UNIVERSIDADE FEDERAL DE ITAJUBÁ

PROGRAMA DE PÓS-GRADUAÇÃO EM
CIÊNCIA E TECNOLOGIA DA COMPUTAÇÃO

Indexação de Faces em Estruturas de Dados Métricas

Rodrigo Lúcio dos Santos Silva

Itajubá, novembro de 2012

UNIVERSIDADE FEDERAL DE ITAJUBÁ

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA E TECNOLOGIA DA COMPUTAÇÃO

Rodrigo Lúcio dos Santos Silva

Indexação de Faces em Estruturas de Dados Métricas

Dissertação submetida ao Programa de Pós-Graduação em Ciência e Tecnologia da Computação como parte dos requisitos para obtenção do Título de Mestre em Ciências em Ciência e Tecnologia da Computação

Área de Concentração: Sistemas de Computação

Orientador: Prof. Dr. Enzo Seraphim

Novembro de 2012

Itajubá - MG

Ficha catalográfica elaborada pela Biblioteca Mauá –
Bibliotecária Cristiane N. C. Carpinteiro- CRB_6/1702

S586i

Silva, Rodrigo Lúcio dos Santos

Indexação de faces em estruturas de dados métricas. / por Rodrigo Lúcio dos Santos Silva. -- Itajubá (MG) : [s.n.], 2012.

45 p. : il.

Orientador: Prof. Dr. Enzo Seraphim.
Dissertação (Mestrado) – Universidade Federal de Itajubá.

1. Indexação de faces. 2. Extratores de características. 3. Similaridade de faces. I. Seraphim, Enzo, orient. II. Universidade Federal de Itajubá. III. Título.

Ao meu filho Matheus.

À minha esposa Maria luzia.

À minha Mãe Maria Maura.

“A mente que se abre a uma nova ideia
jamais voltará ao seu tamanho original.”
Albert Einstein

Agradecimentos

Agradeço primeiramente a Deus pelas oportunidades que eu tive na vida e por ter sempre me iluminado nos meus trabalhos.

A minha mãe por ter me mostrado a importância dos estudos. Sempre com muito carinho me guiando desde pequeno no caminho de uma vida digna.

Ao meu orientador, Prof. Dr. Enzo Seraphim pelo apoio e confiança durante a orientação.

A minha esposa pela paciência e palavras de incentivo nos momentos mais difíceis.

Ao meu filho que me dá a inspiração e motivação fazendo com que eu busque melhorar sempre.

Aos meus irmãos Eduardo e Marcela e ao meu pai José Roberto que sempre me ajudaram e me apoiaram em tudo.

A todos os meus familiares que direta ou indiretamente sempre me ajudaram durante toda a minha formação profissional.

A Universidade Federal de Itajubá (UNIFEI) e ao Departamento de Suporte a Informática (DSI) pelo incentivo a qualificação de seus funcionários.

Sumário

Lista de Figuras

Resumo

Atualmente existe uma imensa quantidade de fotos digitalizadas de faces de pessoas utilizadas para as mais variadas aplicações, o que demanda estruturas de dados eficientes para realizar consultas das faces em um banco de dados. As consultas tradicionais usam a relação de ordem total entre os elementos para minimizar o tempo nos seus algoritmos. Essas consultas tradicionais não são adequados para manipular faces, pois dificilmente duas fotos (mesmo que de uma mesma pessoa) apresentam valores idênticos em todos os pixeis. Portanto, é mais adequado utilizar o critério de similaridade para recuperar as faces mais parecidas a um determinado indivíduo. As estruturas de dados que apresentam melhor desempenho para realizar consultas por similaridade são as métricas. Este trabalho tem por objetivo indexar imagens de faces em estrutura de dados métricas. Para representar a face na estrutura de dados serão explorados métodos de extração de características: *Eigenfaces* e *Scale Invariante Feature Transform* (SIFT). Essas características são representadas por vetores que devem ser capazes de descrever o grau de dissimilaridade entre duas faces. Para calcular a dissimilaridade entre vetores de características foi utilizado as funções de distância euclidiana e média das mínimas distâncias euclidianas. Os experimentos mostraram que extrator SIFT que calcula a distância pela média das mínimas distâncias euclidianas demonstrou ter melhor precisão e revocação comparado com o extrator *Eigenfaces* que calcula a distância pela euclidiana. A vantagem do uso do extrator SIFT é que não há a necessidade de reprocessamento da base antes da inserção dos dados.

Abstract

Currently there is an immense amount of scanned photos of people's faces used for several applications, which demand efficient data structures to perform queries of the faces in a database. Traditional queries use the total order relation between the elements to minimize the time in their algorithms. These queries are not suitable for manipulate faces, because hardly two photos (even of the same person) have identical values at all pixels. Therefore, it is more appropriate to use the criterion of similarity to recover faces more similar of a particular individual. The data structures that provide best to perform for similarity queries are the metrics. This work aims to index images of faces in metric data structure. To represent the face in the data structure will be explored methods of feature extraction: Eigenfaces and Scale Invariante Feature Transform (SIFT). These characteristics are represented by vectors that must be able to describe the degree of dissimilarity between two faces. To calculate the dissimilarity between feature vectors was used functions of Euclidean distance and average of minimums Euclidean distance. The experiments showed that SIFT extractor that calculates the distance by average of minimums Euclidean distance shown to have better precision and recall compared with the extractor Eigenfaces that calculates the distance by Euclidean. The advantage of using SIFT extractor is that there is a need for reprocessing the base prior to insertion of data.

Capítulo 1

1. Introdução

A popularização dos meios de digitalização através do uso de dispositivos como máquinas fotográficas e celulares gerou um aumento expressivo no armazenamento de imagens que pode conter faces de pessoas. Essas imagens são obtidas para diversos fins tais como, postagem de imagens em páginas de redes sociais, auditoria de imagens de câmeras de segurança, registros históricos de eventos, etc.

Essas faces em imagens são exemplos de dados complexos e existe uma grande demanda para armazenamento e recuperação. Dados complexos não apresentam a relação de ordem total e devem ser armazenados em estruturas de dados apropriadas de maneira a aumentar a eficiência das consultas (FALOUTSOS, 1996).

Dados que apresentam relação de ordem são recuperados por consultas feitas pelos gerenciadores de banco de dados convencionais e envolvem critérios baseados na igualdade e na ordem dos dados. Exemplos de dados que permitem este tipo de consulta são números e textos. Esses critérios não seriam adequados para consultar faces, pois dificilmente duas fotos (mesmo que de uma mesma pessoa) apresentam valores idênticos em todos os pixels. A expressão facial muda, o angulo muda, as dimensões, etc. Portanto, é mais adequado utilizar o critério de semelhança para obter as faces mais parecidas a um determinado indivíduo (PATELLA, 1999).

Consultas que usam critério de semelhança são eficientemente manipulados por estruturas de dados métricas e as estruturas de dados espaciais (CIACCIA; PATELLA; ZEZULA, 1997). Essas estruturas possibilitam consultas por abrangência e aos k-vizinhos mais próximos. Um exemplo de consulta por abrangência em uma base de dados de faces é “selecione as faces que diferem da face de Pelé por até duas unidades de distância”. Um exemplo de consulta aos k-vizinhos mais próximos em uma base de dados de faces é “selecione as cinco faces mais semelhantes ao Pelé”.

Um exemplo de estruturas de dados métricas é a M-tree(CIACCIA; PATELLA; ZEZULA, 1997) que para tornar possível a indexação utiliza-se de uma função de dissimilaridade que determina o grau de semelhança entre as faces. Essa função deve retornar

valores próximos de zero para faces parecidas. Essa estrutura possibilita indexar dados com dimensão fixa e variável.

A estrutura M-Tree garante melhor desempenho no processamento das consultas por abrangência e aos vizinhos mais próximos, pois realiza verificações em apenas uma parte do conjunto de dados.

1.1 Motivação

Existem diversas aplicações que podem se beneficiar com indexação e consultas de faces semelhantes, entre elas, sistema de controle de acesso, sistemas de monitoramento, e máquinas de busca de faces na web.

Uma máquina de busca convencional representa uma base de dados contendo os endereços das mais variadas páginas da internet, indexadas por palavras para realizar a busca. Na prática, estes serviços servem a diferentes propósitos, como por exemplo, procura de endereços/telefones de pessoas, busca de imagens por palavras, procura pelo melhor caminho em mapas, busca por temas da área de saúde, entre outras. No entanto, a maioria dessas máquinas de busca utilizam como parâmetro para a consulta uma ou mais palavras.

A internet também tem uma infinidade de imagens que em grande parte são fotos de pessoas. Essa grande quantidade de imagens na Web motiva a criação de uma máquina de buscas que permita aos usuários usarem uma imagem de uma face como parâmetro de sua consulta para localizar as imagens com faces mais semelhantes.

No trabalho de (PAES; SERAPHIM, 2012) é proposto a arquitetura de uma máquina de busca de faces na Web, como mostrado na Figura 1. A arquitetura é composta por 3 módulos:

- Crawler
- Máquina de Busca
- Interface com o Usuário

O módulo Crawler da Figura 1, também conhecido como spider, Web wanderer, Web worm e robot, é um programa que navega pelas páginas da World Wide Web através dos seus links (KOBAYASHI; TAKEDA, 2000). Durante a navegação, os crawlers podem ser utilizados para coletar informações nas páginas da Web, como por exemplo, as faces das imagens.

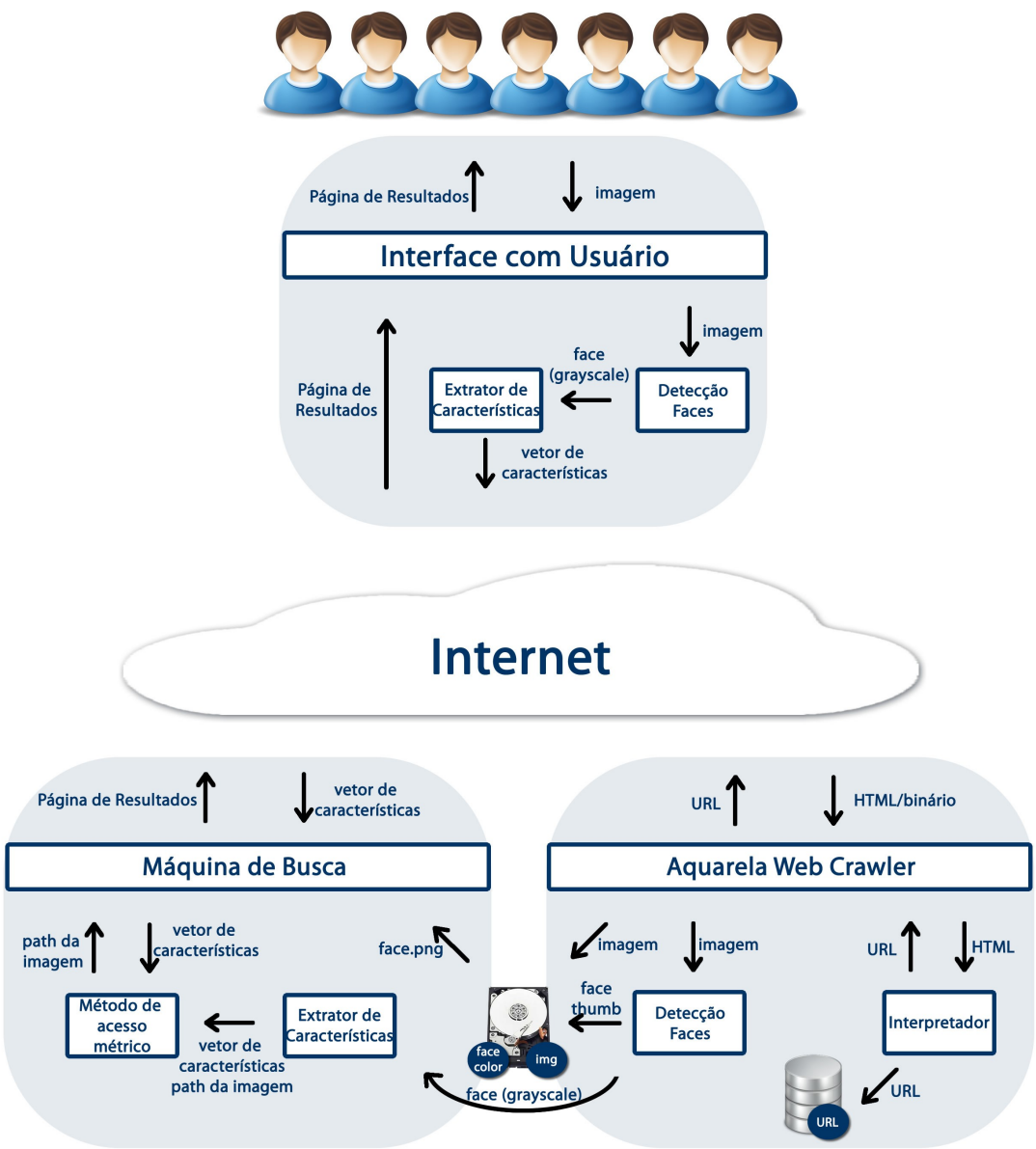


Figura 1: Máquina de Buscas de Faces na Web (PAES; SERAPHIM, 2012)

O módulo Máquina de Busca (Figura 1) é responsável em organizar as características das imagens em uma estrutura de dados métrica afim de otimizar as consultas. Desta forma, há uma aplicação Web que recebe solicitações da Internet sobre faces a serem pesquisadas, e retorna uma página de resultados ao usuário.

