo em seguida suportaria a linguagem de programação de banco de dados OPL.

O apoio colaborativo da indústria promoveu o crescimento do software Psion levando a formação em 1998 da *join venture* Symbian, formada pela Psion, Ericsson, Motorola e Nokia. Presente em mais de 100 milhões de telefones até 2006, a Symbian contribuiu substancialmente para a explosão da inovação de dispositivos móveis.

O passo seguinte da inovação Symbian ocorreu em 2008 com a compra de todos os ativos da Symbian pela finlandesa Nokia, colocando o software no caminho do *open source*. Foi criada então a Symbian Foundation para prover, como todos os seus membros e associados, a evolução do software com trabalho colaborativo visando um sistema mais criativo e produtivo e proporcionando uma experiência mais agradável ao usuário final (SYMBIAN, 2009).

A Symbian, que é a parte central da Fundação Symbian, é uma plataforma de software *open-source* para dispositivos móveis cujas principais características são as seguintes (SYMBIAN, 2009):

- Possui um sistema operacional, um *middleware* e camadas de interface com o usuário;
- A pilha de software é completa e bem integrada, proporcionando todos os recursos necessários para o desenvolvimento de dispositivos, aplicações e serviços;
- É composta de contribuições incluindo S60, Symbian OS, UIQ e MOAP(S);
- Possui funcionalidade com aproximadamente 20 milhões de linhas de código;
- É implementada com o mesmo código vendido em mais de 250 milhões de aparelhos de 14 dos maiores fabricante de telefones do mundo.

Por ser *open-source*, apesar de ser bastante usada por fabricantes de telefones, qualquer programador ou empresa pode contribuir para o desenvolvimento ou aperfeiçoamento da plataforma Symbian. Outra característica interessante da Symbian é a possibilidade de instalação de novas aplicações, além das oferecidas pelo fabricante.

O código da plataforma Symbian está sob a licença SFL (*Symbian Foundation*

License) e disponíveis aos membros da Fundação Symbian, que podem ter e modificar o código, e criar e distribuir novas soluções sobre a plataforma.

A partir do segundo semestre de 2010 o código da plataforma Symbian passou a ser distribuído sob a licença EPL (*Eclipse Public License*) (SYMBIAN, 2009).

Os *frameworks* de interface de usuário não são oferecidos separadamente, mas são parte integrante da plataforma. Os fabricantes de dispositivos podem ainda optar por personalizar a aparência (*look and feel)* da interface do usuário de seus dispositivos para manter a diferenciação de consumo.

Vários ambientes de programação são suportados pela plataforma incluindo C/C++, Ruby, ambiente de execução Web baseado no Webkit e Python. Outras tecnologias como Java ME, Flash Lite e .Net também deverão ser integradas à plataforma.

Para criar um ambiente de desenvolvimento completo, é preciso ter um kit de ferramentas e pelo menos um kit de software. Os kits são direcionados principalmente para desenvolvedores de aplicação ou desenvolvedores de produtos (que estão usando a plataforma Symbian para criar os dispositivos e/ou desenvolvimento de código para contribuir ou complementar a plataforma) (SYMBIAN, 2009).

Para um desenvolvedor de aplicação em Symbian, os kits necessários são (SYMBIAN, 2009): o Application Development Toolkit (ADT) e o Application Development SDK (SDK).

Os seguintes kits são necessários aos desenvolvedores de dispositivos ou contribuidores (SYMBIAN, 2009): o Product Development Toolkit (PDT) e o Product Development Kit (PDK).

Os membros da Fundação Symbian ainda podem utilizar um kit extra, o interim Symbian Software Kit (iSSK), instalado no topo do PDT e PDK como representado na Figura 3.1 (SYMBIAN, 2009).

Um desenvolvedor de aplicativos também pode optar por usar o PDK, a fim de obter acesso a API interna, porém não há promessa de compatibilidade. Como ele contém o código-fonte, ambos são uma excelente referência para uso da API, e podem ser usados para criar produtos derivados (SYMBIAN, 2009).

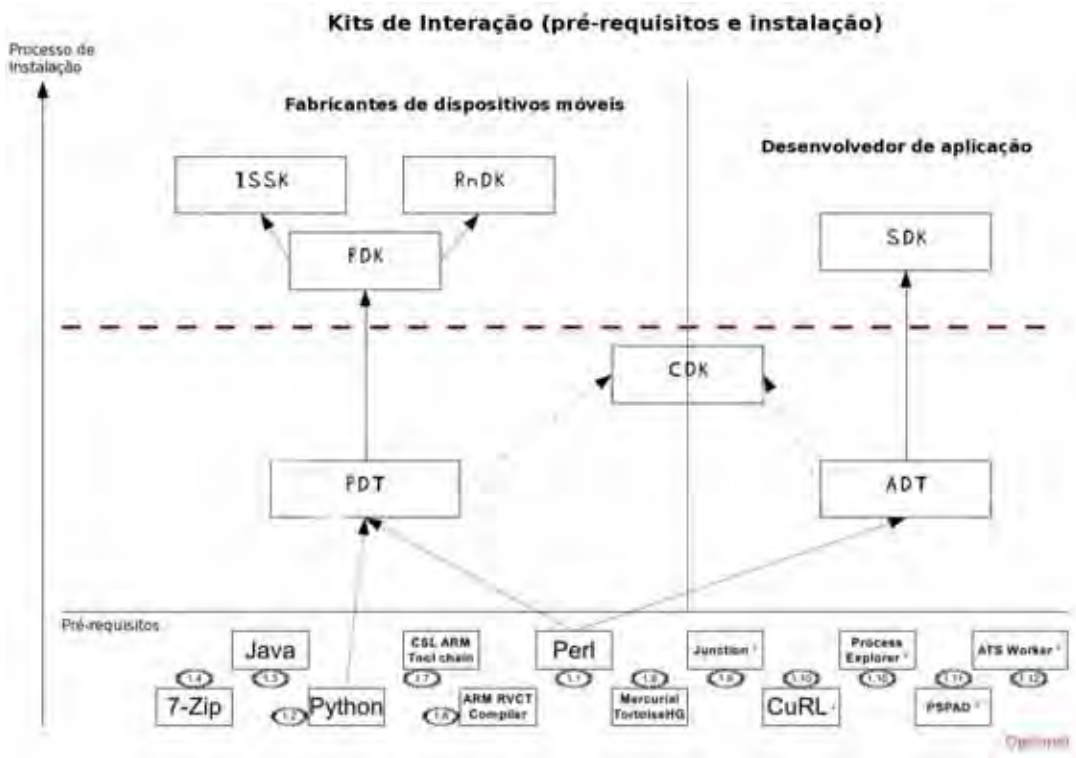

Figura 3.1. Representação Gráfica das Toolkits da plataforma Symbian (SYMBIAN, 2009).

3.3 A Plataforma Java Micro Edition

O propósito original da criação da tecnologia Java Micro Edition (ME) foi lidar com as limitações associadas com o desenvolvimento de aplicativos para pequenos dispositivos. Desse modo, as características e os fundamentos da tecnologia Java ME foram definidos pela Sun visando tornar possível o desenvolvimento de aplicações considerando as limitações desses dispositivos, tais como: memória reduzida, restrições de energia, menor poder de processamento, display pequenos, dentre outros.

Nesta seção são abordadas as principais características da plataforma Java ME, assim como da linguagem de programação Java.

3.3.1 Características da Plataforma

A composição da plataforma Java ME (SUN MICROSYSTEM, 2009) considera uma coleção de tecnologias e especificações que permitem a combinação a fim de construir um ambiente de execução Java completo para se adequar às exigências de um dispositivo

específico ou de mercado. Esta flexibilidade permite a coexistência transparente de vários fornecedores cooperando para produzir uma experiência mais atraente ao usuário final.

Três pilares sustentam a tecnologia Java ME (SUN MICROSYSTEM, 2009):

- Uma configuração que fornece o conjunto mais básico de bibliotecas e recursos de máquina virtual para uma ampla gama de dispositivos;
- Um conjunto de APIs que suporta uma gama reduzida de dispositivos; e
- Um pacote opcional de APIs de tecnologia específica.

A plataforma Java ME foi dividida em duas configurações básicas, uma destinada a pequenos dispositivos com recursos de hardware bastante limitados, como os telefones celulares, e outra destinada a dispositivos com um pouco mais de recursos, como os *smartphones*.

A configuração base para os pequenos dispositivos é conhecida por CLDC (*Connected Limited Device Configuration*) e a configuração para dispositivos com mais recursos de hardware é conhecida por CDC (*Connected Device Configuration*).

A Figura 3.2 mostra os componentes da tecnologia Java ME e como elas estão relacionadas com as outras tecnologias Java. Nesta figura pode-se observar o modo como a tecnologia é arranjada para suportar as diferentes plataformas sob as quais as aplicações Java poderão ser executadas. Note que os arranjos em destaque estão relacionados à tecnologia Java ME destinada a dispositivos móveis e dispositivos não-PC.

Como mencionado, a configuração para dispositivos pequenos como os telefones celulares comuns é chamada de CLDC (*Connected Limited Device Configuration*). Esta configuração, ilustrada na Figura 3.3, foi especificamente concebida para satisfazer as necessidades de uma plataforma Java executada em dispositivos com recursos bastante limitados de memória, poder de processamento e capacidade gráfica.

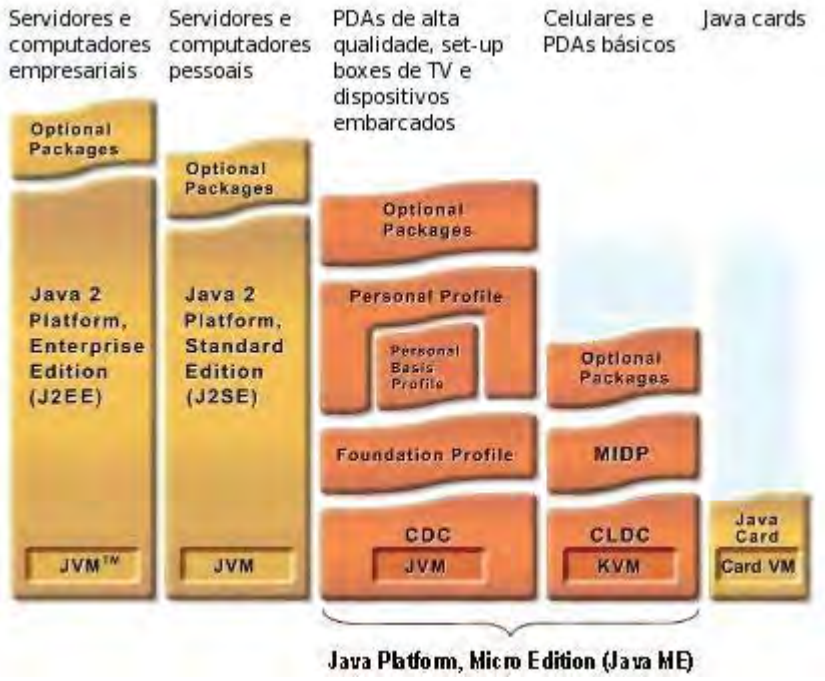

Figura 3.2. A Plataforma Java (SUN MICROSYSTEM, 2009).

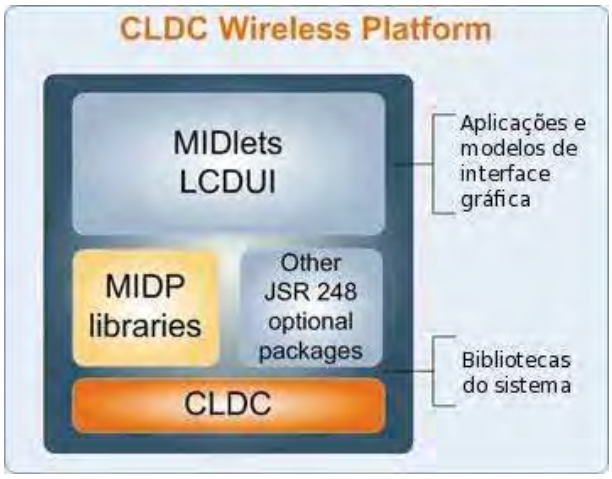

Figura 3.3. Configuração para dispositivos pequenos (CLDC) (SUN MICROSYSTEM, 2009).

Além das diferentes configurações, a plataforma Java ME também especifica um número de perfis que definem um conjunto de APIs de nível superior que melhor definem a aplicação. Um bom exemplo, que é largamente adotado, é combinar a CLDC com o *Mobile Information Device Profile* (MIDP) para fornecer um ambiente para aplicação Java destinado a telefones celulares e outros dispositivos com as mesmas capacidades.

Uma aplicação criada por um desenvolvedor de software Java ME como, por exemplo, um jogo ou uma aplicação de negócio é chamado de MIDlet. Tais MIDlets podem ser escritos

uma única vez e serem executados em todos os dispositivos disponíveis em conformidade com as especificações para a tecnologia Java ME. Eles podem residir em um repositório e o usuário final pode procurar um tipo específico de aplicação e transferir tal aplicação para seu dispositivo.

A Figura 3.4 ilustra a configuração oferecida pela tecnologia Java ME para dispositivos com menor limitação de hardware, como *smartphones* e conversores para TV digital, e chamada de CDC (*Connected Device Configuration*).

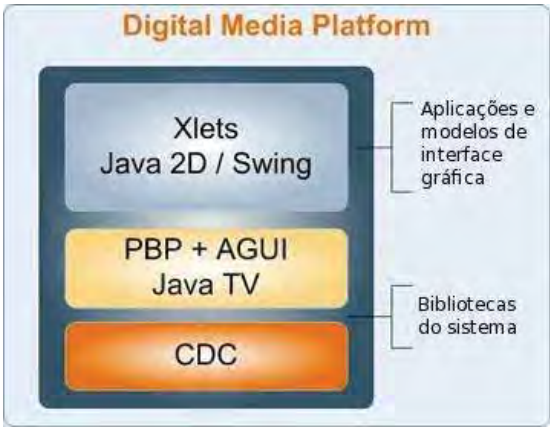

Figura 3.4. Configuração CDC (Connected Device Configuration) (SUN MICROSYSTEM, 2009).

