Universidade da Beira Interior
Departamento de Informática

Detecção de faces humanas em tempo real

Silvio Brás Filipe, Nº17752
Licenciatura em Engenharia Informática

Orientador do Projecto: **Prof. Doutor Hugo Proença**

Covilhã, Julho de 2008

Agradecimentos

Esta é uma parte importante do relatório, uma vez que tenho oportunidade para agradecer às pessoas que de uma forma ou outra foram imprescindíveis para a execução do mesmo.

Em primeiro lugar, gostaria de agradecer ao meu orientador de projecto, o Professor Doutor Hugo Proença, por toda a informação que me forneceu, pela forma clara e precisa com que me transmitiu a sabedoria e conhecimento científico, pela sua disponibilidade de tempo e acima de tudo pelos seus ensinamentos e conselhos, que penso que foram cruciais para a realização deste projecto.

Quero também agradecer à minha família, em especial aos meus pais, pois sempre me apoiaram e ajudaram ao longo desta etapa da minha vida.

Quero agradecer a todos os membros e colaboradores do grupo *Soft Computing and Image Analysis Laboratory* (SOCIA Lab). pelo óptimo espírito de equipa e pela grande inter ajuda que existia entre nós dentro do laboratório.

Por fim, mas não menos importante à minha querida namorada, por todo o apoio e força que me deu o longo deste tempo, e à qual queria pedir desculpa pelos dias em que a troquei pelo computador.

Obrigado!

Conteúdo

Lista de Tabelas

Lista de Figuras

Lista de Algoritmos

Acrónimos

ANSI *American National Standards Institute*

CIE Lab *Commission Internationale dÉclairage Lab*

CImg Lib *The C++ Template Image Processing Library*

CMU *Carnegie Mellon University*

EER *Equal Error Rate*

FNR *False Negative Rate*

FPR *False Positive Rate*

GCC *GNU Compiler Collection*

ICSP'08 *9th International Conference on Signal Processing*

IEEE *Institute of Electrical and Electronics Engineers*

MIT *Massachusetts Institute of Technology*

MLP *Multi Layer Perceptron*

ROC *Receiver Operating Characteristic*

SOCIA Lab *Soft Computing and Image Analysis Laboratory*

Glossário

***Equal Error Rate* (EER) -** é a taxa de erro para a verifição do funcionamento de um sistema para se ajustar o limiar para a aceitação / rejeição da decisão, é adaptada de tal forma a que as taxas de falsos positivos e falsos negativos se tornam iguais.

***True Negative Rate* (TNR) -** ou taxa de verdadeiros negativos é a taxa de não faces detectadas como tal.

***False Positive Rate* (FPR) -** ou taxa de falsos positivos é a taxa de faces erradamente detectadas.

***False Negative Rate* (FNR) -** ou taxa de falsos negativos é a taxa de faces não detectadas.

***True Positive Rate* (TPR) -** ou taxa de verdadeiros positivos é a taxa de faces detectadas como tal.

***Receiver Operating Characteristic* (ROC) -** ou simplesmente curva ROC é a representação gráfica da taxa de verdadeiros positivos com a taxa de falsos positivos.

Capítulo 1

Introdução

O objectivo deste projecto é o desenvolvimento de um sistema de detecção de faces humanas capaz de dar resposta em tempo real. No domínio dos sistemas biométricos, a detecção de faces humanas em imagens / sequências de vídeo revela-se uma tarefa crucial, uma vez que irá dar suporte a todas as etapas posteriores, nomeadamente as de segmentação, normalização e reconhecimento da face, ou até mesmo da íris. Este projecto enquadra-se no âmbito do projecto PTDC/EIA/69106/2006, "BIOREC: Reconhecimento Biométrico Não-Cooperativo", financiado pela FCT/FEDER.

1.1 Motivação

Desde muito cedo que o ser humano tenta desenvolver a sua tecnologia de modo a torná-la extensão, ou mesmo substituta, do seu próprio corpo. A ideia de máquinas e seres animados automatizados habitam o imaginário humano, desde longa data.

Inserido num processo de verosimilhança e aperfeiçoamento da máquina humana, o estudo da visão exerce um grande fascínio. Neste ponto, o interesse está não só na percepção do estímulo luminoso, mas também, no reconhecimento lógico dos objectos circundantes. Muitos são os animais dotados de visão, sendo este, sistemas mais simples ou complexos mais

específicos ou gerais. A visão é usada para inúmeras tarefas, muitas delas relacionadas com a sobrevivência e perpetuação da espécie. Os sistemas de visão de mamíferos superiores, como os primatas, apresentam grande complexidade e mecanismos sofisticados de reconhecimento de padrões.