Finalmente, o módulo Interface com Usuário da Figura 1 é diretamente ligado na interação do usuário com a solução de busca. Este módulo é responsável pela interface final com o usuário, o extrator de características no lado do cliente, e a comunicação com a ferramenta.

O trabalho de Paes validou a implementação do módulo de crawler e esse trabalho pode ser usado no módulo máquina de busca de sua arquitetura.

1.2 Objetivo

O objetivo deste trabalho é agilizar o processo de busca por imagens de faces usando uma representação eficiente para a imagem em um Método de Acesso Métrico (MAM). É desejável que essa agilidade do processo esteja associado a uma medida de qualidade no resultado das consultas.

Para atingir esse objetivo deve-se explorar 3 pontos: técnicas de representação de imagens de faces; algoritmos que aferem a dissimilaridade; e indexação usando métodos de acesso métrico (MAM).

Para a representação de imagens de faces serão comparados os vetores de características gerados a partir dos extratores *eigenfaces* e SIFT.

Os algoritmos que aferem a dissimilaridade entre as faces dependem dos vetores de características extraídos na representação para realizar a comparação e devem retornar valores próximos de zero quando duas faces são semelhantes.

Os MAMs são mecanismos eficientes para processamento de consultas por abrangência e de consultas aos k-vizinhos mais próximos e utilizamos vetores de características associado a função de dissimilaridade.

1.3 Organização do Trabalho

Essa dissertação está organizada da seguinte maneira: o capítulo 2 apresenta a revisão bibliográfica sobre os temas envolvidos neste trabalho, abordando os conceitos sobre métodos de acesso métrico e processamento digital de imagens. Na seção métodos de acesso métrico será apresentado o método de acesso métrico M-Tree e as duas formas mais comuns de consulta: por

abrangência e aos vizinhos mais próximos. Na seção processamento digital de imagens será apresentado técnicas para reconhecimento de faces e para extração de característica com o método eigenfaces e com o método SIFT.

O capítulo 3 apresenta a indexação de faces usando método eigenfaces e método SIFT. Neste capítulo são apresentadas funções de dissimilaridade: a mínima distância euclidiana e a média das mínimas distâncias euclidianas.

Os experimentos, abordados no capítulo 4, medem a qualidade (precisão e a revocação) da resposta da consulta pelos k-vizinhos mais próximos sobre as faces indexadas na estrutura métrica M-tree. Todos os experimentos utilizam o framework ObInject(CARVALHO; SERAPHIM, 2012).

Finalmente no capítulo 5, é mostrado os resultados deste trabalho e o que pode ser feito como trabalhos futuros.

Capítulo 2

2. Revisão Bibliográfica

2.1 Estruturas de dados Métricas

2.1.1 Espaço Métrico

Um espaço métrico é definido por um par, M=(D, d) onde D é o domínio de valores de características - as chaves de indexação – e d é a função de dissimilaridade. Com as seguintes propriedades (CIACCIA; PATELLA; ZEZULA, 1997):

Simetria: $d(O_x, O_y) = d(O_y, O_x)$;

Não Negatividade: $d(O_x, O_y) > 0 (O_x \neq O_y)$ e $d(O_x, O_x) = 0$

Desigualdade triangular: $d(O_x, O_y) \leq d(O_x, O_z) + d(O_z, O_y)$

Sobre o espaço métrico, consultas por similaridade operam usando essas propriedades e avaliam o grau de dissimilaridade entre os objetos e devolve segundo determinados parâmetros aqueles objetos mais parecidos com um objeto de consulta. Os dois tipos de consulta por similaridade mais comuns são a consulta por abrangência e a consulta aos K-Vizinhos mais próximos(CIACCIA; PATELLA; ZEZULA, 1997).

A consulta por abrangência é definida por *range(Q, r(Q)), onde Q é* um objeto de consulta *Q* e *r(Q) é um* raio de consulta. Essa consulta devolve todos os objetos indexados O_j , tal que, $d(O_j, Q) \leq r(Q)$. Um exemplo de consulta por abrangência em uma base de dados de faces é "selecione as faces que diferem da face de Pelé por até duas unidades de distância".

A consulta aos K-Vizinhos mais próximos é definida por *kNeighbors(Q, k)*, onde *Q* é um objeto de consulta *Q e k* um inteiro maior que um. Essa consulta seleciona os *k* objetos indexados que tem a menor distância a *Q*. Um exemplo de consulta aos k-vizinhos mais próximos em uma base de dados de faces é "selecione as cinco faces mais semelhantes ao Pelé".

2.1.2 M-Tree

A M-Tree é uma árvore balanceada capaz de lidar com arquivos de dados dinâmicos, pois, não requer reorganizações periódicas. Isto significa que a estrutura pode ser construída gradualmente, sem a necessidade de ter o conjunto inteiro de dados para iniciar a construção. A M-Tree pode indexar objetos usando uma função de distância, a qual não precisam estar em um espaço vetorial ou não precisam usar uma métrica Lp(CIACCIA; PATELLA; ZEZULA, 1997).

Essa estrutura particiona os objetos com base em suas distâncias, medida pela função de distância d(), e armazena esses objetos em nós de tamanhos fixos.

A estrutura M-Tree é formada por nós folhas e nós índices. Os nós folhas armazenam todos os objetos em si, enquanto que nós índices armazenam os chamados objetos representantes.

O nó folha é formado por várias entradas, sendo que cada entrada tem as três características:

O_j	$oid(O_j)$	$d(O_j, P(O_j))$

Onde:

O_j : Objeto indexado (descritores da imagem da face)

$oid(O_j)$: Identificador do 0bjeto

$d(O_j, P(O_j))$: Distância do objeto ao seu pai

O nó índice também é formado por várias entradas, sendo que cada entrada tem as quatro características:

O_r	$ptr(T(O_r))$	$r(O_r)$	$d(O_r, P(O_r))$

Onde:

O_r : Objeto representante (descritores da imagem da face)

$ptr(T(O_r))$: Ponteiro para subárvore (árvore de cobertura de Or)

$r(O_r)$: Raio de cobertura

$d(O_r, P(O_r))$: Distância de O_r ao seu representante (pai)

O algoritmo para inserção de um novo objeto na M-Tree busca recursivamente pelo nó folha mais apropriado, que não exija o aumento do raio de cobertura.

Para todos objetos O_r cujo raio de cobertura englobem o novo objeto O_n, escolhe-se aquele que tem a menor distância a O_n, $Min(d(O_r, O_n))$, e continua percorrendo pela subárvore $ptr(T(O_r))$.

Se não houver objetos que englobam o novo objeto O_n, escolhe-se o objeto O_r cujo raio de cobertura é mais próximo do novo objeto, $Min(d(O_r, O_n) - r(O_r))$, de maneira a minimizar o aumento do raio, atualiza o raio de cobertura de O_r, $r(O_r) = d(O_r, O_n)$, e continua percorrendo pela subárvore $ptr(T(O_r))$.

Quando a recursão chega ao nó folha, o novo objeto O_n é inserido. Se não houver espaço é necessário fazer a quebra do nó (*split*).

Ao fazer a quebra do nó, tanto de um nó folha quanto de um nó índice, aloca-se um novo nó N' no mesmo nível do nó N, que está sofrendo a quebra. Escolhe-se dois objetos O_{p1} e O_{p2} do nó a serem promovidos, distribui os objetos entre o no N que foi particionado e o novo nó N'. Quais nós a escolher a serem promovidos e como os objetos são distribuídos entre N é que definem a politica da quebra do nó.

A política ideal de quebra do nó visa minimizar o volume e minimizar as sobreposições de maneira a diminuir a quantidade de nós vazios e diminuir o número de possibilidades de caminhos durante a busca.

A escolha dos objetos a serem promovidos pode ser:

- ***m_RAD***: É o mais complexo em termos de computação de distâncias. Ele considera todos os possíveis pares de objetos, e depois de particionar o conjunto de entradas, promove o par de objetos os quais a soma do raio de cobertura, $r(O_{p1}) + r(O_{p2})$, é mínima.
- ***mM_RAD***: Similar ao m_RAD, entretanto é minimizado o máximo dos dois raios de cobertura.
- ***m_LB_DIST***: Maximum Lower Bound Distance. Usa apenas as distâncias pré computadas armazenadas. Considerando $O_{p1} \equiv O_p$. O_{p2} é determinado como o objeto mais distante de O_p:
  $d(O_{p2}, O_p) = max_j\{d(O_j, O_p)\}$

- ***RANDOM***: Seleciona de maneira randômica os objetos a serem promovidos. Embora não seja um algoritmo inteligente, é rápido e pode ser usado para comparar com as outras politicas.
- ***SAMPLING***: É a politica RANDOM, mas aplicada uma amostra de objetos de tamanho s > 1. É aplicado então o mM_RAD a esta amostra. Nos experimentos de (CIACCIA; PATELLA; ZEZULA, 1997) foram usadas amostras com tamanho de um décimo da capacidade dos nós.

Após a escolha dos dois objetos representativos O_{p1} e O_{p2} , o conjunto de entradas N do nó que está sofrendo quebra pode ser particionado em dois subconjuntos N_1 e N_2 seguindo os seguintes tipos de distribuição:

- ***Hiperplano generalizado***: Cada objeto O_j pertencente a N é atribuído ao objeto roteador mais próximo:

```
se (d(O_j, O_p1) ≤ d(O_j, O_p2)) então
    atribuir Oj a N_1
senão
    atribuir Oj a N_2
```

- ***Balanceada***: São calculadas as distâncias $d(O_j, O_{p1})$ e $d(O_j, O_{p2})$ para todos objetos Oj pertencentes a N . Em seguida, os seguintes passos são repetidos até que o conjunto N fique vazio:
  - Atribuir a N_1 o vizinho mais próximo de O_{p1} em N , e remover ele de N
  - Atribuir a N_2 o vizinho mais próximo de O_{p2} em N , e remover ele de N

2.1.3 Framework Object-Injection

O *Object-Injection* é um framework para indexação e persistência de objetos. A persistência dos objetos é realizada usando índices primários e a indexação de chaves é feita

pelos índices secundários. A principal característica do framework é permitir que objetos sejam injetados e indexados em qualquer estrutura de dados que pode ser armazenada em qualquer dispositivo(CARVALHO; SERAPHIM, 2012). Como pode ser visto na Figura 2, o framework *Object-Injection* é dividido em quatro módulos de abstração:

- **Módulo de objetos**: define as entidades persistentes e o domínio das chaves indexadas
- **Módulo de armazenamento**: define as estruturas de índices primários para as entidades persistentes e as estruturas de índices secundários para as chaves indexadas.
- **Módulo de dispositivos**: define como é o armazenamento físico das estruturas de índices
- **Módulo de blocos**: define a maneira pela qual as entidades persistentes e as chaves indexadas são armazenadas em blocos ou páginas.