Os objetivos da configuração CDC são promover as tecnologias e ferramentas de desenvolvimento baseadas na plataforma Java Standard Edition (SE) e suportar o conjunto de recursos de uma ampla gama de dispositivos móveis ponderando suas limitações de recursos.

A configuração CDC apresenta as seguintes vantagens para os diferentes grupos na cadeia de negócios:

- As empresas podem se beneficiar do uso de aplicações baseadas em rede que estendem o alcance da lógica de negócios para clientes móveis, parceiros e trabalhadores;
- Os usuários podem se beneficiar da compatibilidade e segurança da tecnologia Java;
- Os desenvolvedores podem se beneficiar da segurança e da produtividade da linguagem de programação Java e as APIs ricas em plataforma Java.

A configuração CDC apresenta três diferentes perfis:

- O Foundation Profile (JSR 219);
- O Personal Basis Profile (JSR 217); e
- O Personal Profile (JSR 216)

Cada um destes perfis possui um conjunto de pacotes opcionais em que a aplicação é executada.

3.3.2 A Linguagem Java

Java é uma das linguagens mais adotadas atualmente, especialmente em projetos Web e para dispositivos não-PC. Nos últimos anos Java tem conquistado grande participação no mercado de dispositivos móveis.

Fortemente orientada a objetos, esta linguagem fornece um ambiente de execução multiplataforma conhecido como Java Virtual Machine (JVM), que possibilita que programas escritos em uma determinada plataforma possam ser facilmente executados em outra. Para isso, a compilação de um programa Java gera um *bytecode* que, posteriormente, é interpretado pela Java Virtual Machine (JVM), ao invés de gerar um código binário executável nativo, como faz a linguagem C, por exemplo.

De acordo com Deitel e Deitel (2005), a linguagem Java originou-se de um projeto de pesquisa interno financiado pela Sun Microsystem, em 1991, que resultou no desenvolvimento de uma linguagem baseada em C++ que seu criador, James Gosling, chamou de Oak em homenagem a uma árvore de carvalho. Como esse nome já havia sido atribuído a outra linguagem de programação, o nome Oak foi alterado para Java, uma referência à cidade de origem de um tipo de café.

De acordo com Horstmann e Cornell (2003), a linguagem Java apresenta também as seguintes características:

- **Simplicidade**: possui sintaxe muito semelhante com a sintaxe das linguagens C e C++, porém mais simplificada, pois não trabalha com arquivos de cabeçalho, aritmética de ponteiros, estruturas, uniões, herança múltipla e outros;
- **Processamento Distribuído**: possui uma rica API com funções que permitem o acesso a objetos em uma URL da internet usando o protocolo TCP/IP, com a mesma facilidade dos acessos locais;
- **Robustez**: para possibilitar o desenvolvimento de programas mais confiáveis, Java propõe a verificação antecipada de possíveis problemas, a verificação dinâmica

posterior (em tempo de execução) e a eliminação de situações sujeitas a erros, como a aritmética de ponteiros das linguagens C e C++;

- **Segurança**: a proposta de Java é a construção de sistemas livre de vírus e adulterações. Para isso, além das verificações em tempo de compilação, várias outras verificações são realizadas na carga do programa e durante sua execução;
- **Portabilidade**: o programador Java não precisa se preocupar com aspectos dependentes da plataforma de execução, tais como o tamanho ocupado por um tipo primitivo, a implementação do sistema de arquivos, dentre outras. Basta conhecer a API de Java e deixar o resto por conta da máquina virtual específica de cada plataforma;
- **Interpretação**: independente da plataforma, desde que haja uma implementação da JVM, os *bytecodes* Java podem ser executados, mesmo que compilados em uma plataforma diferente daquela de destino.

3.4 A Plataforma Microsoft

O Windows Embedded CE 1.0, lançado em novembro de 1996, marcou o inicio da participação da Microsoft no mercado de sistemas embarcados. A partir deste fato, a Microsoft tem se expandido em uma linha completa de sistemas operacionais embarcados, permitindo que desenvolvedores criem dispositivos de 32 bits com uma vasta gama de produtos que fornecem conjuntos de ferramentas e plataformas de desenvolvimento para dispositivo de pequeno a grande porte (MICROSOFT, 2010).

Com vastos recursos, ferramentas fáceis de usar, kits de avaliação gratuitos e acesso uma grande rede de apoio da comunidade, Windows Embedded ajuda a reduzir o tempo de introdução do produto ao mercado, além de reduzir os custos com o desenvolvimento (MICROSOFT, 2010).

3.4.1 A Família Windows Embedded

A Microsoft oferece uma versão específica do Windows Embedded para cada tipo de dispositivo, tanto para dispositivos móveis como celulares e *smartphones*, como para outros

dispositivos conhecidos como não-PC. Entre os dispositivos não-PC destacam-se os quiosques de consulta (como aqueles disponíveis em *shopping centers*, para localização dos clientes), navegadores GPS, caixas-eletrônicos, urnas-eletrônicas e equipamentos industriais (MICROSOFT, 2010).

3.4.2 O Windows Mobile

Para os dispositivos móveis, a Microsoft disponibiliza o Windows Mobile, uma versão variante da versão Windows Embedded CE, parte integrante da família Windows Embedded, destinada a dispositivos pequenos.

O Windows Mobile leva a familiaridade da área de trabalho Windows para os dispositivos móveis e, baseando-se no Windows Embedded CE, oferece suporte à .NET Compact Framework (MICROSOFT, 2010).

A plataforma Windows Mobile oferece recursos como conectividade de dados, rica API para suporte a funcionalidades e serviços como Bluetooth e o Pocket Outlook Object Model (POOM), uma abrangente gama de modelos de programação que inclui código nativo, código gerenciado (*managed code*) e desenvolvimento para web e recursos de dispositivo como múltipla linhas de processamento (*multithreading*) (Microsoft, 2010).

Apesar de ter seu núcleo baseado no Windows CE, o Windows Mobile fornece muitas características únicas, como a Shell e suporte de comunicação, que o torna ideal para uso em dispositivos móveis, como celulares e assistentes pessoais digitais (MICROSOFT, 2010).

Os dispositivos equipados com o Windows Mobile são classificados em dois grupos principais: os dispositivos que possuem telas sensíveis ao toque (*touchscreen*), e os dispositivos que não possuem tela sensível ao toque.

O Visual Studio, um ambiente de desenvolvimento integrado (IDE – *Integrated Development Environment*), é a peça chave para o desenvolvimento de aplicativos para dispositivos Windows Mobile. Ele fornece as ferramentas necessárias para o desenvolvimento de aplicações em código nativo com o Visual C++, ou código gerenciado com o Visual C#, Visual Basic, ou qualquer combinação destas linguagens. Independente da linguagem, a IDE oferece um amplo suporte para acesso a dados e para o .Net Compact Framework, uma versão do .Net Framework da Microsoft para dispositivos pequenos. O IDE também oferece suporte

completo a depuração e emulação de dispositivos assim como a criação de pacotes para implantação da aplicação no dispositivo (MICROSOFT, 2010).

3.5 A Plataforma Android

O Google Android é uma plataforma de código aberto, destinado a dispositivos móveis cuja composição inclui um sistema operacional de kernel Linux, *middleware*, aplicativos e interface de usuário (ROGERS et al., 2009).

Esta plataforma fornece um Kit de Desenvolvimento de Software (SDK – *Software Development Kit*) que provê ferramentas e um conjunto de bibliotecas necessárias para iniciar o desenvolvimento de aplicações na plataforma Android usando a linguagem de programação Java (ANDROID, 2009).

As principais características da plataforma Android são (LECHETA, 2009):

- ***Framework* da aplicação**: que possibilita a reutilização e substituição de componentes;
- **Máquina virtual "Dalvik":** otimizada para dispositivos móveis;
- **Navegador de internet integrado**: baseado no módulo de código aberto "WebKit";
- **Gráficos otimizados:** através de uma personalizada biblioteca gráfica 2D;
- G**ráficos 3D**: baseados na especificação OpenGL ES 1.0 (aceleração de hardware opcional);
- **SQLite**:para armazenamento estruturado de dados;
- **Suporte a mídias:** áudio, vídeo e os formatos de imagem (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF);
- **Telefonia GSM**;
- **Conectividade**: Bluetooth, EDGE, 3G, e WiFi;
- **Câmera, GPS, bússola e acelerômetro**;
- **Ambiente de desenvolvimento rico**: incluindo um emulador de dispositivo, ferramentas para depuração, perfis de uso de memória e desempenho e um *plugin*

para o ambiente de desenvolvimento Eclipse.

A quase totalidade do Google Android é distribuída sob a licença Apache 2.0, exceto algumas partes como, por exemplo, o Linux kernel patches que estão sob a licença GPLv2, como descreve LECHETA (2009).

3.5.1 Histórico

Em novembro de 2007, um grupo liderado pela Google e outras 33 empresas anunciaram a formação da Open Handset Alliance – OHA (Aliança de Telefonia Móvel Aberta) (ANDROID, 2009). Segundo ROGERS et al. (2009), o comunicado coletivo a imprensa naquele dia foi o seguinte:

> Esta Aliança partilha uma meta comum de inovação em dispositivos móveis e de fornecimento aos consumidores de uma experiência de usuário muito superior a de muitos produtos disponíveis em plataformas móveis da atualidade. Fornecendo aos desenvolvedores um novo nível de abertura que permite um trabalho mais colaborativo, o Android acelerará o ritmo em que novos serviços móveis e competitivos serão disponibilizados aos consumidores (ROGERS et al., 2009).

Por meio da comunidade *open source*, a aliança integra softwares e outras propriedades intelectuais fornecidos pelas empresas que a compõem. O licenciamento do software é feito por meio da licença Apache V2 que permite flexibilidade de uso, alteração do código Android e até mesmo que, após alterado, o código se torne proprietário da empresa que o alterou (ANDROID, 2009).

Entre os participantes da aliança podemos citar algumas grandes empresas como as fabricantes de aparelhos telefônicos HTC, LG, Motorola e Samsung, operadoras de telefonia móvel China Mobile Communications, KDDI, DoCoMo, Nextel, T-Mobile, Telecom Italia e Telefonica, as empresas de semicondutores Audience, Broadcom, Intel, Marvell, Nvidia Qualcomm, Synaptics e Texas Instruments e as empresas de software Ascender, eBay, Google, LivingImage, LiveWire e SONiVOX (ANDROID, 2009).

3.5.2 A Arquitetura

A Figura 3.5 ilustra a arquitetura da plataforma Android. Nela pode ser observada uma composição formada por cinco camadas: Linux Kernel, Libraries, Android Runtime, Application Framework e Applications (ANDROID, 2009).

Baseada no kernel Linux e na linguagem de programação Java, estas camadas apresentam as seguintes características (ANDROID, 2009):

- **Camada de aplicações (*Applications*)**: É a camada mais próxima do usuário e consiste em todos os aplicativos disponíveis que acompanham o Android, como o cliente de email, programa de SMS, calendário, mapas, navegador e agenda de contatos, além dos outros aplicativos fornecidos por terceiros ou desenvolvidos pelo próprio usuário. Todos os aplicativos são escritos na linguagem de programação Java;

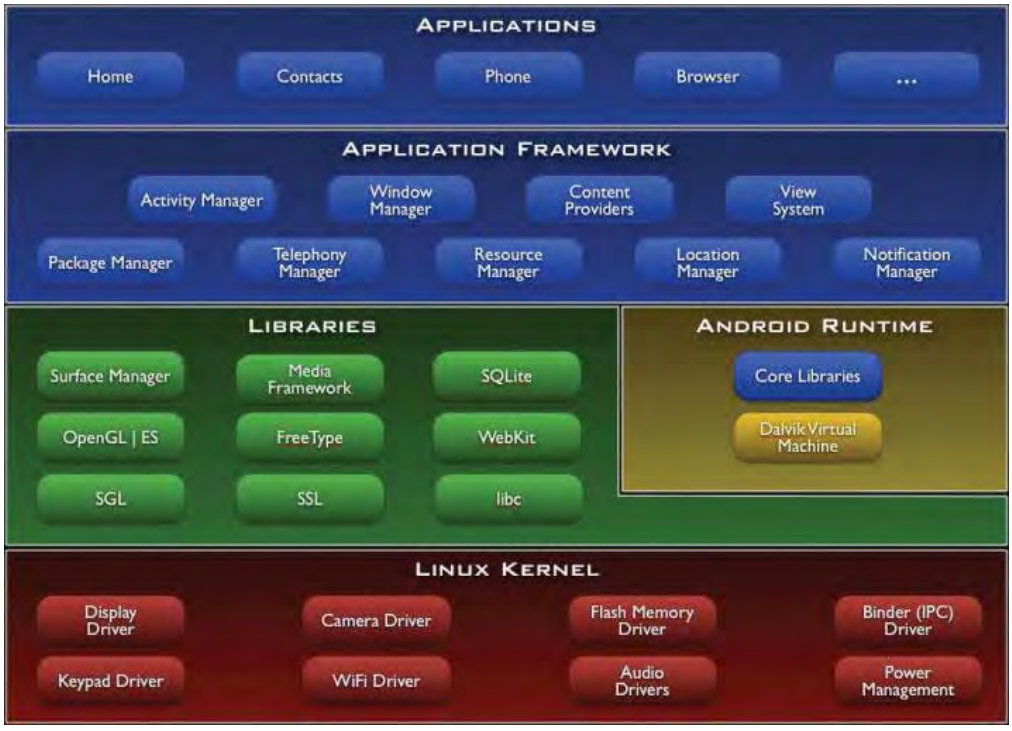

Figura 3.5. A arquitetura da plataforma Android (ANDROID, 2009).