Além do interesse pela visão artificial, os processos automatizados de reconhecimento de padrões ganham importância frente à crescente oposição entre a velocidade da informação e a limitação temporal. Esses processos requerem eficiência, eficácia e agilidade crescentes. Nesse sentido, as tarefas visuais automatizadas tornam-se excelentes soluções, especialmente pelo suporte conferido e pelo avanço da computação e dos sistemas automáticos.

Dentro deste contexto, o processamento de faces humanas em cenas tem crescido significativamente em importância nos últimos anos. O propósito dos investigadores não está voltado apenas para a detecção da face como um todo, mas também, das suas características intrínsecas (olhos, nariz e boca).

Os frutos da pesquisa nessa área do conhecimento são compartilhados pelo campo científico e comercial. Em relação ao primeiro caso, o treino dos computadores para analisar faces tenta melhorar a interacção entre máquinas e humanos. Sob o aspecto financeiro, a aplicação crescente do mesmo ocorre em sistemas de segurança e vigilância, motivada pela violência social, tráfico e terrorismo.

Na simulação do comportamento visual humano, denominado genericamente por Visão Computacional, o reconhecimento do padrão facial de uma pessoa constitui uma tarefa desafiadora. Neste contexto, o processo de treino do reconhecimento é meticuloso e segmentado em etapas bem definidas. É necessário detectar as faces e, em seguida, segmentá-las, por forma que o reconhecimento possa ser mais preciso e rápido.

A comunidade científica tem publicado grande número de trabalhos nesta área. A biometria de faces humanas, ou seja, a técnica de medidas de características intrínsecas à face é, actualmente, uma das frentes de pesquisa

na área da Visão Computacional.

A biometria consiste na medida de características individuais, tais como: impressões digitais ou proporções entre características que podem identificar univocamente uma pessoa. Assim, a biometria é uma palavra-chave que não pode ser esquecida, perdida ou roubada [10].

A indexação e a recuperação de imagens contendo actividades humanas, as técnicas de análise de expressões faciais, as actividades computacionais em tempo real, como a visão de robôs, podem incluir de técnicas de detecção de faces humanas para obter respostas rápidas e tomar decisões num curto espaço de tempo. Da mesma forma, sistemas de segurança e vigilância, de identificação pessoal também requerem técnicas biométricas.

1.2 Abordagem

Para o tema proposto, utilizou-se a abordagem descrita por Viola e Jones (2004) [24], devido à sua elevada fiabilidade, tanto em termos de tempo de computação como de taxas de erro.

A esta abordagem foi adicionado um conjunto de novas características por forma a verificar se o erro deste método ainda poderia ser diminuído.

O novo conjunto de características são baseadas em triângulos, visto que os autores apenas tinham explorado características baseadas em quadrados. O motivo destas características terem sido adicionadas foi que, na área da computação gráfica todos os objectos 3D podem ser representados por malhas de triângulos. Com base nesta ideia achou-se que seria uma boa abordagem tentar convertê-las para 2D e aplicá-las ao problema da detecção de faces humanas.

1.3 Organização do relatório

Este relatório encontra-se dividido por capítulos. No primeiro capítulo optou-se por fazer uma introdução ao tema, frisando vários aspectos com ele relacionados.

Capítulo 2 - Métodos de detecção de faces em imagens digitais Neste capítulo serão apresentados alguns métodos que existem para a detecção de faces, sendo que foram divididos em grupos, dependendo dos procedimentos e tipos de imagem usadas.

Capítulo 3 - Método Proposto Apresentação do método que foi usado na elaboração deste projecto e alterações efectuadas para a obtenção de melhores resultados.

Capítulo 4 - Experiências e Resultados Neste capítulo descreve-se as experiências realizadas, bem como os resultados obtidos pelos diversos métodos.

Capítulo 5 - Conclusão e trabalho futuro Como o próprio nome indica, pretende-se retirar algumas conclusões/argumentos sobre a realização deste projecto bem como descrever um pouco o que poderá ser acrescentado a este trabalho na tentativa da obtenção de ainda melhores resultados.

Capítulo 2

Métodos de detecção de faces humanas em imagens digitais

Na tarefa de processamento de faces humanas, o problema de detecção de faces é um dos mais importantes a serem solucionados. A detecção de faces merece especial estudo, uma vez que é o pré-processamento necessário para as áreas de reconhecimento automático e análise de expressões faciais.