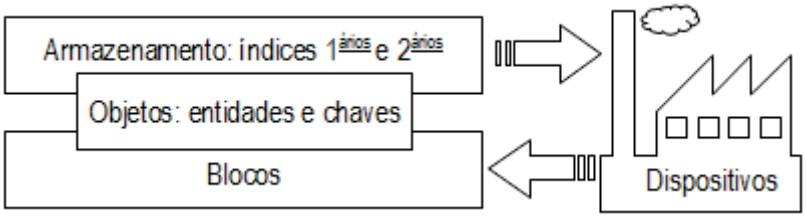


Figura 2: Módulos do framework ObInject (CARVALHO; SERAPHIM, 2012)

2.2 Processamento Digital de Imagens

Uma das áreas de aplicação de técnicas de processamento de imagens digitais, consiste na solução de problemas relacionados a percepção por máquina. Procedimentos para extrair de uma imagem informação de uma forma adequada para o processamento computacional. Exemplos do tipo de informação usado em percepção por máquinas são os momentos estatísticos, os coeficientes da transformada de Fourier e medidas de distâncias multidimensionais(GONZALEZ; WOODS, [S.d.]).

A Figura 3 ilustra os passos fundamentais necessários para executar uma tarefa de processamento de imagem. No caso deste trabalho o domínio o problema é a imagem de faces de pessoas, onde deve-se encontrar uma representação da imagem da face, que seja possível medir o quanto uma face se parece com uma outra. Assim, dentre os passos mostrado na Figura 3 aqueles que atendem o escopo do trabalho são: Representação e descrição, Reconhecimento e interpretação.

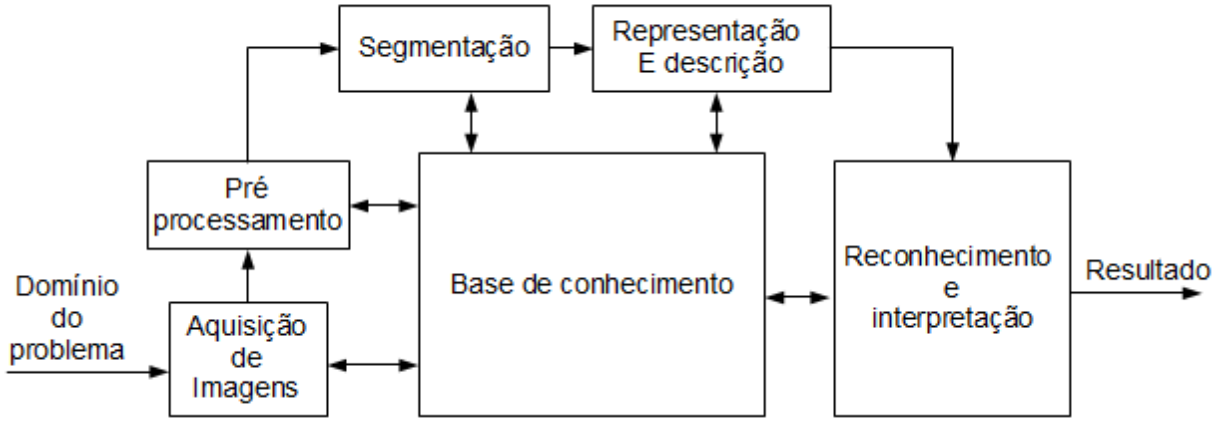


Figura 3: Passos fundamentais em processamento de imagens digitais (GONZALEZ; WOODS, [S.d.])

Os dados podem ser representados como fronteiras ou como regiões completas. A representação como fronteira é adequada quando o interesse se concentra nas características da forma externa, tais como cantos ou pontos de inflexão. A representação por região é adequada quando o interesse se concentra em propriedades internas, tais como textura ou a forma do esqueleto. Em algumas aplicações entretanto, essas representações coexistem.

A escolha de uma representação é parte da solução para transformar os dados iniciais numa forma adequada para o subsequente processamento computacional. Um método para descrever os dados também deve ser especificado, de forma que as características de interesse sejam enfatizadas. O processo de descrição, também chamado de seleção de características, procura extrair características que resultem em alguma informação quantitativa de interesse ou que sejam básicas para discriminação entre classes de objetos (GONZALEZ; WOODS, [S.d.]).

O reconhecimento é o processo que atribui um rótulo a um objeto, baseado na informação fornecida pelo seu descritor. A interpretação envolve a atribuição de significado a um conjunto de objetos conhecidos.

2.2.1 Reconhecimento de Padrões

Reconhecimento de padrões tem como objetivo a classificação de objetos em categorias ou classes. Um padrão é uma descrição quantitativa ou estrutural de um objeto ou alguma outra entidade de interesse em uma imagem. Em geral, um padrão é formado por um ou mais descritores, ou seja, um padrão é um arranjo de descritores. Uma classe de padrões é uma família de padrões que compartilham algumas propriedades comuns. As classes de padrões são denotadas como $w_{1,}w_{2,\ldots},w_M$ onde M é o número de classes. O reconhecimento de padrões por máquina envolve técnicas para a atribuição dos padrões as suas respectivas classes(GONZALEZ; WOODS, [S.d.]).

Os principais arranjos de padrões usados na prática são os vetores. Vetores de padrões são podem ser representados na forma:

$$x=\begin{bmatrix} x_1 \\ x_2 \\ \ldots \\ x_n \end{bmatrix} \tag{1}$$

Em que cada componente, x_i , representa o i-ésimo descritor, e n é a quantidade de descritores.

2.2.2 Reconhecimento de Faces

O reconhecimento de faces apresenta um problema desafiador no campo de analise de imagens e visão computacional e tem recebido bastante atenção nos últimos anos, devido a varias aplicações em vários domínios. O problema de reconhecimento de faces pode ser definido como: Dado uma imagem de face de entrada e um banco de dados de imagem de faces de

indivíduos conhecidos, como podemos verificar ou determinar a identidade da pessoa na imagem de entrada? (JAFRI; ARABNIA, [S.d.]).

Podemos reformular o problema de reconhecimento de faces para o problema de indexar faces das imagens como: Dado uma imagem de face de entrada e um banco de dados de imagens de faces, como podemos consultar as imagens mais parecidas, com a pessoa na imagem da entrada? Nota-se que neste caso, os indivíduos não necessariamente são conhecidos e a resposta pode conter indivíduos que não correspondem com a pessoa na imagem de entrada.

Os processos de reconhecimento de face podem ser baseado em duas abordagens:

- **Abordagem global** que é baseada na aparência da face. Esta abordagem visa reduzir uma imagem de milhares de *pixels* em um conjunto de números.
- **Abordagem local** que é baseada na geometria da face. Esta abordagem visa modelar a face em termos da localização geométrica de diversos pontos nodais, tais como: olhos, boca, nariz, cavidade orbital, ossos laterais da face e do queixo. As medidas registradas são, então, transformadas em um algoritmo, tornando-se a "matriz" da assinatura biométrica facial do indivíduo. Assim, o reconhecimento de face se resume em comparar os sistemas geométricos obtidos.

Dentre os vários métodos de reconhecimento de faces globais e locais disponíveis, estamos interessados naqueles que geram descritores que possam ser comparados por uma função de distância métrica. Patella em (PATELLA, 1999) sugere o extrator de característica global chamado *eigenfaces*, que usa os autovetores e o autovalores da matriz de covariância da base de imagem de faces para encontrar os componentes principais da distribuição de faces. Como resultado do algoritmo cada imagem de face é representada por um vetor de valores numéricos e a métrica euclidiana pode ser usada para indexar a face.

Em (SILVA, 2009) é usado o vetor de características gerado pelo *eigenfaces* para indexar imagens de faces em estruturas de dados métricas. São mostrados experimentos com várias métricas. Contudo, seus experimentos utilizam estruturas de dados métricas armazenadas em memória primária. Neste trabalho, é sugerido em trabalhos futuros a comparação entre os extratores *Eigenfaces* e SIFT.

Assim, este trabalho explora as duas técnicas para reconhecimento de faces que usam abordagens globais descritas nas próximas seções.

2.2.3 Extrator de características *Eigenface*

Proposto por Turk e Pentland(TURK; PENTLAND, 1991) este método de reconhecimento de faces descreve um esquema para reconhecimento baseado na teoria da informação, procurando codificar as mais relevantes informações de um grupo de faces, que irá melhor distinguir elas umas das outras.

Transforma as imagens de faces em um pequeno conjunto de características, chamados *eigenfaces*, que são as componentes principais do conjunto inicial de imagens de treinamento. O reconhecimento é realizado projetando a nova imagem no subespaço gerado, derivado pelas *eigenfaces* ('Espaço de faces') e então classificando a face comparando sua posição no espaço de faces com a posição dos indivíduos conhecidos.

2.2.3.1 Algoritmo Eigenface

Seja uma determinada imagem de face $I(x, y)$ num espaço bidimensional NxN. Essa imagem pode ser vista como um vetor de dimensão N^2. Por exemplo uma imagem de dimensão $256 X 256$ pode ser expressa como um vetor de dimensão 65.536 e pode ser considerada um ponto num espaço com 65.536 dimensões. Dessa forma um conjunto de imagens de faces pode mapear um grupo de pontos nesse imenso espaço, e como as faces são semelhantes de uma forma geral(tem dois olhos, nariz e boca), elas não vão se distribuir de maneira aleatória nesse espaço.

O princípio do algoritmo é encontrar os vetores que melhor representam a distribuição de imagem de faces dentro do espaço de todas as imagens de faces. Esses vetores vão definir o subespaço das imagens de faces (denominado 'espaço de faces')

Seja Z o conjunto de treinamento de imagens de faces:

$$Z = \{Z_{1,} Z_2, \cdots, Z_M\} \tag{2}$$

Cada imagem Z_i é representada na forma de vetor coluna:

$$I_i = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N2} & \cdots & a_{NN} \end{bmatrix} \rightarrow \quad Z_i = \begin{bmatrix} a_{11} \\ \vdots \\ a_{1N} \\ a_{21} \\ \vdots \\ a_{2N} \\ \vdots \\ a_{N1} \\ \vdots \\ a_{NN} \end{bmatrix} \tag{3}$$

A face média é definida por:

$$\psi = \frac{1}{M} \sum_{i=1}^{M} Zi \tag{4}$$

O vetor Φ_i representa a diferença de cada face em relação a face média:

$$\Phi_i = Z_i - \Psi \tag{5}$$

O propósito de subtrair cada imagem da média é para manter somente as características que distinguem cada imagem da face e remover de certa maneira as informações que são comuns. Dessa forma, se considerarmos que todas as faces serão subtraídas da face média, teremos uma nova matriz A que contem somente as variações de cada face em relação a face média:

$$A = \{\Phi_1, \Phi_2, \cdots, \Phi_M\} \tag{6}$$

A matriz de covariância do conjunto de treinamento pode ser dada por:

$$C = AA^T \tag{7}$$

O conjunto de vetores representado pela matriz A é então submetido à análise de componentes principais, dando origem aos autovetores u_i e aos autovalores λ_i da matriz de covariância C . Estes autovetores quando representados como imagem tem aparência muito semelhante a uma face, e por isso então foram denominados *eigenfaces*.

Como vimos o espaço das imagens originais tem dimensão N^2 , sendo muito grande e ineficiente para representar a distribuição de faces. É construído então o espaço de faces escolhendo uma quantidade K (bem menor do que N^2) dos autovetores com maiores valores de λ_i . Lembrando que o valor de K é escolhido heuristicamente e varia de acordo com a base de treinamento.

Agora cada face no conjunto de treinamento (menos a face média), Φ_i , pode ser representada como uma combinação linear desses autovetores u_i :

$$\Phi_i = \sum_{j=1}^{K} w_j u_j \quad \text{onde} \quad u_i \quad \text{são as eigenfaces} \tag{8}$$

Então os pesos podem ser calculados como:

$$\omega_j = u_j^T \Phi_i \tag{9}$$

Cada imagem de face é representada nessa base como um vetor:

$$\Omega_i = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_k \end{bmatrix} \quad \text{onde i = 1, 2, … M} \tag{10}$$

2.2.4 Extrator de características SIFT

Encontrar correspondência de características de duas imagens é uma tarefa comum em visão computacional. O método SIFT (Scale Invariante Feature Transform) transforma uma imagem em uma coleção de vetores de características locais, os quais são invariantes a translação, mudança de escala, rotação e parcialmente invariante a iluminação e ponto de vista 3D (LOWE, 1999).

O algoritmo SIFT é dividido em quatro partes (LOWE, 2004):

- **Detecção de extremos no espaço de escala**: A imagem é representada de forma a assegurarmos a invariância a escala. Estas representações construídas a partir da diferença gaussiana que permitem encontrar candidatos a pontos chave.