- **Camada *Framework* da aplicação (*Application Framework*)**: Os desenvolvedores têm liberdade para usar os recursos do hardware do dispositivo, acessar informação sobre localização, executar serviços em segundo plano (*background services*), configurar alarmes, adicionar notificações na barra de

estado (*status bar*) e muito mais. Os desenvolvedores podem acessar sem restrições o mesmo conjunto de bibliotecas usado por aplicações de núcleo. O reuso de componentes é favorecido pela arquitetura da aplicação que permite que qualquer aplicação publique suas funcionalidades permitindo, desta forma, que outras aplicações façam uso destas funcionalidades (sujeito às regras de segurança imposta pelo framework). Este mesmo mecanismo permite a substituição de componentes pelo usuário;

- **Camada das Bibliotecas (*Libraries*)**: Android oferece um conjunto de bibliotecas C/C++ utilizadas por vários componentes do Sistema Android. Estas funcionalidades estão disponíveis aos desenvolvedores por meio do *framework* da aplicação. Entre estas bibliotecas de núcleo citamos a biblioteca C do sistema que é uma implementação derivada da biblioteca padrão C do sistema (libc), bibliotecas de mídia que permitem a manipulação de formatos populares de áudio e vídeo, a SGL para gráficos 2D, bibliotecas 3D incluindo uma implementação baseada na biblioteca OpenGL ES 1.0 e SQLite que é um poderoso e leve mecanismo para bases de dados relacionais;
- **Camada do Ambiente de execução Android (Android *Runtime*):** Um conjunto de bibliotecas de núcleo que provê a maioria das funcionalidades disponíveis nas bibliotecas de núcleo a linguagem de programação Java. É importante observar que toda aplicação Android é executada em seu próprio processo, uma instância da máquina virtual (VM – Virtual Machine) Dalvik. Esta máquina virtual permite, de modo eficiente, múltiplas instâncias da VM. O arquivo executável da VM Dalvik VM possui extensão ".dex". A VM Dalvik depende do kernel Linux para funcionalidades como múltiplas linhas de processamento associadas a um único processo (*threading*) e gerenciamento de memória (baixo nível);
- **Camada do *Kernel* Linux**: Android conta com o Linux versão 2.6 para serviços de núcleo como segurança, gerenciamento de memória, drivers, etc. O *kernel* também atua como uma camada de abstração entre o hardware e o resto da pilha de software.

3.5.3 O Ambiente de Desenvolvimento

As aplicações Android são desenvolvidas em um ambiente hospedeiro e posteriormente são instaladas nos dispositivos móveis (ANDROID, 2009).

A plataforma Android, que adotou a linguagem Java (ver seção 3.3.2) para o desenvolvimento de aplicações, fornece um *software development kit* (SDK) composto de várias ferramentas, tais como um depurador, um plugin para o ambiente de desenvolvimento Eclipse e um emulador de dispositivo que permite que aplicações Android possam ser escritas sem a necessidade de um dispositivo real para testes (ROGERS et al., 2009).

É possível fazer a integração do SDK Android com outras IDEs como, por exemplo, o Netbeans da Sun, porém a IDE Eclipse apresenta a melhor integração por ter sido eleita a IDE padrão do desenvolvimento Android.

Para configurar o ambiente de desenvolvimento os seguintes passos são necessários (ROGERS et al., 2009):

- instalação do Java Development Kit (JDK) (Android requer a versão 5 ou superior do JDK);
- instalação do IDE Eclipse (versão 3.3 ou superior);
- instalação do SDK Android;
- configuração das variáveis de ambiente;
- e, por último, instalação do plugin do Android (ADT) no Eclipse.

Com esses passos pode ser criado um ambiente sofisticado de desenvolvimento na plataforma Android com recursos necessários ao desenvolvimento de aplicações ricas.

Apesar da linguagem Java ser a linguagem oficial no desenvolvimento de aplicações, a plataforma Android oferece um modo de invocar códigos nativos como, por exemplo, os escritos em C e C++. Esta ferramenta é chamada de Android NDK (*Native Development Kit*) e pode ser obtida no site do Android mantido pela Google (ANDROID, 2009). O Android NDK provê (ANDROID, 2009):

- um conjunto de ferramentas usadas para gerar biblioteca de código nativo a partir de fontes em C e C++;
- um modo de inserir a biblioteca de código nativo dentro dos pacotes de arquivos da aplicação conhecidos como .apks (*application packages*) que podem ser

implantados nos dispositivos Android;

- um conjunto de cabeçalhos (*headers*) do sistema nativo e bibliotecas que serão suportadas em todas as versões futuras da plataforma Android.

A versão atual do NDK suporta o conjunto de instruções de máquina do ARMv5TE e provê cabeçalhos para as seguintes bibliotecas:

- biblioteca padrão C conhecida como libc;
- biblioteca matemática padrão, a libm;
- a interface Java para chamada de códigos nativos chamada de JNI;
- libz, biblioteca de compressão ZLib;
- liblog, usado para enviar mensagens de log ao núcleo.

3.6 Considerações Finais

Neste capítulo foram apresentadas as principais plataformas para o desenvolvimento de aplicação para dispositivos móveis com ênfase na plataforma Android, considerando seus aspectos mais relevantes, tais como a composição da arquitetura da plataforma, as características de cada camada, o ambiente de desenvolvimento e a configuração necessária para se desenvolver aplicativos Android e a ferramenta NDK que oferece a habilidade de invocar códigos nativos como os escritos em C e C++ a partir de aplicações Android (ANDROID, 2009).

As principais características da linguagem de programação Java, usada para o desenvolvimento de aplicação na plataforma Android, também foram descritas.

A Tabela 3.1 apresenta uma comparação entre as plataformas de desenvolvimento para dispositivos móveis consideradas neste estudo. Pode-se observar que as características da plataforma Android a tornaram a opção mais interessante para este trabalho, pois além do mercado estar em ascensão, possui código aberto e o ambiente de desenvolvimento não tem custo.

Tabela 3.1. Tabela comparativa entre as plataformas de desenvolvimento para dispositivos móveis.

	Linguagem de Programação	Código aberto	Ambiente de Desenvolvimento	Mercado Atual
Symbian	C/C++	Sim	Sem custo	Em declínio
Java ME	Java	Não	Sem custo	Em declínio
Microsoft	VB, C#	Não	Com custo	Estável
Android	Java	Sim	Sem custo	Em ascensão

CAPÍTULO 4. **Biometria em Dispositivos Móveis**

A crescente popularização dos dispositivos móveis somada ao aumento dos dados armazenados nestes dispositivos em função da crescente oferta de serviços e aplicações levanta uma importante questão: a segurança dos dados pessoais (IJIRI; SAKURAGI; LAO, 2006). Uma das possibilidades para aumentar a segurança em dispositivos móveis é o uso de Biometria.

Este capítulo apresenta uma revisão bibliográfica sobre Biometria em dispositivos móveis enfatizando a técnica de reconhecimento facial.

4.1 Introdução

Segundo FONG & SENG (2009), as maiores questões relacionadas à segurança de dispositivos móveis incluem:

- devido ao seu pequeno tamanho, os dispositivos móveis são roubados ou perdidos com frequência;
- os dados armazenados nos dispositivos não são criptografados ou quando o são, o protocolo adotado é falho;
- devido à conectividade sem fio, comumente presente nestes dispositivos, eles são propensos a ataques e a vírus;
- é comum a autenticação de usuários estar desabilitada, ser fraca ou usar o método comum e fraco de autenticação baseada em senhas simples e estáticas.

Para proteger o acesso ao dispositivo, o que inclui os dados nele armazenados e serviços neles disponíveis, a autenticação dos usuários se mostra um método eficiente de proteção. Nesse sentido, destacam-se três tipos principais de autenticação de usuário, que podem ser usados sozinhos ou de modo combinado: posse (cartões), conhecimento (senhas) e biometria (FONG; SENG, 2009).

As senhas, tradicionalmente usadas para autenticação de usuários de dispositivos móveis, são fáceis de implementar, porém constituem um método fraco e insuficiente, pois podem ser quebradas (FONG; SENG, 2009). O método baseado em senhas apresenta também a inconveniência de exigir a digitação em um teclado de tamanho reduzido.

Os métodos baseados em posse não são adequados para uso com dispositivos móveis, pois exigem que o usuário mantenha sempre algum tipo de *token* junto ao dispositivo (FONG; SENG, 2009).

Portanto, segundo FONG & SENG (2009), a Biometria se mostra como o método mais promissor, conveniente e eficiente para autenticação de usuários em dispositivos móveis.

4.2 Reconhecimento Facial em Dispositivos Móveis

Para FONG & SENG (2009) as biometrias mais apropriadas para dispositivos móveis incluem face e impressão digital devido à boa qualificação destas técnicas e pela não necessidade de hardware adicional.

Segundo IJIRI, SAKURAGI & LAO (2006), o reconhecimento facial é a técnica mais útil para autenticação biométrica em dispositivos móveis pelo fato da maioria destes dispositivos já estarem equipados com pelo menos uma câmera, o que é o suficiente para a implementação desta técnica. Para os fabricantes, isto representa uma vantagem, pois podem oferecer dispositivos aptos a sistemas biométricos sem nenhum custo adicional com hardware específico. Neste sentido, o reconhecimento facial se mostra o mais eficiente método biométrico para dispositivos móveis uma vez que concilia segurança, usabilidade e baixo custo.

Porém, os dispositivos móveis apresentam desafios específicos para a detecção e reconhecimento de face especialmente em função da sua característica de mobilidade, o que acarreta alta variabilidade nas condições do ambiente, como por exemplo, na iluminação das imagens capturadas (HADID et al., 2007).

Também deve-se levar em consideração as características do dispositivo que, embora auxiliem no processo uma vez que a participação do usuário é colaborativa e a distância da captura da face é pequena, apresenta restrições de hardware destacando a limitada quantidade de memória, a capacidade reduzida de processamento e a comum ausência do coprocessador matemático nestes tipos de dispositivos (HADID et al., 2007).

4.3 Trabalhos Correlatos

Nesta seção são apresentados alguns trabalhos relacionados ao tema desta

dissertação de mestrado, com destaque para as técnicas e as tecnologias adotadas, bem como os resultados reportados.

4.3.1 Métodos para Detecção de Faces e Olhos

HADID et al. (2007) implementaram e compararam dois métodos para detecção de faces em dispositivos móveis considerados “estado da arte”. O primeiro método é baseado na detecção de pele com análise baseada em cor, enquanto o segundo método é baseado nas características de Haar (seção 2.3.1).

Esses pesquisadores concluíram que o método baseado em cores é mais simples e rápido, porém mais sensível as alterações de iluminação e do ambiente. Por outro lado, o método baseado nas características de Haar é mais lento, porém mais eficiente. A Tabela 4.1 apresenta às diferenças identificadas entre esses dois métodos. Ela mostra que o método baseado em cores foi executado a 8 quadros por segundo e detectou 117 faces de um total de 163, com 12 falsos positivos. O método baseado nas características de Haar, por sua vez, foi executado a 2 quadros por segundo e detectou 129 faces de um total de 163, com apenas 2 falsos positivos.

Tabela 4.1. Resultado comparativo da detecção de faces usando as técnicas baseada em cor e nas características de Haar (HADID et. al., 2007).

Técnica	Detectados	Falsos positivos	Desempenho
Cor da pele	117	12	8 frames por segundo
Haar + AdaBoost	129	2	2 frames por segundo

Para a realização deste estudo, HADID et al. (2007) usaram 150 imagens de teste contendo 163 faces. O dispositivo utilizado foi o celular Nokia N90, ilustrado na Figura 4.1, equipado com uma câmera rotativa, sistema operacional Symbian versão 8.1a, processador ARM9 com 220 Mhz e memória interna de 31 MB.

Figura 4.1. Celular Nokia N90 (HADID et al., 2007).

4.3.2 Arquitetura Cliente Servidor para Reconhecimento de Faces proposta por Kumar

KUMAR et al. (2010) propuseram uma arquitetura cliente/servidor onde a detecção de face é feita no dispositivo móvel (lado cliente) e o reconhecimento facial é realizado em um computador dedicado (lado servidor).

Na fase de segmentação ou detecção de face o algoritmo adotado foi o Viola-Jones que apresentou bons resultados com faces frontais. No lado servidor, onde acontece a fase de reconhecimento da face, foi usado um algoritmo baseado na técnica PCA (*eigenfaces*).

Este tipo de arquitetura pode ser implementada tanto usando Bluetooth quanto usando o protocolo HTTP, com ambos apresentando vantagens e desvantagens.

Na proposta de KUMAR et al. (2010) foi adotada a solução com Bluetooth para promover transferências de dados do dispositivo móvel para o servidor e vice-versa.

No lado cliente, para promover interação com o dispositivo, foi adotada a API (*Application Program Interface*) JME com o perfil MIDP 2.0, além da API Mobile Media para acesso à câmera do dispositivo.

O processo no lado cliente ocorre da seguinte forma:

- uma imagem é capturada pela câmera do dispositivo;
- por meio do algoritmo Viola-Jones (seção 2.3.1) é detectada a face, sendo a região de interesse destacada e convertida em bytes;
- os bytes são enviados para o servidor via conexão Bluetooth por meio da API Bluecove para uso na fase de reconhecimento.

No lado servidor, a solução foi desenvolvida usando as linguagens C++ e Java. O processo no servidor ocorre da seguinte forma:

- os bytes enviados pelo cliente são recebidos por um *javabean* usando a API bluecove e convertidos em uma imagem jpg;
- então é criada uma entrada no arquivo xml de treinamento ou no arquivo xml de teste;
- como o algoritmo escolhido para esta fase é baseado em *eigenfaces*, deve-se realizar um treinamento o que é feito por meio do arquivo xml de treinamento;
- o passo seguinte é executar o algoritmo *eigenface* para reconhecer a face (seção 2.3.2);
- a resposta então é enviada ao lado cliente via conexão Bluetooth.