- **Localização dos Pontos Chaves**: Muitos candidatos a pontos chaves não são bons como pontos ao longo de bordas e regiões com baixo contraste. Com o cálculo da localização exata de cada ponto é possível descartar esses pontos considerados instáveis.
- **Atribuição de orientação**: Para permitir invariância a rotação é atribuída uma orientação a cada ponto chave. A orientação é calculada a partir do gradiente ao redor do ponto chave.
- **Descritor do ponto chave**: Na etapa final é criado uma representação única para cada ponto chave. Esta representação é normalizada para garantir a invariância a iluminação.

2.2.4.1 Detecção de Extremos no Espaço de Escala

Para ressaltar características importantes da imagem devemos levar em consideração a sua estrutura multi-escala. Por exemplo, às vezes é útil eliminar estruturas menores, como ruídos e pequenas texturas indesejáveis, antes de buscar estruturas maiores, como objetos, blobs, arestas ou cantos. Quando não conhecemos a escala é necessário representar a imagem em vários níveis de detalhe (em várias escalas). Assim cria-se um espaço de escala, uma sequência de imagens que vai passando por uma suavização, durante a qual, detalhes menores são eliminados rapidamente enquanto detalhes maiores sobrevivem por mais tempo.

O espaço de escalas é construído ao aplicar um filtro de borramento (tipicamente o filtro gaussiano) progressivamente na imagem:

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \tag{11}$$

Onde:

- L : Imagem borrada
- G : É o operador gaussiano
- I : Imagem que vai sofrer o borramento (pode ser uma imagem já filtrada anteriormente)
- x, y : Coordenada dos pixeis

- σ : Parametriza a quantidade de borramento, quanto maior mais borrada a imagem fica
- $*$: Operador de convolução em x e y

O operador gaussiano é definido por:

$$D(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2} \tag{12}$$

Ao criar o espaço de escala, pegamos a imagem original e aplicamos progressivamente o filtro gaussiano. Então a imagem é redimensionada para a metade do tamanho, e é aplicado o filtro gaussiano progressivamente novamente, agora na imagem reduzida. E assim sucessivamente.

A cada conjunto de imagens que passa pelo borramento dá se o nome de oitava, e cada imagem desse conjunto é a escala. O número de oitavas e escalas dependem do tamanho da imagem original. Esses valores devem ser parametrizados e influenciam na eficiência do algoritmo. O trabalho de Lowe, sugere 4 oitavas e 5 escalas (5 níveis de borramento).

As imagens que foram borradas pelo filtro da gaussiana agora são usadas para gerar um novo conjunto de imagens, as diferenças de gaussiana(DoG). Essas imagens DoG são ideais para encontrar pontos chaves na imagem, pois ressaltam bordas e cantos. A diferença de gaussiana $D(x, y, \sigma)$ é calculada pela diferença de duas escalas vizinhas separadas por um multiplicador constante k :

$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma) \tag{13}$$

A construção de $D(x, y, \sigma)$ é mostrado na Figura 4.

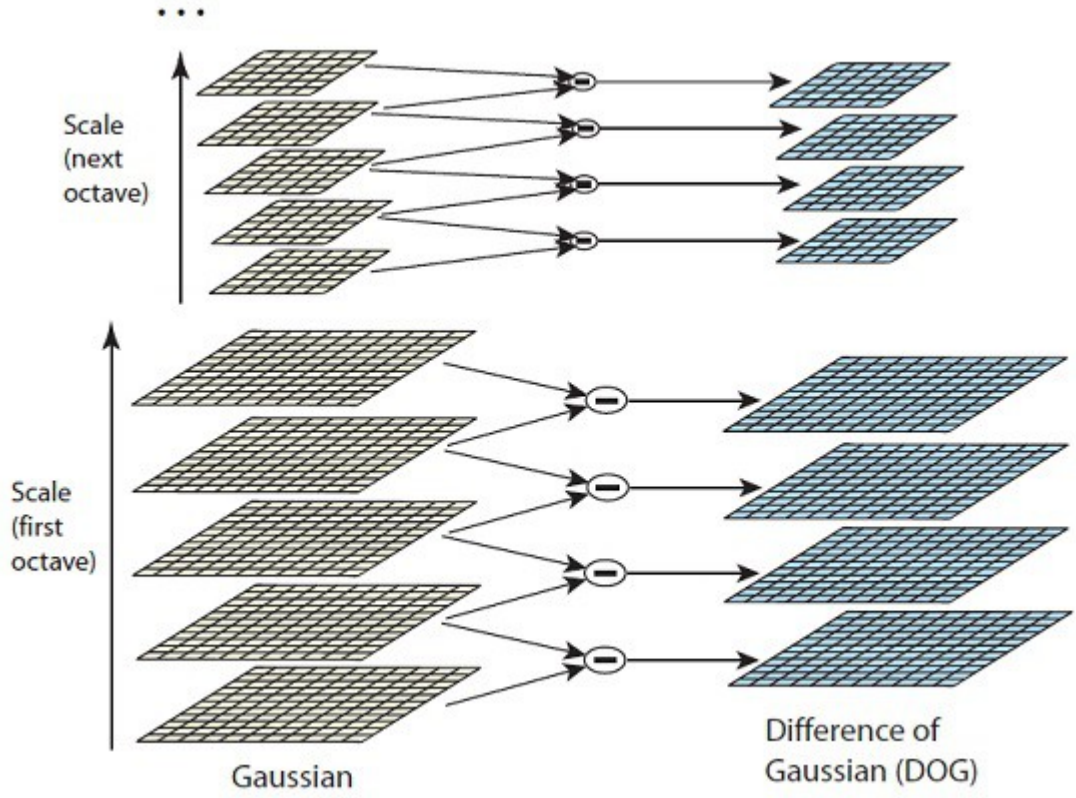


Figura 4: Diferença de Gaussianas no Espaço de Escala (LOWE, 2004)

O próximo passo consiste em encontrar os pontos que são máximos e os pontos que são mínimos nas imagens DoG. Isso é feito pegando cada pixel e checando seu valor com todos os pixeis vizinhos. Essa checagem é feita tanto para os oito pixeis vizinhos do pixel na mesma imagem quanto para com os nove pixeis da imagem da escala superior e os nove pixeis vizinhos da imagem da escala inferior. A Figura 5, ilustra esse procedimento ***X*** representa o pixel da imagem corrente que está sendo analisada, os círculos representam os vizinhos que vão ser comparados. No total são realizadas 26 checagens (8 + 9 + 9) e ***X*** é marcado como candidato a ponto chave se for mínimo ou máximo entre os 26 vizinhos.

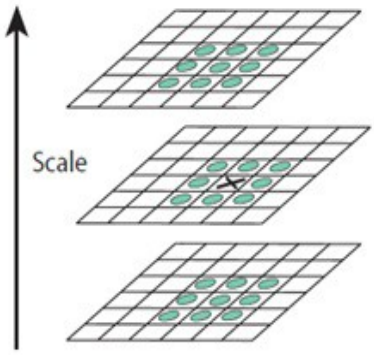


Figura 5: Detecção de Máximos e Mínimos

2.2.4.2 Localização dos Pontos Chaves

A detecção de extremos resulta em vários candidatos a pontos chave, muitos deles se localizam em bordas, ou tem baixo contraste. Nestes casos os candidatos a pontos chave não são úteis como características e são descartados.

Para a análise e possível descarte do candidato a ponto chave é preciso, inicialmente, determinar sua localização exata, visto que muitas vezes, o ponto pode estar localizado entre pixels. Brown, em (BROWN; LOWE, 2002), propôs um método para determinar a localização interpolada do ponto, o qual usa a expansão de Taylor (até o termo quadrático) da função do espaço de escala $D(x, y, \sigma)$ deslocada de modo que a origem é o ponto de amostragem, sendo:

$$D(X) = D + \frac{\partial D^T}{\partial X} X + \frac{1}{2} X^T \frac{\partial^2 D}{\partial X^2} X \tag{14}$$

Onde D e suas derivadas são avaliadas no ponto de amostragem e $X = (x, y, \sigma)^T$ é o deslocamento a partir deste ponto. A localização do extremo $\hat{x}$ é determinado tomando a derivada da função acima em relação a X e igualando a zero, resultando em:

$$\hat{x} = \frac{-\partial^2 D^{-1}}{\partial X^2} \frac{\partial D}{\partial X} \tag{15}$$

O deslocamento final $\hat{x}$ é adicionado a localização do ponto de amostragem para obter a estimativa interpolada da localização do extremo. O valor da função no extremo, $D(\hat{x})$ é útil para rejeitar extremos instáveis com baixo contraste, sendo:

$$D(\hat{x}) = D + \frac{1}{2}\frac{\partial D^T}{\partial X}\hat{x} \tag{16}$$

Segundo experimentos realizados por Lowe, todos os extremos com valor de $|D(\hat{x})|$ menor do que *0.03* devem ser descartados, considerando que os valores dos pixeis estejam normalizados entre *0* e *1*.

Por questão de estabilidade, não é suficiente rejeitar somente pontos com baixo contraste. A função de diferença de gaussiana tem resposta forte ao longo de bordas e pontos localizados em bordas não são desejáveis.

Um pico mal definido na função DoG tem uma grande curvatura principal ao longo da borda mas uma pequena curvatura principal na direção perpendicular. A curvatura principal pode ser calculada a partir de uma matriz hessiana 2x2, H , calculada na localização e escala do ponto chave, sendo:

$$H = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix} \tag{17}$$

As derivadas são estimadas pegando a diferença dos vizinhos do ponto de amostragem. Os autovalores de H são proporcionais a curvatura principal de D e podem ser calculados a partir do traço e do determinante de H . Sendo:

$$Tr(H) = D_{xx} + D_{yy} = \alpha + \beta \tag{18}$$

$$Det(H) = D_{xx}D_{yy} - (D_{xy})^2 = \alpha + \beta \tag{19}$$

Onde

α : autovalor de maior magnitude

β : autovalor de menor magnitude

$Tr(H)$: Traço da matriz hessiana H

$Det(H)$: Determinante da matriz hessiana H

Seja r a razão entre o autovalor de maior magnitude e o autovalor de menor magnitude:

$$r = \alpha / \beta \tag{20}$$

Então:

$$\frac{Tr(H)^2}{Det(H)} = \frac{(\alpha+\beta)^2}{\alpha\beta} = \frac{(r+1)^2}{r} \tag{21}$$

Que depende apenas da razão entre os autovalores ao invés de seus valores. O valor de $(r+1)^2/r$ é mínimo quando os dois autovalores são iguais e ele aumenta em função de r. Portanto para verificarmos se a razão entre as curvaturas principais está abaixo de um determinado limiar, r, é preciso apenas fazer a seguinte checagem:

$$\frac{Tr(H)^2}{Det(H)} < \frac{(r+1)^2}{r} \tag{22}$$

Em seus experimentos, Lowe usa o valor de $r = 10$, que elimina pontos chaves que tem a razão entre as curvaturas principais maior que *10*.

2.2.4.3 Atribuição de orientação

Atribuir uma orientação consistente a cada ponto chave baseada em propriedades locais da imagem permite que o descritor do ponto chave seja representado em relação a essa orientação de maneira a garantir a invariância a rotação da imagem.

A orientação do ponto chave é obtida através do gradiente e da orientação do gradiente na região do ponto chave. Para cada amostra de imagem L(x, y), a magnitude do gradiente, m(x, y), e a orientação do gradiente teta(x, y) são calculadas a partir de diferenças de pixeis:

$$m(x, y) = \sqrt{(L(x+1, y) - L(x-1, y))^2 + (L(x, y+1) - L(x, y-1))^2} \tag{23}$$

$$\theta(x, y) = \tan^{-1}((L(x, y+1) - L(x, y-1))/(L(x+1, y) - L(x-1, y))) \tag{24}$$

Um histograma de orientações é construído a partir das orientações dos gradientes de pontos de amostragem dentro de uma região ao redor do ponto chave. O histograma de orientações tem 36 entradas, cobrindo o intervalo de 360 graus de orientações possíveis. Na

primeira entrada do histograma são adicionadas orientações de 0 a 9 graus, na segunda entrada orientações de 10 a 19 e assim sucessivamente. O valor que é adicionado a cada entrada é ponderado pela magnitude do gradiente e por uma janela gaussiana circular com valor de σ igual a 1.5 vezes o valor do σ da escala do ponto chave.

Picos no histograma de orientações indicam as direções dominantes dos gradientes locais. O pico mais alto do histograma é a orientação do ponto chave em análise. Entretanto se houver picos com valores maior ou igual a 80 % do pico mais alto, devem ser criados múltiplos pontos chaves, que terão mesma localização, mesma escala, porém orientações diferentes.

2.2.4.4 Descritor do ponto chave

As operações até aqui asseguram ao ponto chave uma localização, uma escala e uma orientação. O próximo passo é determinar um descritor para o ponto chave que seja altamente distintivo e invariante a iluminação e ponto de vista 3D.

Para criar o descritor do ponto chave primeiramente é calculado a magnitude e a orientação ao redor da localização do ponto chave, usando a escala do ponto chave. Para garantir a invariância a rotação as coordenadas do descritor e as orientações dos gradientes são rotacionadas em relação à orientação do ponto chave, determinada na sessão anterior.

Uma função gaussiana com σ igual à metade da largura da janela do descritor é usada para dar pesos as magnitudes dos gradientes de cada ponto da amostragem. Isso é feito para dar mais ênfase a gradientes que estão mais próximos do ponto chave. As amostras são então acumuladas em um histograma de orientações com 8 entradas, cada entrada cobrindo 45 graus.

A Figura 6 ilustra um descritor construído a partir de uma janela de amostragem de dimensão 16x16 (esta dimensão foi a que obteve melhores resultados nos experimentos feitos por Lowe). A esquerda o circulo representa o filtro gaussiano e as setas indicam as direções e magnitudes dos gradientes. A janela de dimensão 16x16 é dividida em 16 janelas de dimensão 4x4, e para cada uma dessas janelas menores é criado o histograma de orientações de dimensão 8 (setas da imagem a direita). Dessa forma cada descritor tem 4x4x8 = 128 valores.

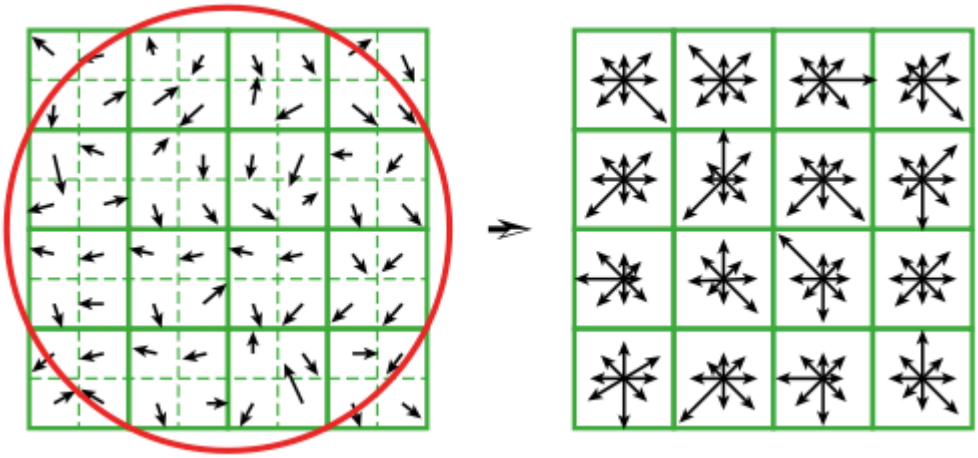


Figura 6: Descritor do Ponto Chave (LOWE, 2004)

Uma vez construído o descritor ele deve ser normalizado (dividindo pela raiz da soma dos quadrados), para diminuir os efeitos causados por possíveis variações de iluminação. Assim, cada imagem terá ao final um conjunto de descritores que são invariantes a escala, rotação, e iluminação.

2.3 Considerações Finais

Neste capítulo, foram apresentados os conceitos e técnicas sobre estruturas de dados métricas e processamento digital de imagens. Na seção estrutura de dados métricas foi apresentado a M-Tree. Em processamento digital de imagens foram a presentados dois potenciais extratores de características de faces, o *eigenfaces* e o SIFT. No capítulo seguinte são abordadas as contribuições deste trabalho: funções de distância que podem ser aplicadas aos descritores e indexação em estrutura de dados métrica.

Capítulo 3

3. Indexação de Faces

A indexação possibilita processar algoritmos de consultar sem que se tenha que realizar verificação do operador de consulta em todos os elementos da base de faces. Como bases de faces podem ser muito grande, o processamento da consulta seria muito lento. A indexação possibilita processar a consulta utilizando uma estrutura hierárquica que evita a verificação do operador de consulta em todos os elementos da base de faces, tornando o processamento mais rápido.

Os operadores de consulta por similaridade são a forma mais adequada de consulta para esse domínio de dados, já que podemos estabelecer uma relação de distâncias na comparação de faces. É importante notar que comparar simplesmente matriz de pixel que forma a imagem não nos dá nada de significativo em termos das suas semelhanças ou diferenças. Duas formas de representação de imagens de faces mais adequadas foram descritas no capítulo anterior: extrator *eigenfaces* e o extrator SIFT (*Scale Invariant Feature Transform*). Ambos extratores geram vetores que representam características da face.

O extrator *eigenfaces* gera apenas um descritor de características, sendo que seu tamanho foi determinado experimentalmente para cada base. A escolha dos autovetores de maiores autovalores que determina a dimensão do espaço de faces e consequentemente a dimensão dos descritores. A quantidade de autovetores escolhidos é incrementada e decrementada até que se obteve o melhor resultado.

O extrator SIFT gera vários descritores com várias características. Utilizou-se nesse trabalho 128 características que é a recomendação do trabalho de Lowe. Esse extrator consome mais espaço de armazenamento que o vetor de características obtido pelo algoritmo *eigenfaces.* No entanto, o extrator SIFT garante o dinamismo da estrutura de dados, visto que não é necessário conhecimento prévio de toda a base de faces (base de treinamento).

A seguir serão apresentadas as funções de distância para cada algoritmo de extração de característica e os experimentos realizados. A função de distância Euclidiana foi usada com o extrator *eigenfaces* e duas funções de distância são usadas para o extrator SIFT: a distância

euclidiana mínima entre os descritores e a média das mínimas distâncias euclidianas entre os descritores.

Para validar a indexação de faces foi utilizada uma base de 400 faces de 40 pessoas obtidas pelo Laboratório Olilivetti da Universidade de Cambridge. Um segundo experimento processou 750 faces de 50 pessoas obtidas do Centro de Processamento de Imagens e Sinal do Instituto de Tecnologia da Georgia. O último experimento usou 9.180 imagens de 102 pessoas do Laboratório de Visão Computacional da Universidade de Columbia.

3.1 Dissimilaridade para extrator *eigenface*

As projeções de imagens de faces de diferentes pessoas ocupam posições diferentes dentro do espaço de faces. As imagens de faces de uma mesma pessoa quando projetadas no espaço de faces tendem a ocupar posições próximas umas das outras. Depois de projetada uma imagem de face, sua posição pode (Figura 7):

1. Estar perto do espaço de faces e perto de uma classe
2. Estar perto do espaço de faces e distante de uma classe
3. Estar distante do espaço de faces e perto de uma classe
4. Estar distante do espaço de faces e distante de uma classe

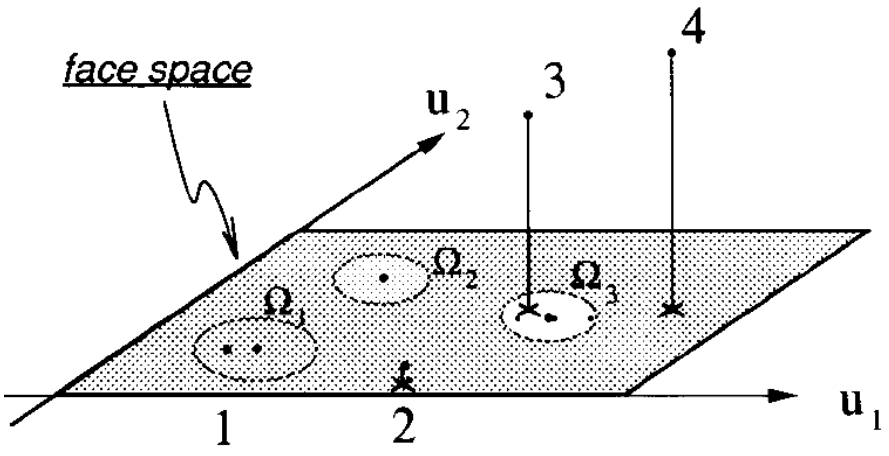


Figura 7: Exemplo espaço de faces (TURK; PENTLAND, 1991)

Para medir a dissimilaridade entre as faces podemos usar a função da distância euclidiana entre dois vetores de características:

$$d_{x,y} = \sqrt{\sum_{k=1}^{K} (x_k - y_k)^2} \tag{25}$$

Onde:

d : Distância euclidiana;

x : Vetor de características de uma imagem de face que se deseja buscar;

y : Vetor de características de uma imagem de face da base de faces;

K : Tamanho do vetor de características, ou seja, o número de componentes escolhidas para construir o espaço de faces.

3.2 Dissimilaridade para o extrator SIFT

O extrator SIFT gera vários descritores com 128 características para cada imagem. Uma forma de medir a dissimilaridade é escolher a menor distância euclidiana entre todos os pares de descritores das duas imagens(BICEGO *et al.*, 2006). Ou seja, dado duas imagens de faces Q e T e seus respectivos conjuntos de descritores:

$X = \{x_1, x_2, \ldots, x_n\}$ onde n = quantidade de descritores da imagem Q .

$Y = \{y_1, y_2, \ldots, y_m\}$ onde m = quantidade de descritores da imagem T .

A função de dissimilaridade $DistEuclMin$ (Distância Euclidiana Mínima), entre X e Y é definida por:

$$DistEuclMin_{(x,y)} = min_{(1<i<m)} \left(min_{(1<j<n)} \left(\sqrt{\sum_{k=1}^{128} (x_{ik} - y_{jk})^2} \right) \right) \tag{26}$$

O algoritmo 3.2.1 a seguir mostra os passos determinar a *DistEuclMin* . São necessários três laços, um laço para percorrer os descritores da primeira imagem (linha 4), um laço para percorrer os descritores da segunda imagem (linha 5) e um terceiro laço para percorrer os valores de cada descritor (linha 6). O terceiro laço calcula a distância euclidiana entre cada par de descritores e armazenada na variável *delta*. A variável *min* armazena a menor das distâncias. A linha 8 representa uma otimização, onde se for atribuído a *delta* um valor maior que *min*, o terceiro laço é interrompido, visto que a distância já não vai mais ser a mínima. Ao fim do terceiro laço é sempre checado se o *delta* representa a distância mínima. E quando chega ao fim dos dois primeiros laços, a função deve retornar a raiz quadrada do valor armazenado na variável *min*.

Algoritmo 3.2.1 *DistEuclMin*(Matriz *M1*, Matriz *M2*)

```
1.  // M1 e M2 : Matrizes de descritores das imagens
2.  min = maior distância possível
3.  delta = 0
4.  para i=0 até i=M1.numlinhas faça
5.    para j=0 até j=M2.numlinhas faça
6.      para k=0 até k=M1.numcolunas faça
7.        delta = delta + ( (M1[i][k] - M2[j][k]) * (M1[i][k] - M2[j][k]) )
8.        se delta > min então
9.          interrompe o laço
10.     fim para
11.     se delta < min então
12.       min = delta
13.   fim para
14. fim para
15. retorna sqrt(min)
```

Ao invés de usar a mínima distância euclidiana entre os descritores, pode-se calcular uma média entre essas mínimas(HUA *et al.*, 2012). Sendo assim a função de dissimilaridade *AvgMinDistEucl* (Média das Minimas Distâncias Euclidiana) é definida por:

$$AvgMinDistEucl_{(x,y)} = \frac{1}{m}\sum_{i=1}^{m}\left(min_{(1<j<n)}\left(\sqrt{\sum_{k=1}^{128}(x_{ik}-y_{jk})^2}\right)\right) \quad (27)$$

Entretanto devem ser tomados alguns cuidados na implementação para garantir a simetria. Por se tratar da média, ela pode variar se pegarmos os descritores em ordem invertida. O algoritmo 3.2.2 descreve como pode ser feita a implementação da *AvgMinDistEucl* , bastante semelhante ao algoritmo 3.2.1, a diferença é que ao fim do segundo laço as distâncias mínimas são somadas para se retornado a média ao final do primeiro laço. Outro ponto importante é a

função *DeterminarOrdemDescritores* que vai determinar a ordem dos laços de maneira a garantir a simetria.