A Figura 4.2 ilustra o funcionamento da solução proposta por KUMAR et al. (2010).

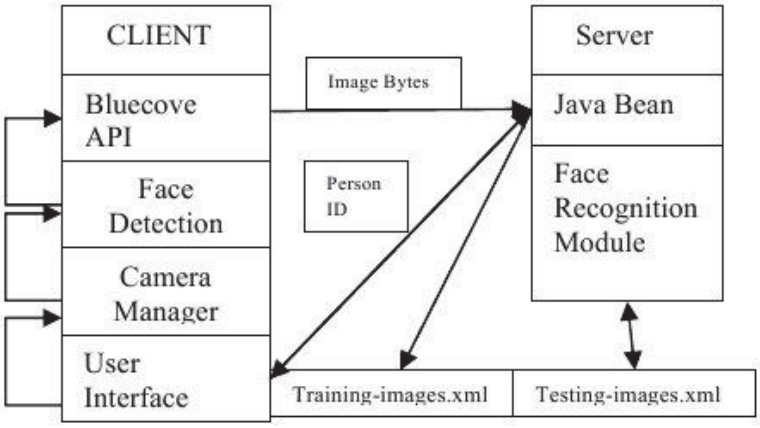

Figura 4.2. Arquitetura Cliente-Servidor usando Bluetooth (KUMAR et al. 2010).

A avaliação dos resultados desta proposta considerou o seguinte ambiente de hardware e software:

- Lado cliente: smartphone Sony Ericsson w550i;
- Lado servidor: Dell Inspiron 1520 Intel Pentium core 2 duo, 2.1 Ghz, 1GB de memória RAM, sistema operacional Windows Vista e Bluetooth Dell TrueMobile 355.

A Tabela 4.2 exibe uma análise de desempenho no que tange ao tempo necessário para a execução do processo. Os valores representam o intervalo médio de tempo em milissegundos (ms).

Tabela 4.2. Intervalos médio de tempo para cada processo na arquitetura cliente-servidor com Bluetooth (KUMAR et al., 2010).

Etapa	Tempo (ms)
Captura da imagem	34
Detecção da Face	19142
Estabelecer conexão com o servidor	566
Transferência dos bytes da imagem do cliente ao servidor	320
Escrever no arquivo XML	85
Reconhecimento facial	18451
Enviar resposta ao cliente e exibi-la	642
Total	39240

4.3.3 Arquitetura Cliente Servidor para Reconhecimento de Faces proposta por Yu

YU (2010) desenvolveu uma aplicação de reconhecimento facial em *smartphones* para desbloquear o aparelho quando o usuário registrado for reconhecido.

Na fase de segmentação ou detecção de face foi usado o método de segmentação de cor da pele e o algoritmo *eigenface* (seção 2.3.2) na fase de reconhecimento facial.

O processo básico consiste da captura da imagem no dispositivo móvel, a detecção da face, o envio da face detectada ao servidor/base de dados, a comparação da face com imagens de treinamento e o retorno do resultado pelo servidor ao dispositivo. A arquitetura proposta é ilustrada na Figura 4.3.

Figura 4.3. Arquitetura do sistema proposto (YU, 2010).

A solução proposta apresenta três etapas básicas com suas subetapas. A primeira é a aquisição da imagem, a segunda é a detecção da face e a terceira é o reconhecimento facial.

A primeira trata basicamente da captura da imagem, o ponto inicial do processo.

A etapa de detecção da face consiste no processo de redução da resolução para diminuir a carga computacional, no processo de segmentação pela cor da pele para localizar pixels de pele e, finalmente, é feito um tratamento na imagem resultante para eliminar imperfeições.

A etapa final do processo consiste no reconhecimento facial usando o algoritmo *eigenface* (seção 2.3.2). Nesta etapa, o primeiro passo é treinar o algoritmo considerando o conjunto de imagens para treinamento. A seguir são calculados os coeficientes da projeção da face no *eigenspace*, utilizados finalmente para identificar o indivíduo.

Para avaliar o desempenho da solução proposta, YU (2010) utilizou 50 imagens capturadas de 5 pessoas diferentes sendo 10 imagens por pessoa. Ele considerou a distância Euclidiana entre a imagem de teste e as de treinamento e a classificou por meio de um limiar estabelecido.

Foi utilizada a linguagem Matlab para a implementação dos algoritmos de detecção e reconhecimento facial. Uma aplicação no dispositivo móvel captura as imagens e as envia a uma servlet em um servidor Dell Inspiron 6400. Após o processamento da imagem o resultado é tornado ao dispositivo onde é exibido.

YU (2010) reporta os resultados obtidos nos experimentos utilizando os seguintes parâmetros:

- Falsa rejeição (FR): Quando a face de teste é genuína, porém a distância excede o limiar estabelecido;
- Falsa aceitação (FA): Quando a face de teste é impostora, porém a distância é menor do que o limiar estabelecido;
- Correta rejeição (CR): Quando a face é impostora e a distância excede o limiar estabelecido;
- Correta aceitação (CA): Quando a face é genuína e a distância é menor do que o limiar estabelecido.

Com um limiar igual a 125, um total de 40 imagens foram identificadas corretamente, como apresentado na Tabela 4.3.

Tabela 4.3. Resultados experimentais (YU, 2010).

CA	CR	FA	FR	Total correto	Taxa de acerto
27	13	3	7	40	80%

4.3.4 Arquitetura Cliente Servidor para Reconhecimento de Faces proposta por Pabbaraju

PABBARAJU & PUCHAKAYALA (2010) propuseram um sistema para reconhecimento facial em dispositivos móveis cuja arquitetura também adotou o modelo cliente/servidor. Eles mencionam que as restrições do poder computacional aliadas à limitada área de armazenamento dos dispositivos móveis são fatores limitantes para que o processo ocorra completamente no próprio dispositivo. Portanto, eles também propõem uma arquitetura cliente-servidor.

Na arquitetura proposta (Figura 4.4), o lado cliente captura a imagem e realiza o procedimento de detecção facial usando a segmentação pela cor da pele e extrai as características que descrevem a face. Após, estas características são enviadas para o servidor que realiza a tarefa de comparação com a base de dados, tarefa computacionalmente intensa. O resultado é, então, retornado ao lado cliente que o exibe ao usuário.

O sistema foi desenvolvido para dispositivos baseados na plataforma Android. Na fase de detecção de face foi adotada a biblioteca (API) Face Detector desta plataforma e o algoritmo Fisherfaces, que faz uso da técnica *Linear Discriminant Analysis* (LDA) (PABBARAJU & PUCHAKAYALA, 2010) na fase de reconhecimento.

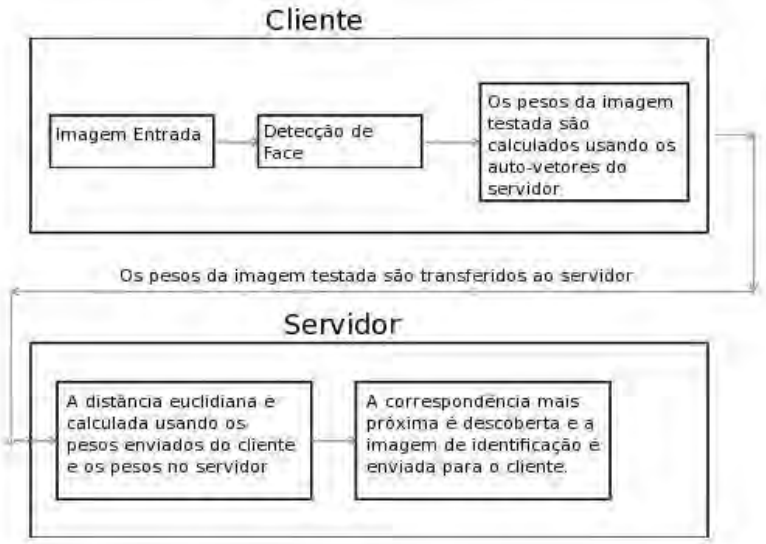

Figura 4.4. A arquitetura proposta por PABBARAJU & PUCHAKAYALA (2010).

Para avaliação da plataforma proposta, foram realizados experimentos utilizando-se duas bases de dados: *Labeled Faces in the Wild* (LFW) e a Essex University Database (EUD). A LFW contém em torno de 13.233 faces coletadas de fotos na web. Nesta base de dados 1680 pessoas possuem duas ou mais imagens distintas. A base de dados EUD possui 395 imagens de pessoas de ambos os sexos, sendo 20 imagens por indivíduos de diferentes raças.

No treinamento foram usadas 20 classes (pessoas) com 5 imagens de cada uma, totalizando 100 imagens oriundas das duas bases de dados.

Nos testes foram usadas 110 imagens de ambas as bases acima citadas. Para o cálculo da similaridade foi utilizada a distância euclidiana. O desempenho do sistema foi mensurado considerando quatro aspectos: a Taxa de Aceitação Correta (CAR), a Taxa de Rejeição Correta (CRR), a Taxa de Falsa Aceitação (FAR) e a Taxa de Falsa Rejeição (FRR).

Com um limiar em torno de 56, o sistema alcançou uma taxa de acerto de 91%. A taxa de erro igual (EER) alcançada foi de 6%, o que significa que o sistema apresenta um bom desempenho.

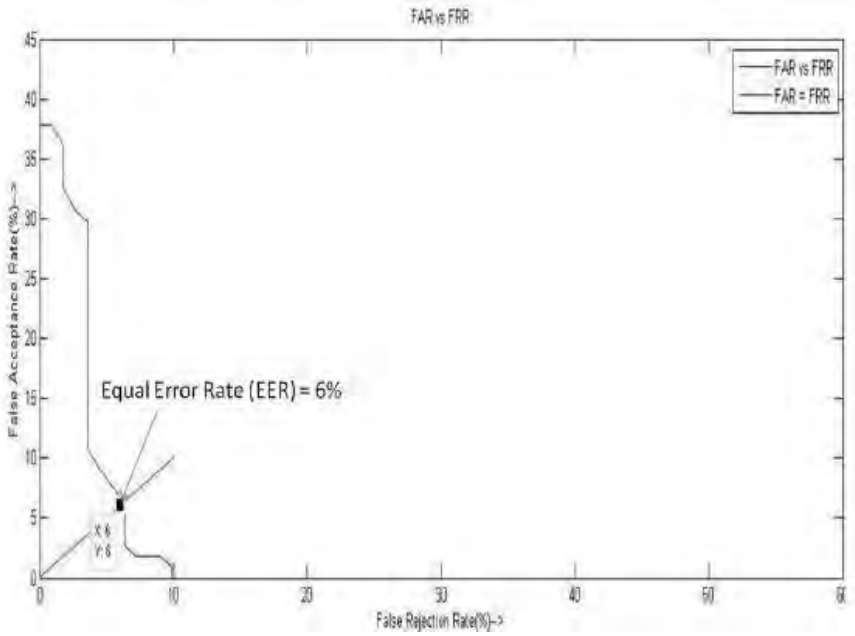

Figura 4.5. Comparação entre as taxas FAR e FRR (PABBARAJU; PUCHAKAYALA, 2010).

4.4 Considerações Finais

Neste capítulo foram apresentadas as principais questões relacionadas aos dispositivos móveis. Nota-se a necessidade de um modo eficiente de proteção dos dados armazenados nestes tipos de dispositivos devido a sua crescente popularização e consequente aumento nos dados neles armazenados. Nesse sentido a biometria apresenta-se como um modo eficaz e conveniente de proteção por meio da autenticação do usuário e, considerando as características destes tipos de dispositivos, segundo FONG & SENG (2009) o reconhecimento facial é considerada a técnica biométrica mais promissora.

Foram apresentados também alguns trabalhos relacionados ao apresentado nesta dissertação. O primeiro trabalho comparou as técnicas de detecção de faces por segmentação da cor da pele com a técnica que faz uso das características de Haar com resultado favorável para o segundo método no quesito eficácia. Os demais trabalhos propuseram uma arquitetura cliente-servidor para o reconhecimento facial em dispositivos móveis. Enquanto a proposta de Kumar procurou medir o desempenho, Yu e Pabbaraju se dedicaram a medir a eficácia do sistema apresentando suas taxas de acerto.

A Tabela 4.4 mostra uma comparação entre as principais características destes

trabalhos.

Tabela 4.4. Comparação entre as propostas dos trabalhos correlatos apresentados.

	Arquitetura	Detecção de Faces	Reconhecimento facial	Tempo médio	Taxa Acerto
Kumar	Cliente-servidor	Viola-Jones	PCA	39240 ms	-
Yu	Cliente-servidor	Cor da pele	PCA	-	80%
Pabbaraju	Cliente-servidor	Cor da pele	PCA	-	91%

CAPÍTULO 5. **BioMobile**

Uma das principais contribuições deste trabalho foi o desenvolvimento do BioMobile, sistema biométrico baseado em reconhecimento facial projetado para dispositivos móveis equipados com a plataforma Android versão 2.3.3 ou superior como, por exemplo, *tablets* e *smartphones*.

Este capítulo apresenta detalhes do projeto e da implementação do BioMobile, bem como os seus modos de parametrização, que permitem realizar medições de desempenho de diferente técnicas sendo executadas sob limitada configuração de hardware.

5.1 Arquitetura do BioMobile

O sistema BioMobile foi desenvolvido para ser executado sobre a plataforma Android e, ao contrário dos sistemas descritos na seção 4.3, é baseado em uma arquitetura na qual a aplicação é auto-suficiente, ou seja, ela não depende de nenhum servidor ou de nenhuma outra aplicação para realizar sua tarefa o que significa que todas as etapas do sistema, da captura da imagem até a identificação do indivíduo, são realizadas no próprio dispositivo.