Algoritmo 3.2.2 AvgMin*DistEucl*(Matriz *M1*, Matriz *M2*)

```
1.  // M1 e M2 : Matrizes de descritores das imagens
2.  M1, M2 = DeterminarOrdemDescritores(M1,M2)
3.  somaMin = 0
4.  para i=0 até i=M1.numlinhas faça
5.    min = maior distância possível
6.    para j=0 até j=M2.numlinhas faça
7.      delta = 0
8.      para k=0 até k=M1.numcolunas faça
9.        delta = delta+( (M1[i][k] - M2[j][k]) * (M1[i][k] - M2[j][k]) )
10.       se delta > min então
11.         interrompe o laço
12.     fim para
13.     se delta < min então
14.       min = delta
15.   fim para
16.   somaMin = somaMin + sqrt(min)
17. fim para
18. retorna somaMin / M1.numlinhas
```

O Algoritmo 3.2.3 a seguir mostra uma implementação para a função *DeterminarOrdemDescritores* onde a simetria é garantida ao se pegar em primeiro sempre a imagem que tiver o número maior de descritores, e caso as duas imagens tenham a mesma quantidade de descritores é escolhido aquela cujo elemento numa dada posição seja maior que o da outra na mesma posição.

Algoritmo 3.2.3 ***DeterminarOrdemDescritores***(Matriz *M1*, Matriz *M2*)

```
1.  // M1 e M2 : Matrizes de descritores das imagens
2.  se M1.numlinhas > M2.numlinhas então
3.    retorna M1,M2
4.  senao
5.    se  M1.numlinhas < M2.numlinhas entao
6.      retorna M2,M1
7.    senao // mesma quantidade de descritores
8.      para i=0 até i=M1.numlinhas faça
9.        para j=0 até j=M2.numcolunas faça
10.         se M1[i][j] > M2[i][j] então
11.           retorna M1,M2
12.         senao
13.           retorna M2,M1
14.       fim para
15.     fim para
16.     retorna M1,M2
17.   fim se
18. fim se
```

3.3 Experimentos

Nesta seção é avaliada a indexação de imagens de faces na M-tree usando os descritores apresentados no capítulo anterior com suas respectivas funções de distância.

Para realizar os experimentos foram usadas três bases de faces (com imagens frontais das faces de pessoas) construídas por grupos de pesquisas na área de reconhecimento de faces. Essas bases de faces tem o proposito de avaliar os métodos de reconhecimento de faces.

As máquinas onde foram realizados os experimentos têm as seguintes especificações:

- **Hardware:** Processador Intel Core i7 - 930 - 2.8GHz - Primeira Geração – Sandy Bridge, Memória Principal: 24GB DD3 1600MHz, Placa Mãe Intel DX58SO, Unidade de Armazenamento Samsung SpinPoint SATA2 500GB.
- **Software:** IDE - Netbeans, Linux Kubuntu - kernel 3.0.0-23-generic, Versão do KDE - 4.7.4

3.3.1 Bases de faces usadas no experimento

Três bases de faces são usadas no experimento do Laboratório Olilivetti da Universidade de Cambridge (LOUC), do Centro de Processamento de Imagens e Sinal do Instituto de Tecnologia da Georgia (CPISITG), e do Laboratório de Visão Computacional da Universidade de Columbia (LVCUC).

O conjunto LOUC foi capturado entre abril de 1992 a abril de 1994 no laboratório de pesquisas Olilivetti da universidade de Cambridge contendo 400 faces. São 10 imagens diferentes de 40 pessoas distintas. Para alguns indivíduos, as imagens foram feitas em diferentes momentos, com variação de iluminação, expressões faciais (olhos abertos ou fechados, sorrindo ou não). Toda as imagens são tiradas sob um fundo escuro homogêneo e os indivíduos estão na posição frontal (com pequena tolerância a movimentos de lado). Os arquivos estão no formato PGM. O tamanho de cada imagem é de 92x112 pixels, com 8-bits de níveis de cinza, Figura 8 (SAMARIA; HARTER, 1994).

Figura 8: Exemplos de faces - base LOUC

O conjunto CPISITG contém 750 faces de 50 pessoas obtidas pelo Centro de processamento de Imagens do Instituto de Tecnologia da Georgia. São 15 imagens para cada pessoa, no formato JPEG com fundo variado e dimensão de *640x480*. As expressões faciais são variadas bem como as condições de iluminação e escala, Figura 9 (GEORGIA TECH, 1999).

Figura 9: Exemplos de faces - base CPISITG

O conjunto LVCUC contém 9.180 faces de 102 pessoas populares coletadas da internet, sendo 90 imagens por pessoa. Os nomes das pessoas estão relacionados no Apêndice A. Esse conjunto foi gerado a partir do trabalho(KUMAR *et al.*, 2009), do Laboratório de Visão Computacional da Universidade de Columbia. Diferentemente das bases de faces anteriores, essas faces são tomadas em situação completamente incontroladas. Desta forma existe uma grande variação na pose, iluminação, expressão, ângulo, etc. Figura 10.

Figura 10: Exemplos de faces - base LVCUC

Para obter as faces do conjunto LVCUC foi desenvolvido um programa em Java para gerar a base de faces usando de atributos como URL, nome da pessoa e retângulo da face. Para cada imagem foi recortada a face e salvo em um diretório com o nome da pessoa popular.

Na base de faces resultantes foi realizado um processo de limpeza para apagar as imagens que não representavam faces frontais de pessoas. Em seguida foram apagadas imagens de dimensão inferior a 90x90 pixeis. Para o processo de avaliação é necessário que os diretórios tivessem a mesma quantidade de imagens. Assim, foi estabelecido a quantidade de 90 faces, sendo descartado os diretórios com quantidade inferior. Os diretórios que tinham quantidade superior escolheu-se randomicamente 90 faces, resultando em 9.180 imagens de 102 pessoas

Finalmente imagens de dimensão maior que 200x200 foram redimensionadas para esta dimensão para minimizar o custo computacional.

3.3.2 Classes para integração com o framework obinject

Para indexar as imagens de faces utilizando o framework obinject é preciso seguir a herança de classes criando uma classe *EntitySift* como mostrado na figura 11.

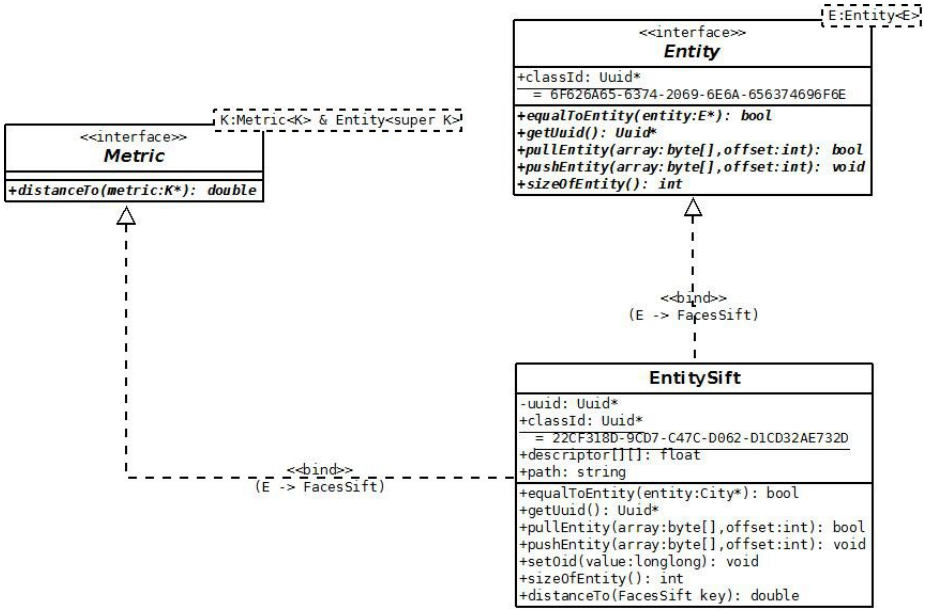


Figura 11: EntitySift no framework obinject

O atributo *descriptor[][]* É responsável por armazenar o conjunto de descritores gerados pelo método SIFT. O atributo *path* armazena uma *String* que contém o endereço da imagem. *O* método *distanceTo()* utiliza a implementação do algoritmo Média das Minimas Distâncias Euclidiana (algoritmo *avgLeastDistance()).*

Para a indexação, deve ser instanciado um objeto da classe *Mtree* o qual através do método *add(K key)* vai adicionar os objetos na estrutura da arvore. Uma vez que os objetos foram indexado é possível agora realizar as buscas por similaridade. As consultas por abrangência e k-vizinhos mais próximos são implementadas respectivamente nas classes *RangeQuery e KnearestNeighbor.*

De maneira semelhante é criada uma classe Entity*Pca*, com a diferença de que o atributo *descriptor[]* nesse caso é unidimensional. Outra diferença está no método *distanceTo()* que implementa o algoritmo da distância Euclidiana.

3.3.3 Medidas de Avaliação

As medidas mais comuns para avaliar a qualidade de um sistema de busca e recuperação de informação são conhecidas como revocação e precisão.

Considere,

I : Uma imagem de face requisitada

R : O conjunto de imagens relevantes, imagens de faces semelhantes a I

A : O conjunto resposta, gerado pela consulta

$R \cap A$: Imagens relevantes na resposta

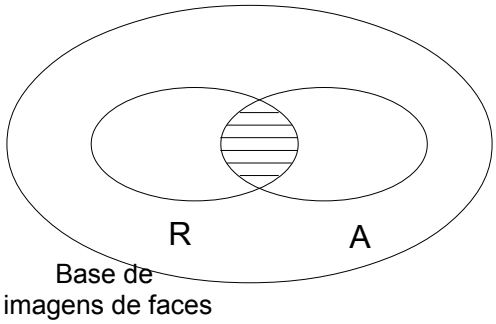


Figura 12: Revocação e Precisão(BAEZA-YATES; RIBEIRO-NETO, 1999)

As medidas de revocação e precisão, são definidas da seguinte maneira (BAEZA-YATES; RIBEIRO-NETO, 1999).

Revocação é a fração de documentos(ou imagens) relevantes(conjunto R) que foram recuperados:

$$Recall = \frac{|R \cap A|}{|R|} \tag{28}$$

Precisão é a fração de documentos(ou imagens) recuperados(conjunto A) que são relevantes:

$$Precision = \frac{|R \cap A|}{|A|} \tag{29}$$

Derivada da precisão e revocação, a Medida-F (*F-measure*, também conhecido como *F-score* ou *F1-score*) é uma outra forma de avaliar a qualidade de um sistema de busca e recuperação. A **Medida-F**: é definida como uma média harmônica entre precisão e revocação. Seu valor varia entre 0 e 1 e quanto mais perto de 1 melhor.

$$F = 2\,x\,\frac{Precision\,x\,Recall}{Precision + Recall} \tag{30}$$

Para a avaliação da precisão, revocação e medida-f, as bases de faces foram organizadas de tal forma que imagens de um mesmo indivíduo se encontram no mesmo diretório, dessa forma o valor de $|R|$ é sempre a quantidade de imagens desse diretório. O valor de $|A|$ é o valor do K de entrada na consulta dos K-Vizinhos mais próximos. O valor de $|R \cap A|$ é obtido contando no conjunto resposta da consulta quais imagens resultantes tem o mesmo caminho de diretório da imagem de consulta.

Nas duas primeiras bases (LOUC e CPISITG) o valor de k variou de 1 até a quantidade de imagens do diretório, sendo o máximo de k=10 para base LOUC e o máximo de K=15 para base CPISITG. Para a base LVCUC foi estabelecido um roteiro de teste com 10 imagens sorteadas aleatoriamente por diretório, sendo assim máximo de k = 10.

Considerando:

- base LOUC: 400 imagens, validação 400 imagens, MAX=10
- base CPISITG: 750 imagens, validação 750 imagens, MAX=15
- base LVCUC: 9180 imagens, validação 1020 imagens, MAX=10
- k: k-vizinhos mais próximos
- MAX: máximo de faces por pessoa
- p: precisão
- r: revocação
- valida[]: imagens validação
- descritor[]: descritor da imagem
- resultado[]: imagens resposta Knn()
- conta: objetos relevantes da resposta
- extrator(): SIFT ou Eigenfaces
- Knn(): Consulta K-Vizinhos mais Próximos
- totalImagem(): quantidade de imagens no diretório
- plotagem(): plota ponto no gráfico

A o fluxograma(Figura 13) a seguir mostra como foram feitos os experimentos.

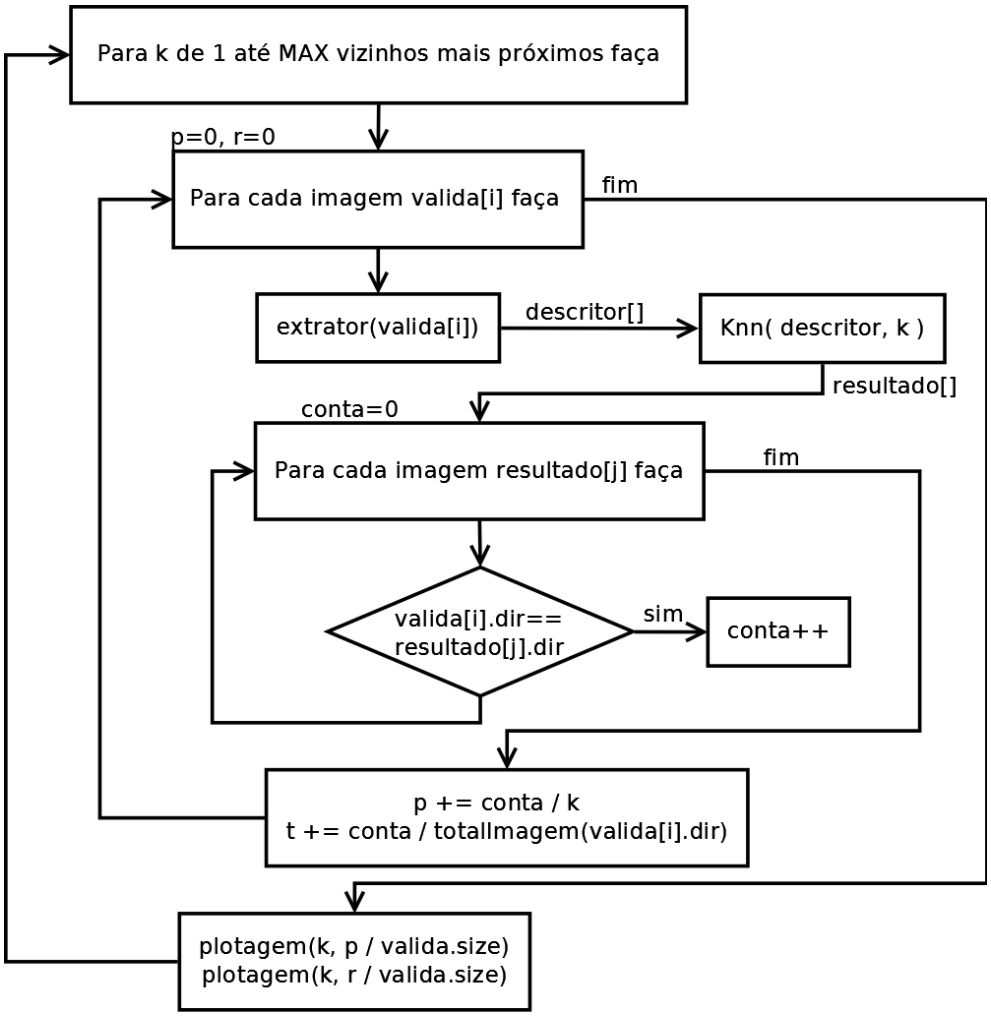
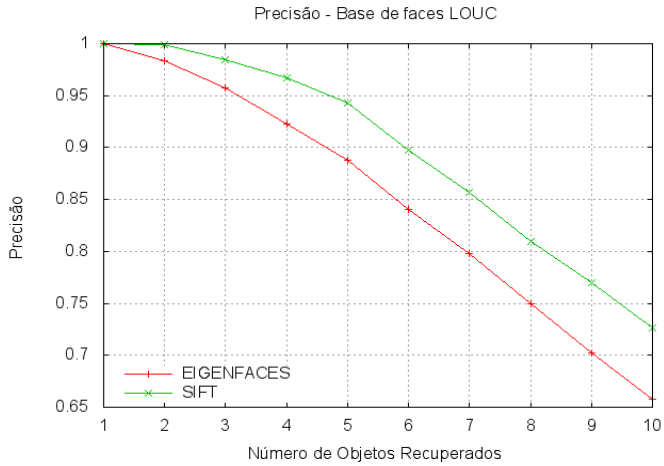
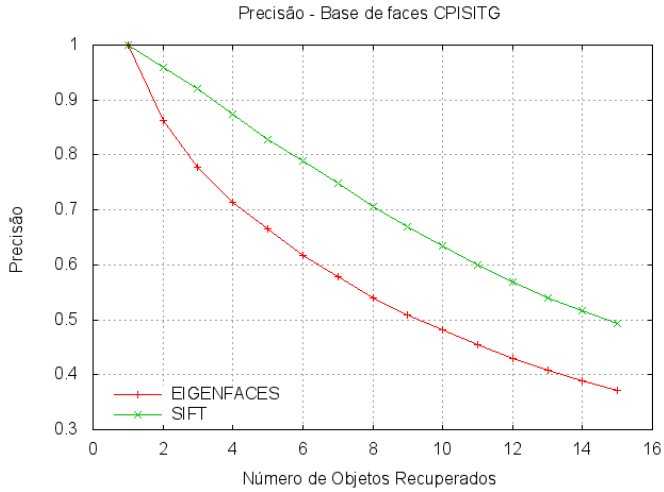


Figura 13: Fluxograma

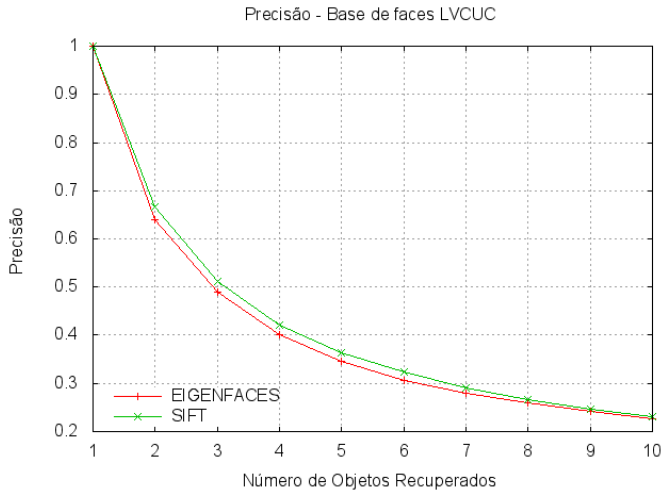
Os resultados desse processo são apresentados nos gráficos da figura 14,figura 15 e figura 16. A figura 14 mostra os gráficos para precisão, enquanto que a figura 15 apresenta os gráficos da revocação. Finalmente na figura 16 são apresentados os gráficos da medida-f.



(a) base de 400 faces

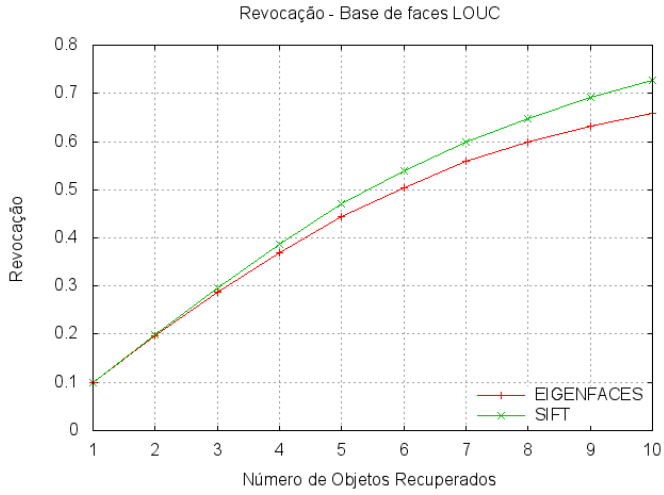


(b) base de 750 faces

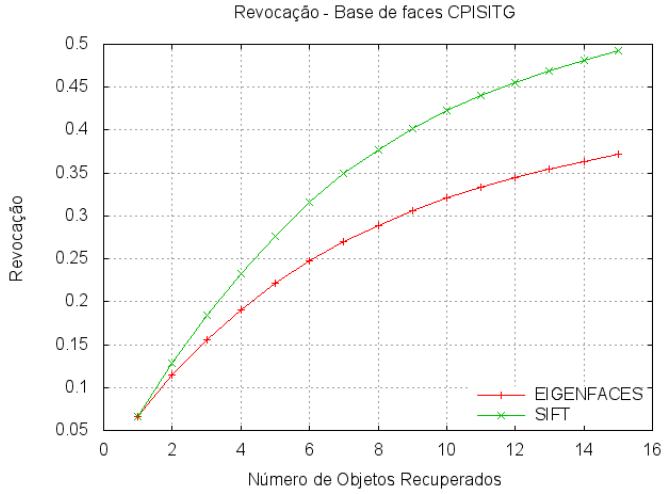


(c) base de 9.180 faces

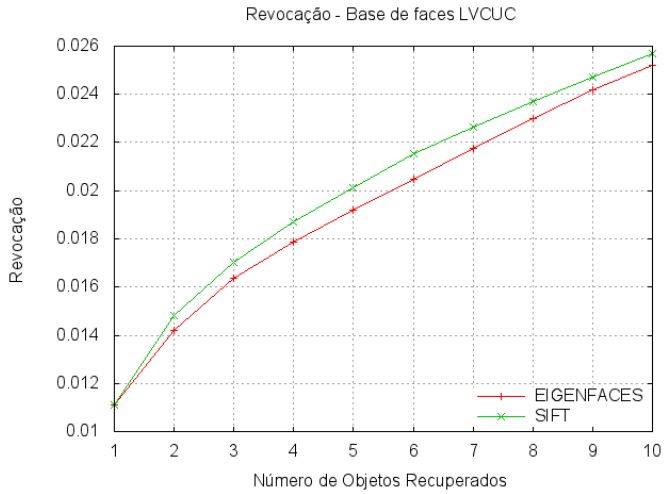
Figura 14: Gráficos valores médios da Precisão para bases LOUC, CPISITG e LVCUC.