A vantagem dessa arquitetura está na conveniência proporcionada por esta auto-suficiência, especialmente para aplicações de reconhecimento facial que usam o modo de identificação de faces. Considere uma aplicação que usa a imagem de uma determinada pessoa, cadastrada na base de dados de contatos, para identificar na galeria de imagens do dispositivo as imagens relacionadas com essa pessoa. Quão inconveniente seria se a cada novo contato cadastrado ou imagem adicionada ao dispositivo fosse necessário se conectar a um servidor a fim de realizar o processamento biométrico. Um sistema auto-suficiente torna mais convenientes aplicações desta natureza. Além disso, há sempre o risco do recurso externo não estar disponível no momento em que é necessário realizar a identificação biométrica do usuário do dispositivo.

A fim de permitir uma comparação entre diferentes técnicas de reconhecimento facial

foram avaliadas duas técnica de reconhecimento de faces, uma baseada em PCA e outra baseada em LBP.

Quanto a funcionalidades o BioMobile permite o cadastramento, a identificação e a autenticação de indivíduos.

Outro diferencial do BioMobile é que o reconhecimento facial é baseado em vídeo, com isso a identificação do usuário pode ser realizada utilizando-se conjuntos de imagens ao invés de uma única imagem estática, o que permite maior tolerância à falhas na fase de detecção de face e melhor desempenho na fase de reconhecimento por meio da técnica de maioria de votos.

A arquitetura do sistema BioMobile, ilustrada na Figura 5.1, é composta pelos módulos de aquisição de vídeo, extração de quadro, detecção de face, pré-processamento, extração das características da face e reconhecimento do usuário.

No módulo de aquisição de vídeo são gerados vídeos de no máximo 5 segundos usando a resolução padrão do dispositivo e com a configuração de 20 quadros por segundos dos quais são extraídos os quadros (*frames*) usados no processamento.

Na fase de extração de quadro são extraídos quadros do vídeo por meio da classe MediaMetadataRetriever da biblioteca padrão do Android. Na tentativa de obter uma maior variação das condições de captura (pose, iluminação, etc.) são selecionados um quadro do início, um da metade e um do final do vídeo.

Após a extração do quadro é realizada a detecção de faces usando o algoritmo Viola-Jones, implementado na biblioteca OpenCV. O algoritmo faz uso de um arquivo de treinamento de faces frontais que acompanha a biblioteca OpenCV.

Caso uma face seja detectada na imagem esta é submetida a um pré-processamento. Isto é necessário para ajustar a imagem para um padrão pré-concebido pelo sistema antes que ela seja enviada à etapa de extração das características da face. O Biomobile trabalha com imagens em tons de cinza com dimensões de 100x100 pixels.

No módulo de extração das características foram avaliados dois algoritmos: um baseado em autofaces e outro baseado no operador LBP. O algoritmo baseado em autofaces é o PCA cuja implementação usada foi a da biblioteca Intel OpenCV. O algoritmo baseado no operador LBP é mais recente e foi usada uma implementação própria feita na linguagem Java.

O último módulo do sistema é o reconhecimento do usuário. Nele o descritor de uma face amostrada é comparado com descritores de faces previamente cadastradas e o melhor resultado desta comparação é avaliado por um limiar a fim de determinar a autenticidade de uma pessoa. O método usado nesta comparação é a medida da distância dos descritores das imagens que é calculada pela técnica conhecida por *Chi Square*, cujo algoritmo foi implementado no próprio sistema usando a linguagem Java.

5.2 Modos de Operação do BioMobile

A fim de melhor analisar o comportamento do sistema proposto no ambiente para o qual foi projetado, foram definidos alguns modos de operação com várias opções de configurações.

Além do cadastro de pessoas o sistema também pode executar nos modos de identificação ou autenticação.

No cadastro de pessoas, após a etapa de extração das características, é criado um registro em uma tabela do banco de dados SQLite onde são armazenadas as informações da pessoa cadastrada. O SQLite é composto de uma biblioteca que vem embarcada na plataforma Android e provê funcionalidades de banco de dados a aplicações Android por meio de instruções SQL.

No modo de autenticação, após a extração das características da face, é realizada uma consulta no banco de dados para obter as informações cadastrais da pessoa sendo autenticada. Então, é realizada uma comparação entre as características da face amostrada e as características da pessoa cadastrada. Como o BioMobile utiliza vários quadros capturados de vídeos, ao invés de utilizar apenas uma imagem estática, no modo de autenticação foi adotada a técnica de “maioria de votos” na qual são realizadas várias comparações vencendo a identidade mais votada. Os vídeos possuem 20 fps (quadros por segundo) e na versão corrente do sistema, foi adotada a configuração de maioria dentre 3 votos.

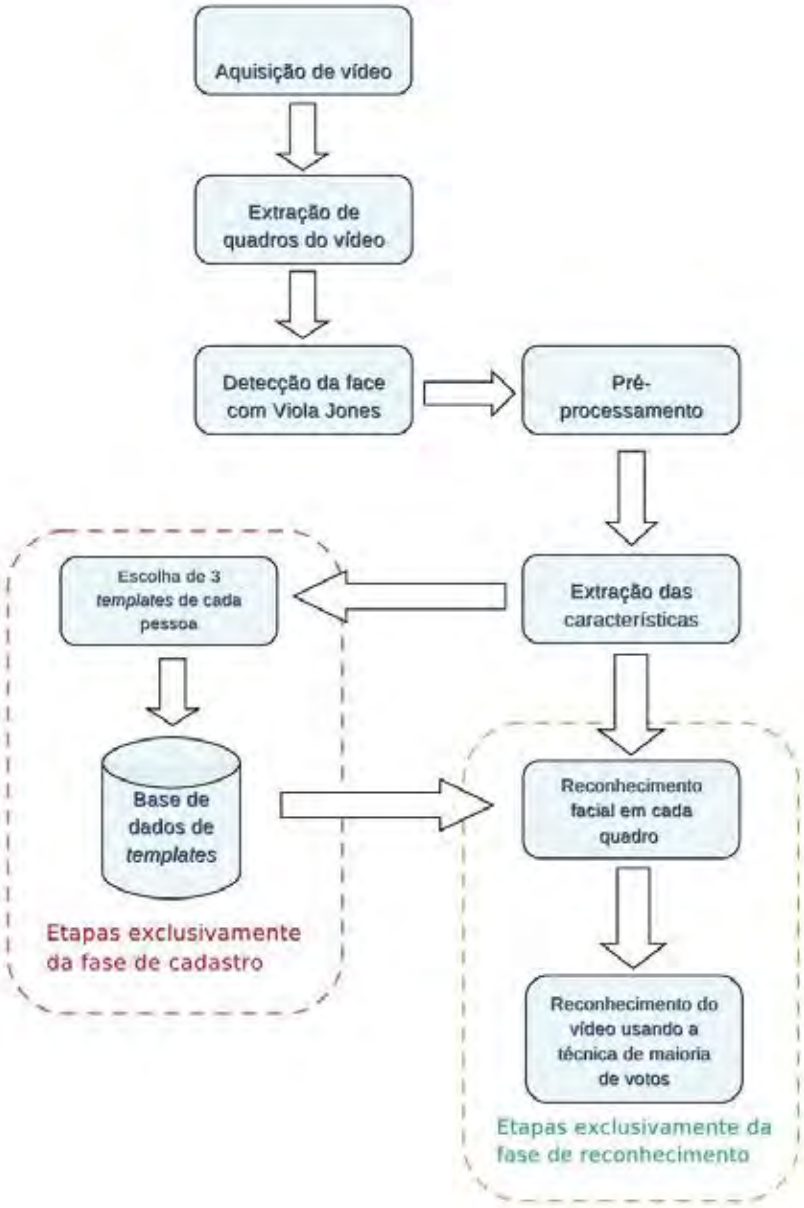

Figura 5.1 Arquitetura do BioMobile.

Quando operando no modo identificação, as características da face amostrada são comparadas com as características de todas as faces cadastradas no banco de dados do sistema, sendo eleita a identidade da pessoa cadastrada no banco de dados cujas características da face apresentam a menor distância durante o processo de comparação. O menor valor de distância é também comparado com um valor de limiar para se determinar se aquela identidade deve ser aceita ou não.

A Figura 5.2 mostra a tela inicial do sistema com os modos de operação, identificação e autenticação. Como pode ser observado, existe um campo onde o usuário pode definir o

valor de limiar e, no rodapé da tela há informações sobre a configuração atual do sistema.

Figura 5.2 Interface inicial do sistema BioMobile**.**

Na Figura 5.3 pode-se observar um diagrama com as telas do menu com os dois tipos de parâmetros: o tipo do operador LBP e o tipo de janelamento.

Na Figura 5.4 pode-se observar a dinâmica das operações do sistema, com os três tipos de operações suportadas pelo BioMobile.