(a) base de 400 faces

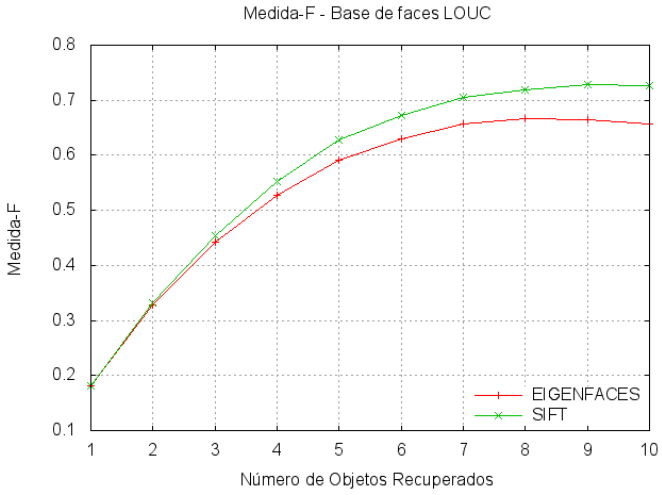


(b) base de 750 faces

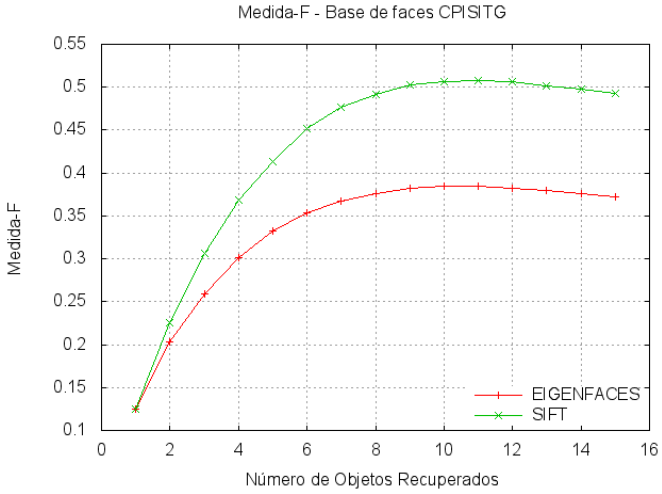


(c) base de 9.180 faces

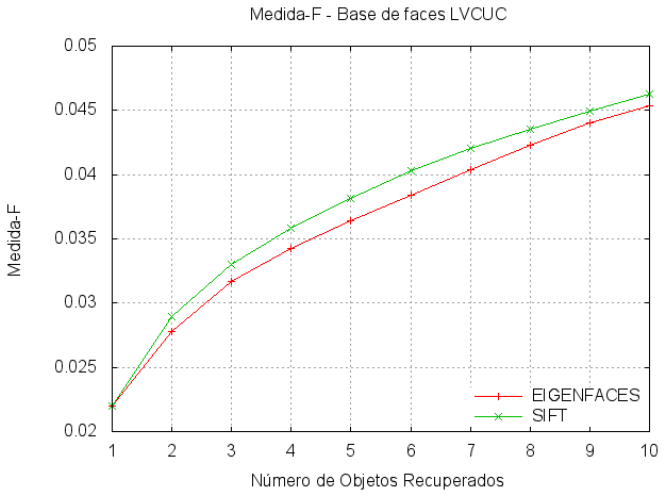
Figura 15: Gráficos valores médios da Revocação para bases LOUC, CPISITG e LVCUC



(a) base de 400 faces



(b) base de 750 faces



(c) base de 9.180 faces

Figura 16: Gráficos valores médios da Medida-F para bases LOUC, CPISITG e LVCUC

Os testes mostraram que em termos de precisão e revocação a melhor combinação foi a função da média das mínimas distâncias euclidianas com os descritores gerados pelo método SIFT. Entretanto as funções sobre tais descritores são mais caras, devido ao fato do número elevado dos descritores (cada um de dimensão 128), e principalmente por terem complexidade cúbica comparado com a complexidade quadrática da função de distância euclidiana entre vetores unidimensionais das características geradas pelo método *eigenfaces*.

Os vetores de características gerados pelo método *eigenfaces* portanto permitem uma indexação e posterior consulta por similaridade mais rápidos, mas a grande limitação é a necessidade de conhecimento prévio de toda a base imagens de face o que deixa a estrutura de indexação estática, sendo necessário a reorganização da árvore, a cada inserção de uma nova imagem de face.

Com relação a precisão a base LVCUC foi a que apresentou menor diferença entre os descritores eigenfaces e SIFT, Figura 14c. Quanto a revocação sob a base CPISITG os descritores SIFT obtiveram os melhores resultados, Figura 15b. E com respeito a medida-f a base LOUC apresentou os melhores resultados uma vez que os valores tendem a se aproximar de 1, Figura 16a.

3.4 Considerações Finais

Foram apresentadas neste capítulo para os descritores SIFT as funções *MinDistEuclMin* e *AvgMinDistEucl*, contudo não faz parte do escopo deste trabalho apresentar a prova matemática de que estas funções sejam realmente métricas. Experimentalmente os conjuntos de testes garantiram as propriedades de simetria, não-negatividade e desigualdade triangular para a função *AvgMinDistEucl*.

No entanto, os mesmos testes mostram que função *MinDistEuclMin* não garante a propriedade da desigualdade triangular, sendo este o motivo de não se utilizar nos experimentos.

Capítulo 4

4. Conclusão

Concluímos neste trabalho que é possível indexar as imagens de faces em estruturas de dados métricas utilizando o descritor gerados pelo extrator *eigenfaces* ou os descritores gerado pelo extrator SIFT.

A função de média das mínimas distâncias euclidianas não tinham sido usadas como métrica para indexar imagens de face em estruturas de dados métricas. Experimentos com as bases mostraram que é uma função métrica. Essa função usando o extrator SIFT mostrou-se melhor em termos de precisão e revocação comparado ao extrator *eigenfaces.*

Ainda como resultado o método *eigenfaces* utiliza uma menor quantidade de descritores que o método SIFT. Contudo, o método *eigenfaces* não garante dinamismo à estrutura de indexação devido à necessidade de recalcular os descritores a cada inserção de uma nova imagem de face considerada como um novo indivíduo.

Como resultado secundário pode-se utilizar a metodologia usada neste trabalho para implementar o módulo Máquina de Busca viabilizando a arquitetura proposta em (PAES; SERAPHIM, 2012).

4.1 Trabalhos Futuros

Um trabalho futuro que pode seguir após a conclusão deste é minimizar o tamanho dos descritores e suas características. O custo computacional está relacionado diretamente ao cálculo das funções de distância no método SIFT que usa descritores e suas características. Técnicas alternativas para minimizar a quantidade características dos descritores, como a PCA-SIFT(KE; SUKTHANKAR, 2004), poderia ser avaliada.

Uma outra técnica que permite gerar descritores locais é o SURF(BAY; TUYTELAARS; GOOL, 2006), que segundo os autores, é mais rápido com relação ao SIFT para gerar os descritores e gera uma quantidade menor de descritores.

Para efeito de comparação de tempo, outras estruturas de dados poderiam ser usadas para a indexação e como a estrutura de dados espacial R-tree(GUTTMAN, 1984) usando as mesmas técnicas apresentadas neste trabalho.

Finalmente, outros algoritmos de extratores poderiam ser usados para verificar se é possível melhorar na qualidade da precisão e revocação.

Referências Bibliográficas

BAEZA-YATES, R. A.; RIBEIRO-NETO, B. *Modern Information Retrieval*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1999.

BAY, H.; TUYTELAARS, T.; GOOL, L. V. Surf: Speeded up robust features. 2006, [S.l: s.n.], 2006. p. 404–417.

BICEGO, M. *et al.* On the Use of SIFT Features for Face Authentication. jun. 2006, [S.l: s.n.], jun. 2006. p. 35.

BROWN, M.; LOWE, D. G. Invariant features from interest point groups. 2002, [S.l: s.n.], 2002. p. 656–665.

CARVALHO, L.; SERAPHIM, E. *ObInject: a NoODMG Persistence and Indexing Framework for Object Injection*. 2012. Universidade Federal de Itajubá (UNIFEI), Instituto de Engenharia de Sistemas e Tecnologias da Informação (IESTI), 2012.

CIACCIA, P.; PATELLA, M.; ZEZULA, P. M-tree: An Efficient Access Method for Similarity Search in Metric Spaces. VLDB '97, 1997, San Francisco, CA, USA. *Anais*... San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997. p. 426–435. Disponível em: <http://dl.acm.org/citation.cfm?id=645923.671005>. Acesso em: 28 jun. 2012.

FALOUTSOS, C. *Searching Multimedia Databases by Content*. [S.l.]: Springer, 1996.

GEORGIA TECH. *Georgia Tech face database*. . [S.l: s.n.]. Disponível em: <http://www.anefian.com/research/face_reco.htm>. , 15 nov. 1999

GONZALEZ, R.; WOODS, R. *Processamento de Imagens Digitais*. [S.l.]: Edgard Blucher, [S.d.].

GUTTMAN, A. R-trees: A Dynamic Index Structure for Spatial Searching. 1984, [S.l.]: ACM, 1984. p. 47–57.

HUA, S. *et al.* Similarity measure for image resizing using SIFT feature. *EURASIP Journal on Image and Video Processing*, v. 2012, n. 1, p. 6, 23 abr. 2012. Acesso em: 13 ago. 2012.

JAFRI, R.; ARABNIA, H. R. *A Survey of Face Recognition Techniques*. [S.l: s.n.], [S.d.].

KE, Y.; SUKTHANKAR, R. PCA-SIFT: a more distinctive representation for local image descriptors. CVPR'04, 2004, Washington, DC, USA. *Anais*... Washington, DC, USA: IEEE Computer Society, 2004. p. 506–513. Disponível em: <http://dl.acm.org/citation.cfm?id=1896300.1896374>. Acesso em: 29 ago. 2012.

KOBAYASHI, M.; TAKEDA, K. Information retrieval on the web. *ACM Comput. Surv.*, v. 32, n. 2, p. 144–173, jun. 2000. Acesso em: 29 jun. 2012.

KUMAR, N. *et al.* Attribute and Simile Classifiers for Face Verification. 2009, [S.l: s.n.], 2009.

LOWE, D. G. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vision*, v. 60, n. 2, p. 91–110, nov. 2004. Acesso em: 28 jun. 2012.

LOWE, D. G. Object Recognition from Local Scale-Invariant Features. ICCV '99, 1999, Washington, DC, USA. *Anais*... Washington, DC, USA: IEEE Computer Society, 1999. p. 1150–. Disponível em: <http://dl.acm.org/citation.cfm?id=850924.851523>. Acesso em: 28 jun. 2012.

PAES, V.; SERAPHIM, E. *Crawler de Faces na Web*. 2012. Universidade Federal de Itajubá (Unifei)), Instituto de Engenharia de Sistemas e Tecnologias da Informação (IESTI), 2012.

PATELLA, M. *Similarity Search in Multimedia Databases*. 1999. Università degli Studi di Bologna, Dipartimento di Elettronica Informatica e Sistemistica, Bologna, Italy, 1999.

SAMARIA, F. S.; HARTER, A. C. Parameterisation of a stochastic model for human face identification. In: , PROCEEDINGS OF THE SECOND IEEE WORKSHOP ON APPLICATIONS OF COMPUTER VISION, 1994, dez. 1994, [S.l: s.n.], dez. 1994. p. 138 -142.

SILVA, P. *Pesquisa de Imagens de Rosto*. 2009. Universidade Nova de Lisboa, Faculdade de Ciências e Tecnologia, Departamento de Informática, Lisboa, Portugal, 2009.

TURK, M. A.; PENTLAND, A. P. Face recognition using eigenfaces. jun. 1991, [S.l: s.n.], jun. 1991. p. 586 -591.

Apêndice A: Base de Pessoas Populares

Nomes das 102 pessoas populares obtidas da base do Laboratório de Visão Computacional da Universidade de Columbia.

Aaron Eckhart	Daniel Radcliffe	Jessica Alba	Nicolas Cage
Adam Sandler	David Beckham	Jessica Simpson	Nicole Kidman
Adriana Lima	Denzel Washington	Jimmy Carter	Oprah Winfrey
Alberto Gonzales	Donald Trump	Joaquin Phoenix	Orlando Bloom
Alec Baldwin	Drew Barrymore	Jodie Foster	Owen Wilson
Alicia Keys	Dustin Hoffman	John Travolta	Reese Witherspoon
Angela Merkel	Eliot Spitzer	Kate Moss	Renee Zellweger
Angelina Jolie	Eliza Dushku	Kate Winslet	Ricky Martin
Anna Kournikova	Eva Mendes	Katie Couric	Rosario Dawson
Antonio Banderas	Gael Garcia Bernal	Keanu Reeves	Russell Crowe
Ashley Judd	George Clooney	Keira Knightley	Salma Hayek
Ashton Kutcher	George W Bush	Lance Armstrong	Shania Twain
Avril Lavigne	Gillian Anderson	Leonardo DiCaprio	Sharon Stone
Ben Affleck	Gisele Bundchen	Liam Neeson	Silvio Berlusconi
Beyonce Knowles	Gordon Brown	Lindsay Lohan	Simon Cowell
Bill Clinton	Gwyneth Paltrow	Liv Tyler	Steven Spielberg
Brad Pitt	Halle Berry	Lucy Liu	Susan Sarandon
Brendan Fraser	Harrison Ford	Mariah Carey	Tina Fey
Bruce Willis	Hugh Grant	Martha Stewart	Tom Cruise
Cameron Diaz	Jack Nicholson	Matt Damon	Tom Hanks
Cate Blanchett	James Franco	Mel Gibson	Tyra Banks
Celine Dion	Jason Statham	Meryl Streep	Uma Thurman
Charlize Theron	Jay Leno	Michael Bloomberg	Victoria Beckham
Cindy Crawford	Jennifer Aniston	Michael Douglas	Will Smith
Colin Farrell	Jennifer Lopez	Monica Bellucci	
Colin Powell	Jennifer Love Hewitt	Morgan Freeman	