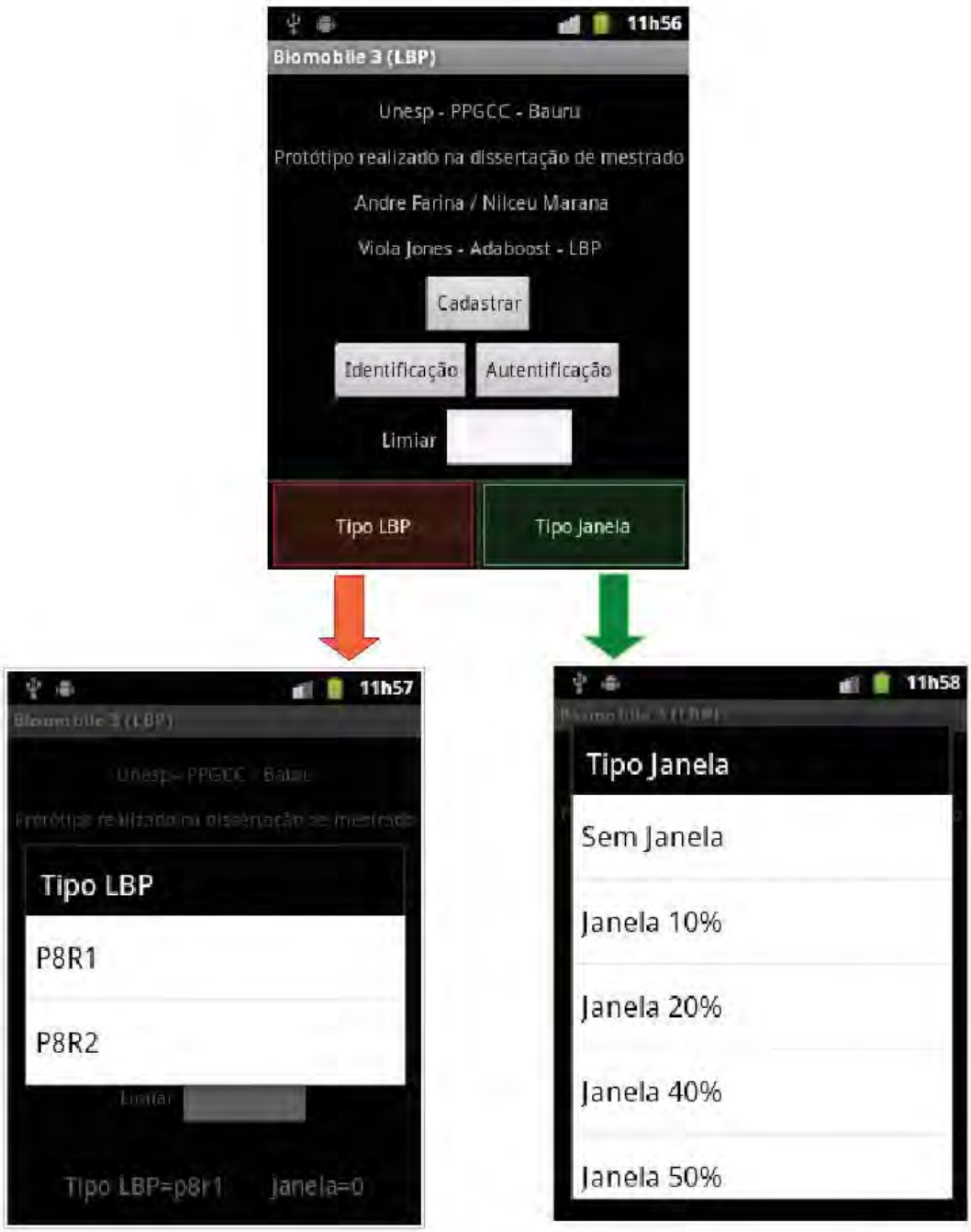

Figura 5.3. Dinânica do menu de opções de configurações do BioMobile**.**

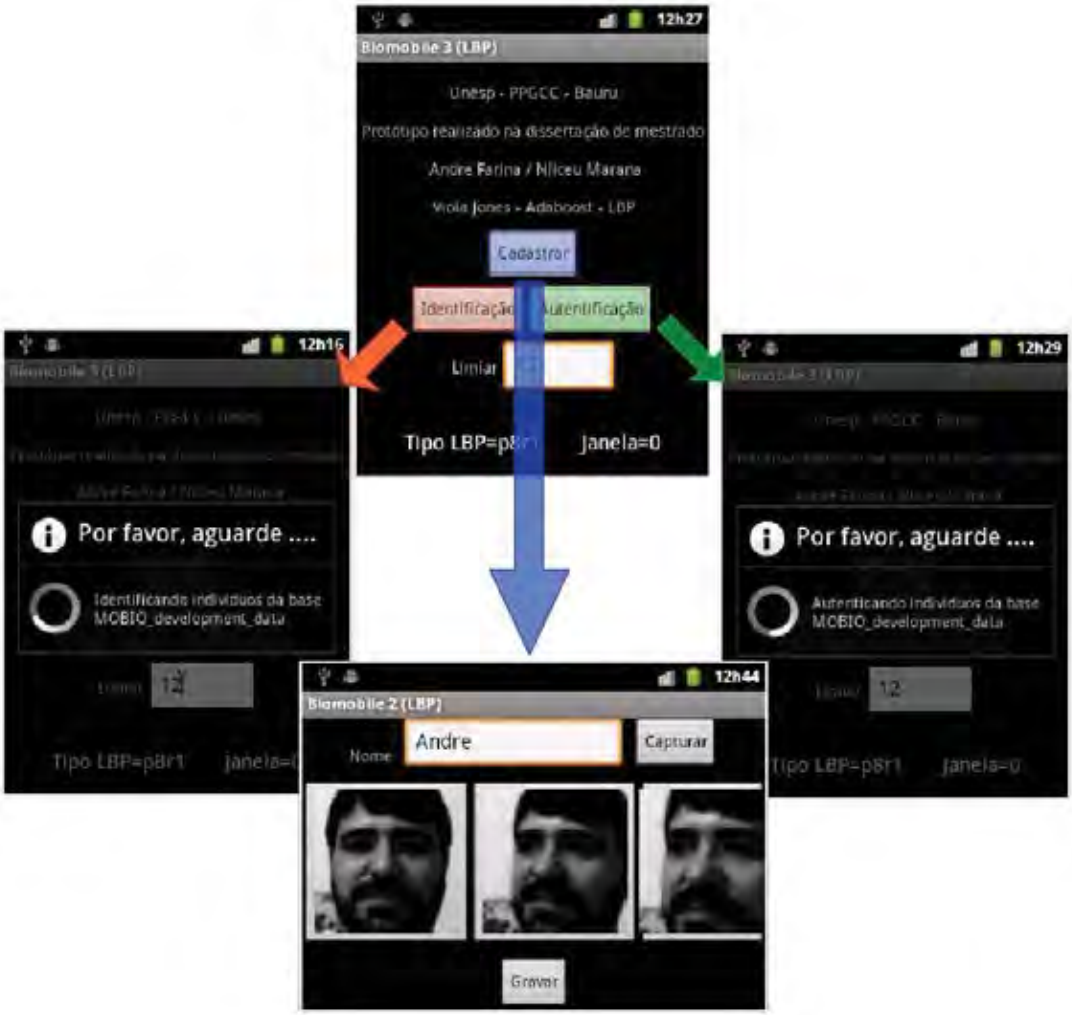

Figura 5.4. Diagrama das opções de operações do BioMobile**.**

CAPÍTULO 6. **Resultados Experimentais**

Neste capítulo são descritos os experimentos realizados visando a avaliação do desempenho do sistema BioMobile, bem como os resultados obtidos. Também é descrito o material utilizado durante os experimentos.

6.1 Dispositivos Móveis Utilizados

Os experimentos foram realizados em dois dispositivos: um *smartphone* Motorola Spice Key, equipado com o sistema operacional Android 2.3.3, câmera de 3.2 Mega Pixels, memória RAM de 256 MB, cartão SD de 2 GB e processador Arm de 600 MHz e um *Tablet* Samsung Galaxy Tab 10.1 equipado com o sistema operacional Android 3.1, câmera de 3.2 Mega Pixels, memória RAM de 1 GB, memória interna de 16 GB e processador Arm Dual Core de 1 GHz.

A Figura 6.1 apresenta imagens desses dispositivos móveis.

Figura 6.1. Imagem dos dispositivos móveis utilizados nos experimentos.

6.2 Base de Dados MoBio

Para análise de desempenho do sistema BioMobile foi adotada uma base de dados multimodal composta de vídeos contendo faces e áudio (voz), chamada MoBio, um acrônimo em inglês para Mobile Biometry, disponibilizada pelo instituto de pesquisa Idiap no endereço http://www.idiap.ch/dataset/mobio (MCCOOL; MARCEL, 2009).

A base de dados MoBio foi criada visando refletir potenciais cenários do mundo real para autenticação de face e fala em dispositivos móveis. A coleta dos dados foi realizada usando telefones celulares em seis lugares diferentes, em cinco países diferentes.

Por considerar também a possibilidade de autenticação por reconhecimento de voz, cada participante respondeu um total de 21 questões compostas de perguntas e respostas, leitura de texto e fala livre. Essas amostras foram divididas em 6 sessões onde há pouquíssima variação de pose e condição de iluminação. Os dados amostrados foram divididos em 3 conjuntos distintos: um para treinamento, um para desenvolvimento e um para testes, de acordo com o tipo de fala. Esta divisão tem o propósito de apoiar o desenvolvimento de sistema de reconhecimento de voz (MCCOOL; MARCEL, 2009).

Nos experimentos realizados neste trabalho foi usado o conjunto de desenvolvimento composto por 47 indivíduos de ambos os gêneros. Como o reconhecimento de voz está fora do escopo deste trabalho, foi considerada apenas uma amostra de cada sessão ao invés das 21 amostras disponíveis. Esta seleção foi necessária para reduzir o custo operacional do sistema e como a variação de iluminação e pose é mínima, não houve perda significativa na diversidade do conjunto de dados.

Na Figura 6.2 são apresentadas amostras de imagens de indivíduos da base de dados MoBio. Tais amostras são de 4 indivíduos, sendo 2 do sexo masculino e 2 do sexo feminino, obtidas em quatro sessões distintas.

Figura 6.2. Exemplos de registros da base de dados MoBio.

6.3 Experimentos com Descritores Faciais Baseados em PCA

Dada a complexidade computacional e, consequentemente, a grande demanda pelos recursos de processamento e armazenamento de dados, a avaliação da versão do sistema BioMobile utilizando a técnica PCA não foi possível de ser realizada com toda a base de dados disponível. Para que fosse possível executar o BioMobile nos dispositivos móveis descritos na seção 6.1, com o algoritmo PCA disponibilizado na biblioteca OpenCV, foi necessário reduzir a base de vídeos de 47 indivíduos com 6 sessões cada para 6 indivíduos com apenas 1 sessão cada e para 11 indivíduos com apenas uma sessão para o *smartphone* e o *tablet,* respectivamente. Com estas reduções na base de dados foi possível avaliar o desempenho do sistema com a técnica PCA.

Nos trabalhos correlatos apresentados no Capítulo 5, a técnica PCA também foi utilizada, porém os sistemas foram implementados sob a arquitetura cliente-servidor e o processamento computacionalmente mais caro era realizado no servidor.

6.4 Experimentos com Descritores Faciais Baseados em LBP

Se a versão do BioMobile baseada na técnica PCA não pode ser executada utilizando-se toda a base de dados por exigir uma quantidade de recursos de hardware bem maior do que os dispositivos móveis são equipados, a versão com LBP mostrou-se bastante promissora. Em função de apresentar baixa complexidade computacional, foi possível realizar vários experimentos, com diferentes parâmetros de configuração do sistema.

Foram realizados experimentos nos quais a imagem de entrada era considerada completamente ou nos quais a imagem era dividida em subimagens (janelas) não sobrepostas. Nos experimentos com janelas, adotou-se tamanhos 20x20 pixels. Com relação ao raio de vizinhança do operador LBP foram considerados os raios 1 e 2, ambos com relação de adjacência data pela vizinhança-8. Na notação LBP (ver seção 2.3.3) as configurações de vizinhança e raio adotadas são P8R1 e P8R2.

Os experimentos foram realizados no modo de operação de identificação, usando a base de dados MoBio, descrita na seção 6.2. Foi usado o conjunto de desenvolvimento, chamado de MOBIO_development_data.

Esta gama de opções de configurações e condições exigiram a realização de vários experimentos, que estão descritos nas próximas subseções.

6.4.1 Configuração P8R1 sem Janelas

Na configuração P8R1, sem janelas, no modo identificação e usando um limiar ajustado em torno de 30 pontos, a taxa de erro igual (EER) apresentada pelo BioMobile foi de 15,5%, enquanto que os tempos médios de processamento foram de 1,8 segundos e 6,4 segundos em cada comparação, para o *tablet* e o *smartphone*, respectivamente, como apresentado na Tabela 6.1.

Tabela 6.1. Resultados obtidos na configuração P8R1 e sem janelamento.

Resultados		
Taxa de erro igual (EER)	Tempo médio de processamento	
15,5 %	*Tablet*	*Smartphone*
	1,8 segundos	6,4 segundos

As Figuras 6.3 e 6.4 apresentam, respectivamente, as curvas de distribuição da pontuação das comparações impostoras e genuínas, bem como as taxas de falsa aceitação (FAR) e falsa rejeição (FRR).

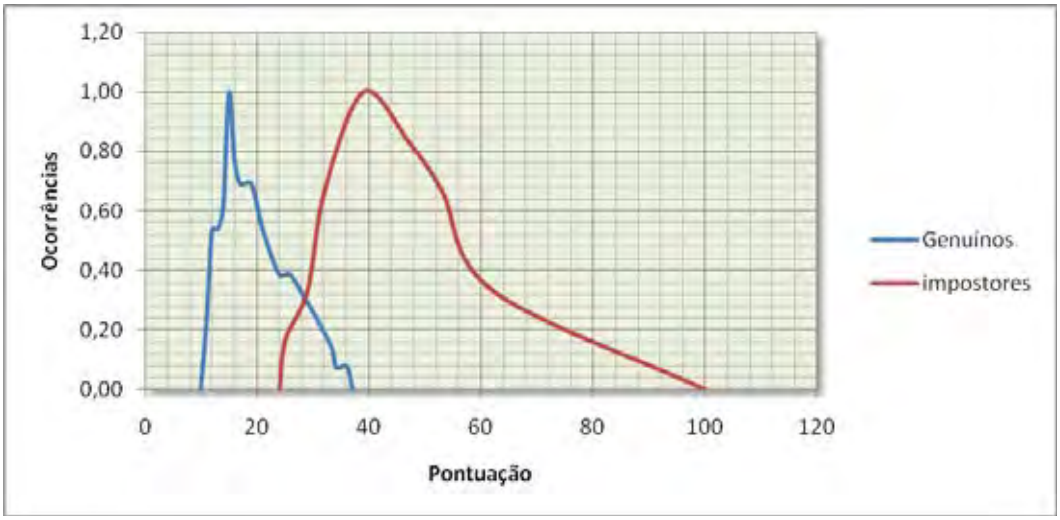

Figura 6.3. Distribuições das pontuações das comparações impostoras e genuínas, com P8R1 e sem janelas.

6.4.2 Configuração P8R1 com Janelas 20x20 pixels

Mantendo a vizinhança 8 e raio 1, porém alterando a configuração de janelas para 20x20 pixels, no modo identificação e usando um limiar ajustado em torno de 59 pontos, a taxa de erro igual caiu consideravelmente de 15,5% (sem o uso de janelas) para 2% (com o uso de janelas 20x20 pixels). Porém, como era de se esperar, o tempo de processamento também aumentou consideravelmente. A Tabela 6.2 apresenta a taxa de erro igual (EER) bem como os tempos médios de processamento para cada comparação realizada.

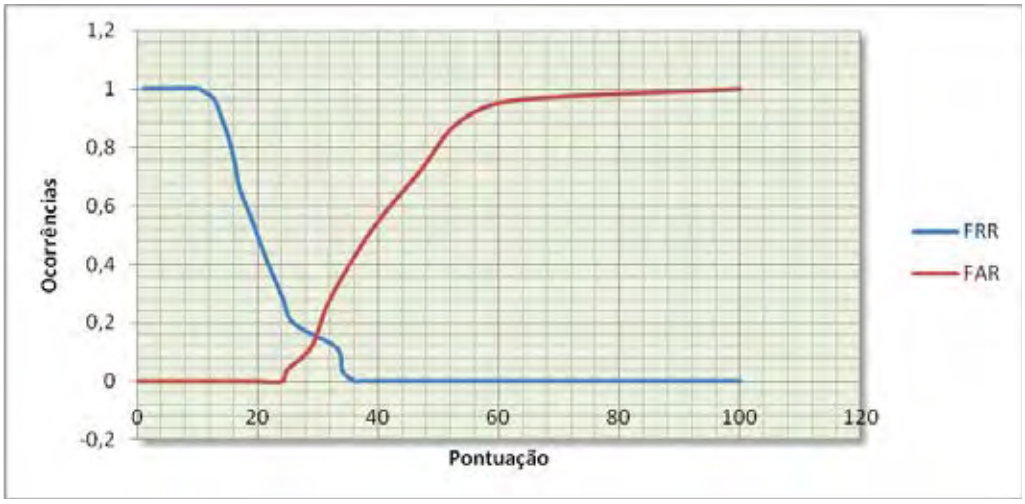

Figura 6.4. Taxas de falsa aceitação (FAR) e falsa rejeição (FRR), com P8R1 e sem janelas. Para o valor de pontuação em torno de 30, a taxa de erro igual (EER) foi de 15,5%.

Tabela 6.2. Resultados obtidos na configuração P8R1 e janelamento 20x20.

Resultados		
Taxa de erro igual (EER)	Tempo médio de processamento	
2 %	*Tablet*	*Smartphone*
	15,4 segundos	54,4 segundos

As Figuras 6.5 e 6.6 apresentam, respectivamente, as curvas de distribuição da pontuação das comparações impostoras e genuínas, bem como as curvas das taxas de falsa aceitação (FAR) e falsa rejeição (FRR), para cada valor de limiar.

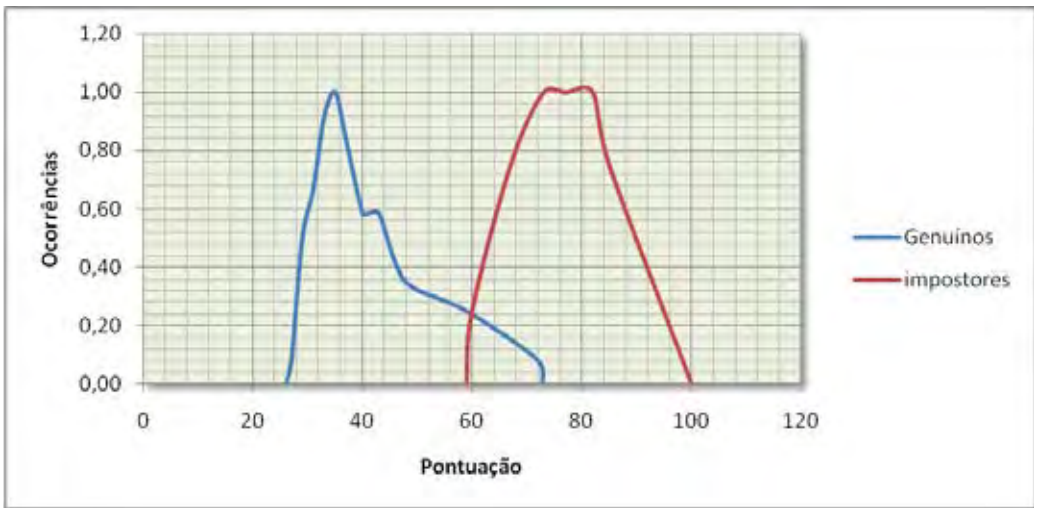

Figura 6.5. Distribuições das pontuações das comparações impostoras e genuínas, com P8R1 e janelas 20x20 pixels.

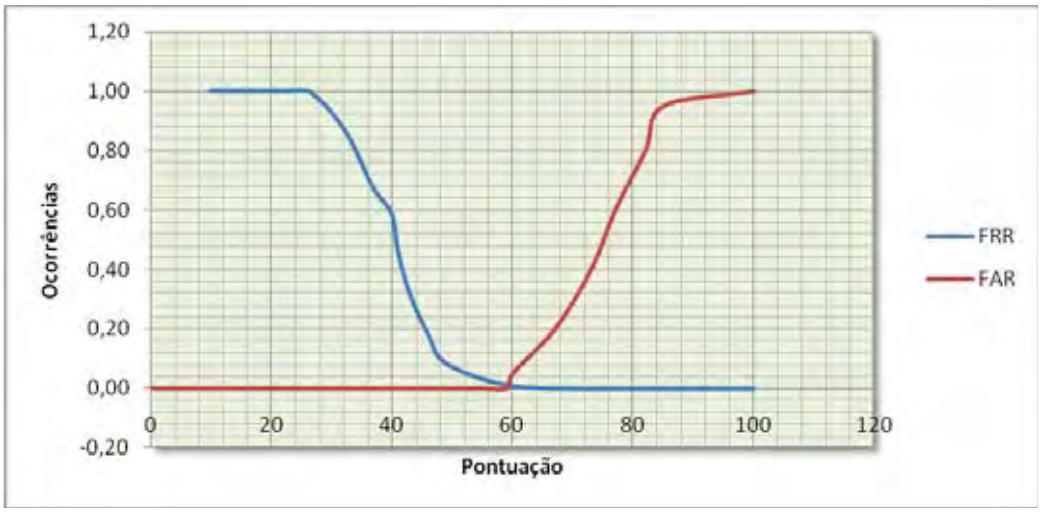

Figura 6.6. Curvas das taxas de falsa aceitação (FAR) e falsa rejeição (FRR), com P8R1 e janelas 20x20 pixels. Para o valor de pontuação em torno de 59, a taxa de erro igual (EER) foi de 2%.

6.4.3 Configuração P8R2 sem Janelas

Na configuração P8R2, sem janelas, no modo identificação e usando um limiar ajustado em torno de 43, a taxa de erro igual (EER) apresentada pelo BioMobile foi de 4,46%, enquanto que o tempo médio de processamento em cada comparação foi de 1,9 segundos no *tablet* e de 6,5 segundos no *smartphone*, como apresentado na Tabela 6.3.

Tabela 6.3. Resultados obtidos na configuração P8R2, sem janelas.

Resultados		
Taxa de erro igual (EER)	Tempo médio de processamento	
4,46 %	*Tablet*	*Smartphone*
	1,9 segundos	6,5 segundos

As Figura 6.7 e 6.8 apresentam, respectivamente, as curvas de distribuição da pontuação das comparações impostoras e genuínas, bem como as curvas das taxas de falsa aceitação (FAR) e falsa rejeição (FRR) para cada valor de limiar.

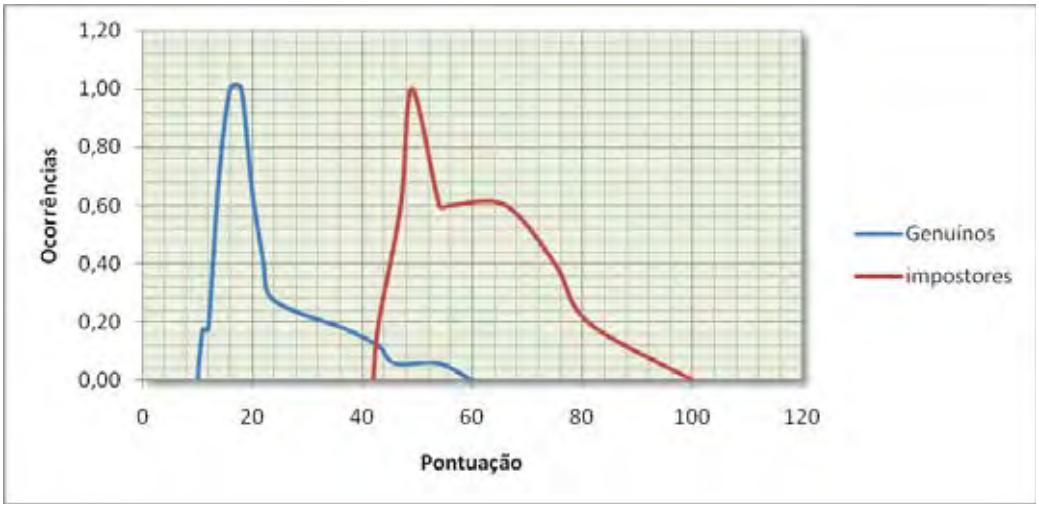

Figura 6.7. Distribuições das pontuações das comparações impostoras e genuínas, com P8R2 e sem janelas.

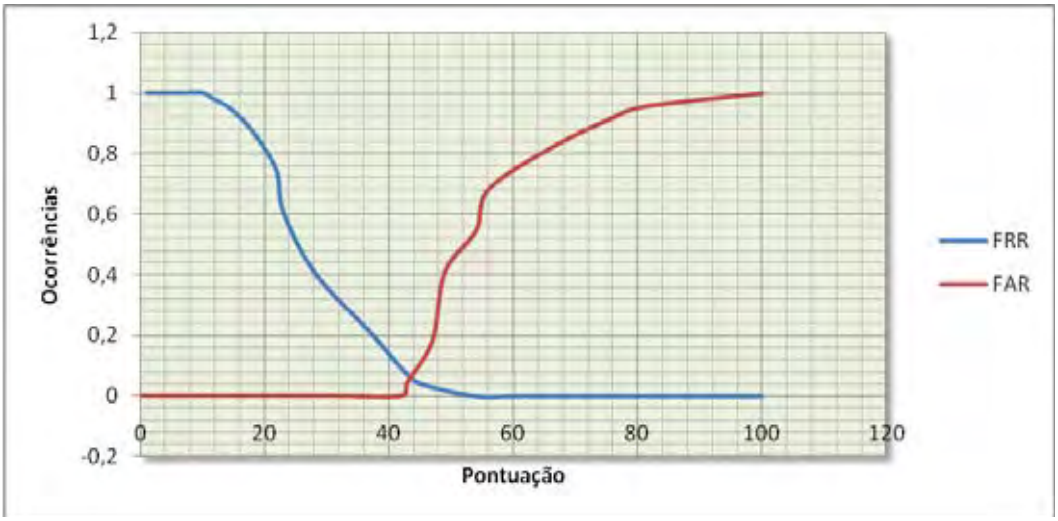

Figura 6.8. Curvas das taxas de falsa aceitação (FAR) e falsa rejeição (FRR), com P8R2 e sem janelas. Para o valor de pontuação em torno de 43, a taxa de erro igual (EER) foi de 4,46%.

6.4.4 Configuração P8R2 com Janelas 20x20 pixels

Na configuração P8R2, com janelas 20x20 pixels, no modo identificação e usando um limiar ajustado em torno de 69, a taxa de erro igual (EER) apresentada pelo BioMobile foi de 8%, enquanto que os tempos médios de processamento em cada comparação foram de 13,9 segundos e 43,9 segundos para o *tablet* e o *smartphone*, respectivamente. como apresentado na Tabela 6.4.

Tabela 6.4. Resultados obtidos na configuração P8R2 e janelas de 20X20 pixels.

Resultados		
Taxa de erro igual (EER)	Tempo médio de processamento	
8 %	*Tablet*	*Smartphone*
	13,9 segundos	43,9 segundos

As Figuras 6.9 e 6.10 apresentam, respectivamente, as curvas de distribuição da pontuação das comparações impostoras e genuínas, bem como as curvas das taxas de falsa aceitação (FAR) e falsa rejeição (FRR) para cada valor de limiar.

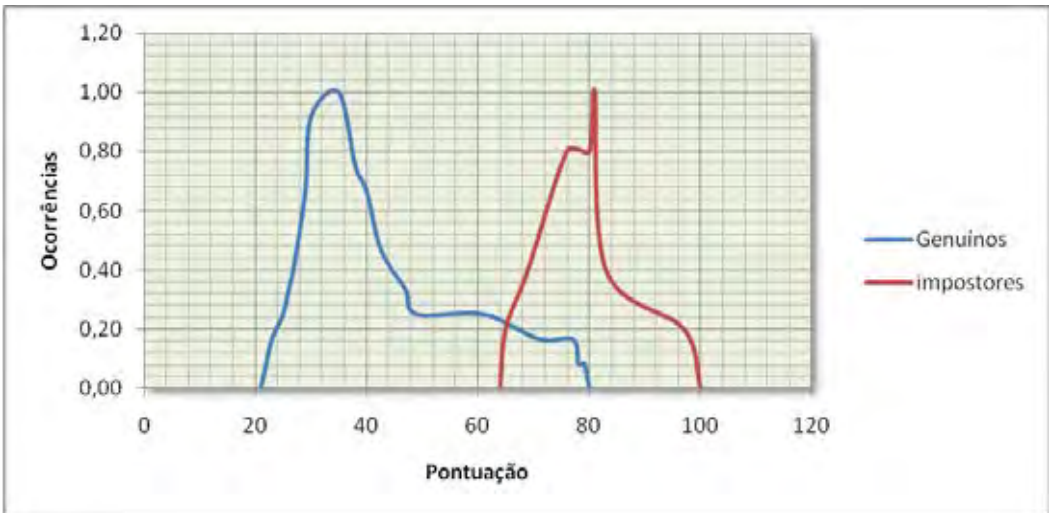

Figura 6.9. Distribuições das pontuações das comparações impostoras e genuínas, com P8R2 e janelas 20x20 pixels.

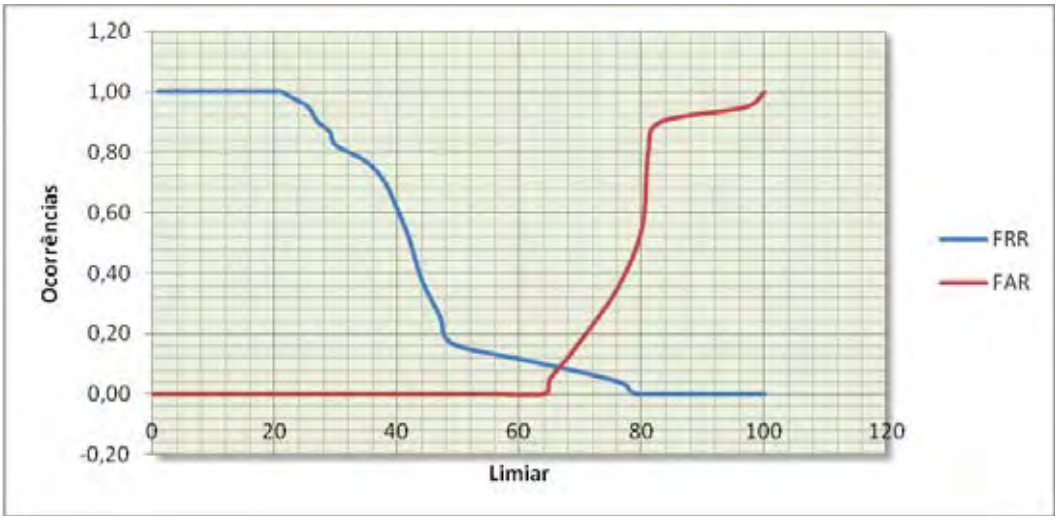

Figura 6.10. Curvas das taxas de falsa aceitação (FAR) e falsa rejeição (FRR), com P8R2 e janelas 20x20 pixels. Para o valor de pontuação em torno de 69, a taxa de erro igual (EER) foi de 8%.

6.5 Discussão

Devido às limitações atuais de hardware dos dispositivos móveis, foi observado experimentalmente que a técnica de descrição de faces baseada em PCA não é viável em sistemas de reconhecimento biométrico de usuários projetados para serem executados completamente no dispositivo móvel.

Nos testes usando os dispositivos descritos na seção 6.1, para que fosse possível executar uma bateria completa de testes, foi necessário reduzir a base de videos de 47 indivíduos com 6 sessões cada para 6 indivíduos (*smartphone*) ou 11 indivíduos (*tablet*) com apenas 1 sessão cada. Com a redução da base de dados para permitir a execução do programa não foi possível avaliar o desempenho do sistema com a técnica PCA. A maior dificuldade está na distribuição da pontuação genuína uma vez que como as sessões apresentam quase nenhuma variação de pose e iluminação, ao utilizarmos uma única sessão, as pontuações das comparações genuínas foram praticamente 0 em todas as comparações.

Por outro lado, a técnica de descrição de faces baseada em padrões binários locais (LBP) apresenta bons resultados se comparado aos resultados obtidos em outros trabalhos como os citados na seção 4.3, tanto com relação às taxas de reconhecimento, quanto com relação ao tempo de processamento. A Tabela 6.5 mostra os resultados obtidos pelo Biomobile nos diferentes modos de configuração. Observa-se que, como era de se esperar, o

desempenho do sistema BioMobile apresenta um tempo de processamento bem maior ao se utilizar a técnica baseada em LBP com janelas se comparado à técnica LBP sem janelas. Isso ocorre porque o custo computacional para realizar as operações em cada janela é muito alto uma vez que para cada janela se faz necessário obter um histograma que será concatenado em um histograma maior usado como descritor da face. Com relação às configurações do sistema, a configuração P8R1 sem janelas foi a que apresentou o pior desempenho no quesito eficácia, com uma taxa de erros em torno de 15,5%. Portanto, sua adoção não é indicada nem para o modo de operação de autenticação, nem para o modo de operação de identificação. A configuração P8R2 com janelas de 20x20 pixels apresentou um resultado bem melhor no quesito eficácia do que a configuração P8R1 sem janelas, porém apresentou um baixo desempenho no quesito tempo de processamento o que inviabiliza sua adoção, mesmo em operações de autenticação.

Tabela 6.5. Comparações dos resultados obtidos nas diferentes confgurações do Biomobile.

	P8R1 sem janela		P8R1 jan. 20x20		P8R2 sem janela		P8R2 jan. 20x20	
Tempo médio	*tablet*	*smartphone*	*tablet*	*smartphone*	*tablet*	*smartphone*	*tablet*	*smartphone*
	1,8 s	6,4 s	15,4 s	54,4 s	1,9 s	6,5 s	13,9 s	43,9 s
EER	15,5 %		2 %		4,46 %		8 %	

O melhor desempenho no quesito eficácia foi obtido com a configuração P8R1 com janelas de 20x20 pixels. Com uma taxa de erro igual em torno de 2%, esta configuração se mostrou muito eficaz, porém, a exemplo da outra configuração com janelas, seu desempenho no quesito tempo de processamento é baixo o que a torna inviável para uso nos dispositivos testados, independentemente do modo de operação.

Portanto, a configuração P8R2 sem janelas torna-se a mais indicada por apresentar a melhor relação entre eficácia e tempo de processamento; com uma taxa de erro em torno de 4,4% e um tempo de processamento médio no *Tablet* em torno de 1,9 segundos essa configuração pode ser considerada viável, especialmente para operações de autenticação.

Nota-se que quando adotado raio 2 a configuração sem janelas apresentou um resultado melhor que a configuração com janelas, quando se esperava um resultado contrário. A causa provável deste efeito está na perda da informação das bordas de cada janela pois

nesta área da imagem o operador LBP não pode atuar uma vez que ele precisa dos dados da vizinhança em seu cálculo. Os pixels de borda não possuem todos os vizinhos e, portanto, não são considerados e quanto maior o raio maior será a área de borda. O tamanho da borda das janelas levou a um resultado diferente do esperado nas configurações com janelamento 20x20 pixels: a configuração P8R2 apresentou um tempo de processamento menor que a configuração P8R1. Isto ocorreu porque a perda da informação localizada na borda de raio 2 foi maior que na borda de raio 1 e com isso houve uma diminuição da quantidade de cálculos necessários para compor o descritor da imagem, levando a um tempo de processamento melhor.

CAPÍTULO 7. **Conclusão**

Vivemos em uma sociedade cada vez mais conectada e dependente da tecnologia que a permeia. Esta realidade se revela no crescente número de dispositivos móveis disponíveis atualmente, com destaque para os *smartphones*.

Este cenário de mobilidade "*online*" promove o aumento de transações eletrônicas e, consequentemente, a necessidade de aumentar a segurança dos dados, respeitando as limitações de hardware e o modo de interação com o usuário deste tipo de dispositivo. A biometria vai ao encontro destas questões, se apresentando com uma opção eficiente e conveniente para a autenticação de usuários nesses dispositivos.

Esta dissertação de mestrado teve como objetivo apresentar um estudo sobre a reconhecimento facial em dispositivos móveis que culminou com o projeto e a implementação de um sistema na plataforma Android, denominado BioMobile.

A adoção da plataforma Google Android se deveu ao fato desse sistema operacional ser um software livre e de código aberto, além de estar sendo rapidamente adotado por vários segmentos da sociedade, tais como a comunidade de desenvolvedores de aplicações, a indústria de dispositivos móveis, os fornecedores de software, as empresas de telefonia móvel, e muitos outros. Além disso, a plataforma adota a linguagem de programação Java que é robusta, de fácil sintaxe e possui total suporte ao paradigma de orientação a objetos.

Outra característica que contribuiu para a adoção da plataforma Android é o kit de desenvolvimento para códigos nativos chamado de NDK (*Native Development Kit*), que permite a execução de códigos como C e C++. Isto é importante, pois no desenvolvimento do BioMobile foi adotada a implementação da API OpenCV para os algoritmos Viola-Jones e PCA.

Devido ao seu reconhecido bom desempenho e aceitação pela comunidade, o algoritmo Viola-Jones foi a escolha para a fase de detecção de faces.

A opção por trabalhar com vídeos em detrimento a fotos confere ao sistema maior tolerância a falhas relacionadas a detecção da face. Além disso, a técnica de maioria de votos possibilita um melhor desempenho do sistema no quesito eficácia no processo de autenticação de faces.

Os resultados experimentais obtidos neste trabalho mostraram que, dada a alta

demanda computacional, o algoritmo baseado em PCA se torna inviável quando usado em dispositivos com restrições de memória e de processamento. Neste cenário, os descritores baseados em LBP mostraram-se mais adequados. Os resultados obtidos também indicam que a configuração P8R1 sem janelas não é interessante devido ao seu baixo desempenho no quesito eficácia, mesmo apresentando os melhores tempos de processamento. Também foi observado que as configurações com janelas não são convenientes por apresentarem altos tempos de processamento, embora a configuração P8R1 tenha apresentado uma ótima taxa de erro. Assim, a melhor configuração do BioMobile, de acordo com os experimentos realizados utilizando-se a base de dados Mobio, foi a P8R2 sem janelas, por apresentar a melhor relação entre taxa de erro e tempo de processamento, sendo viável sua adoção especialmente em operações de autenticação no *tablet*.

Como trabalhos futuros, indicamos a adoção da biblioteca (API) Face Detector da plataforma Android que usa a técnica baseada em segmentação da cor da pele para detecção de faces o que permitirá não apenas realizar uma comparação com a técnica Viola-Jones, mas também desenvolver uma aplicação sem a necessidade de uso de uma biblioteca adicional como o OpenCV, dispensando até mesmo o framework para uso de código nativo no Android chamado de NDK (*Native Development Kit*).

Podemos indicar também o desenvolvimento de aplicativos de reconhecimento de faces a partir do projeto desenvolvido visando solucionar problemas específicos como, por exemplo, apoio a deficientes visuais na identificação de pessoas em um determinado ambiente.

Referências Bibliográficas

AHONEN, TIMO; HADID, ABDENOUR; PIETIKÄINEN, MATTI. Face Recognition with Local Binary Patterns, Eighth European Conference Computer Vision, p. 469-481, 2004.

AHONEN, TIMO; HADID, ABDENOUR; PIETIKÄINEN, MATTI. Face Description with Local Binary Patterns: Application to Face Recognition, IEEE Transactions On Pattern Analysis and Machine Intelligence, v. 28, n. 12, 2006.

ANDROID. Android Developer Guide. Disponível em <http://developer.android.com/>. Acesso em 15/12/2009.

BOLLE et al. Guide To Biometrics. Springer Professional Computing, 1st edition. 2004.

BRADSKI, G.R.; PISAREVSKY, V. Intel's Computer Vision Library: applications in calibration, stereo segmentation, tracking, gesture, face and object recognition. Computer Vision and Pattern Recognition. Proceedings. IEEE Conference on, vol.2, p. 796 - 797. 2000.

CHANG-YEON, JO. Face Detection Using LBP Features. Stanford University. Vol 1, p. 1 - 4. 2008. Disponível em Acesso em 28/12/2011. http://www.stanford.edu/class/cs229/proj2008/Jo-FaceDetectionUsingLBPfeatures.pdf.

COSTA, LUCIANO R.; OBELHEIRO, RAFAEL R.; FRAGA, JONI S. Introdução à Biometria. Livro-texto dos Minicursos, VI Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SBSeg'2006), p. 103-151. Santos, SP, 2006.

DEITEL, HARVEY M.; DEITEL, PAUL J. Java: Como Programar. 6ª Ed. Porto Alegre: Bookman. 2005.

FERREIRA, M. E MORAES, A. Tutorial OpenCV. Tecgraf, PUC – Rio de Janeiro. 2007. Disponível em <http://www.tecgraf.puc-rio.br/~malf/opencv/>. Acesso em 15/12/2009.

FIGUEIREDO, CARLOS M. S.; NAKAMURA, EDUARDO. Computação Móvel: Novas Oportunidades e Novos Desafios. T&C Amazônia, Ano 1, nº 2, Junho 2003. Página 16. Disponível em <https://portal.fucapi.br/tec/index.php?sidrevista=2>. Acesso: 30 de dezembro de 2009.

FONG, LEONG LAI; SENG, WOO CHAW. User Authentication On Mobile Phones – What Is The Best Approach?. Proceeding of the 3rd International Conference on Informatics and Technology. 2009.

HADID, A.; HEIKKILÄ, J.Y; SILVEN, O.; PIETIKÄINEM, M. Face and Eye Detection

for Person Authentication in Mobile Phones. 2007.

HASLINGER, M. Protótipo para Localização de Pontos de Referência na Cidade de Chapecó Utilizando Google Android e Google Maps. 2009.

HORSTMANN, CAY S.; CORNELL, GARY. Core JAVA 2. Volume I, Fundamentos. São Paulo: Pearson Makron Books. 2003.

IJIRI, YOSHIHISA; SAKURAGI, MIHARU; LAO, SHIHONG. Security Management for Mobile Device by Face Recognition. 2006.

JAIN, A. K. Face Recognition. Disponível em: <http://biometrics.cse.msu.edu>. Acessado em 03 de janeiro de 2010. 2009.

JAIN, A. K.; ROSS, A.; PRABHAKAR, S. An Introduction to Biometric Recognition, IEEE Transactions on Circuits and Systems for Video Technology Special Issue on Image and Video-Based Biometrics. v. 14, n. 1, p. 4-20, 2004.

KHALILI, AMIR H. OpenCV Tutorial, Sharif University of Technology. 2007.

KIRBY, M.; SIROVICH, L. Application Of The Karhunen-Loéve Procedure For The Characterization Of Human Faces. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 12 n. 1:103-108, 1990.

KUMAR, SHISHIR et. al. Architecture for Mobile based Face Detection / Recognition. (IJCSE) International Journal on Computer Science and Engineering, v. 02, n. 03, 2010, 889-894, 2010.

LECHETA, RICARDO R. Google Android: aprenda a criar aplicações para dispositivos móveis com o Android SDK. São Paulo: Novatec Editora, 2009.

LIN, SHANG-HUNG. An Introduction to Face Recognition Tecnology. Informing Science, Special Issue on Multimedia Informing Tecnologies, Part 2, Volume 3, No 1, 2000.

MCCOOL, CHRISTOPHER; MARCEL, SÉBASTIEN. MOBIO Database for the ICPR 2010 Face and Speech Competition. Idiap Research Institute, Switzerland, 2010.

MICROSOFT. Windows Embedded. Disponível em <http://www.microsoft.com/windowsembedded/en-us/default.mspx>. Acesso em 03 de janeiro de 2010.

MILLER, B. Vital signs of identity. IEEE Spectrum. 1994.

NOKIA. The Symbian Platform. 2000. Disponível em

<nds2.ir.nokia.com/NOKIA_COM_1/About_Nokia/Press/White_Papers/pdf_files/dec002_net.pdf>. Acesso em 31 de dezembro de 2009.

PABBARAJU, ADITYA; PUCHAKAYALA, SRUJANKUMAR. Face Recognition in Mobile Devices. Electrical Engineering and Computer Science, University of Michigan, 2010.

ROGERS, R.; LOMBARDO, J.; MEDNIEKS, Z.; MEIKE, B. Desenvolvimento de Aplicações Android. São Paulo: Novatec Editora, 2009.

SYMBIAN. The Symbian Foundation Community. Disponível em: <http://www.symbian.org>. Acesso: 28 de dezembro de 2009.

SUN MICROSYSTEMS. Java ME Technology. Disponível em <http://java.sun.com/javame/technology/index.jsp>. Acesso em 31 de dezembro de 2009.

TIEU, K; VIOLA, P. Boosting Image Retrieval. International Journal of Computer Vision, vol.1, p. 228-235. 2000.

TURK, M.; PENTLAND, A. Face recognition using eigenfaces. In IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pages 586–591, Maui, HI, USA. 1991.

VIOLA, P.; JONES, M. Rapid Object Detection using a Boosted Cascade of Simple Features. Conference On Computer Vision And Pattern Recognition. 2001.

VIOLA, P.; JONES, M. Robust Real-Time Object Detection, International Journal of Computer Vision, vol. 57, no.2, p. 137-154. 2001.

VIOLA, P.; JONES, M. Robust Real-Time Face Detection. International Journal of Computer Vision 57(2), 137-154. 2004.

YU, HAO. Face Recognition for Mobile Phone Using Eigenfaces. Department of Mechanical Engineering, University of Michigan. 2010.

ZHAO, W.; CHELLAPPA, R.; PHILLIPS, P. J.; ROSENFELD, A. Face recognition: A literature survey. ACM Computing Surveys. 2003.