A localização de faces humanas em imagens digitais é uma tarefa importante em diversas aplicações. A indexação e a recuperação de imagens de vídeo contendo actividades humanas requerem a detecção automática da localização das faces dentro da cena. As técnicas que reconhecem faces ou analisam expressões faciais também requerem conhecimento sobre a sua localização dentro da imagem. Aplicações em tempo real, tal como a visão de robôs, devido à necessidade de resposta rápida, podem utilizar técnicas de detecção de faces humanas para que se possam tomar decisões num curto espaço de tempo.

De início, temos duas abordagens principais para o problema de detecção de faces. Na primeira, a face é tratada como um todo e usado um modelo geral, representa os principais traços da face fazendo-se uma aproximação para a face que se deseja localizar. Esse modelo pode ser estático (quando as informações são conhecidas antecipadamente, por exemplo: o tamanho aproximado das faces ou o número de faces presentes na cena) ou dinâmico

(métodos mais genéricos em que nenhuma informação a respeito da cena é previamente conhecida). Na segunda abordagem, a face é localizada através de alguns dos seus componentes, tais como olhos, boca e nariz. Da mesma forma a disponibilidade das informações prévias definirá uma estratégia mais específica ou genérica.

Existem vários factores que tornam a detecção de faces mais complexa, tais como, cabelo na face, maquilhagem, barba ou bigode, o uso de óculos ou chapéus, factores que podem esconder as características faciais. Outro problema é a escala e a orientação da face na imagem, pois isso dificulta a utilização de modelos de faces fixos para encontrar as características. A presença de ruídos e oclusões constitui outro tipo de problema que também ocorre [21].

2.1 Alguns métodos para detecção de faces humanas

Os métodos de detecção de faces humanas baseiam-se em algumas informações prévias. Alguns métodos de detecção de faces usam a informação de imagens em nível de cinza [20], [21] e [24]. Outros métodos utilizam as informações das arestas contidas numa imagem [26]. Existe um número considerável de métodos que usam a informação de cor para detectar faces [4], [6], [9] e [17]. Também existem métodos que utilizam a informação da geometria da face para detectar faces [16], [18] e [29].

2.1.1 Escala de cinza

Os métodos baseados em escala de cinza usam características pré-definidas da imagem, tanto para treinar o sistema como para criar o modelo.

O método desenvolvido por Rowley et al. (1998) examina pequenas subjanelas da imagem e decide que cada janela contém uma face [20], usando

redes neuronais artificiais. Ao ser usado o algoritmo de *bootstrap* durante o treino da rede neuronal, este irá adiciona imagens que não contêm faces no conjunto de treino, eliminando assim a difícil tarefa manual de seleccionar os exemplos de não caras no treino. O sistema pode detectar entre 77.9% e 90.3% das faces num conjunto de 130 imagens, com um número aceitável de falsas detecções. Embora projectado para detectar faces frontais, a rede pode ser também treinada para detectar faces de perfil.

Alguns resultados obtidos pelo método de Rowley et al. (1998) são apresentados na figura 2.1. No canto superior esquerdo de cada imagem, podem-se observar três números, sendo estes o número de faces contidas na imagem, o número de faces detectadas correctamente e o número de falsas detecções, respectivamente. Na imagem A, todas as faces foram detectadas, mas o sistema apresentou uma falsa detecção. Na imagem B, as faces foram detectadas devido à oclusão de uma face, numa outra, existe um grande ângulo da face em relação à imagem frontal. Em C, o desenho no canto superior direito não foi detectado pelas redes neuronais. Na imagem D, a face foi detectada, mas o método apresentou uma falsa detecção na região do pescoço. Embora o sistema tenha sido treinado somente em faces reais, alguns desenhos de faces são detectadas na imagem C e na imagem E.

Sung e Poggio (1998) apresentaram uma abordagem para o problema de detecção de faces humanas frontais em cenas complexas [21], através de uma aprendizagem baseada em modelos. Em cada posição da imagem, um vector de características é processado entre o modelo local da imagem e o modelo de distribuição. Um classificador treinado determina, baseado nas medidas do vector de características, se existe ou não uma face humana na posição actual da imagem. Estes autores testaram este sistema em duas bases de dados, onde processaram o número de detecções correctas e os falsos alarmes. A primeira base de dados é constituída por 301 imagens frontais e quase frontais de 71 pessoas diferentes. A segunda base de dados de 23 imagens, contem um total de 149 modelos de faces. Para a primeira base de dados, o sistema encontrou 96.3% de todos os modelos de faces e retornou três falsas detecções, utilizando neste teste uma rede neuronal *Multi Layer Perceptron* (MLP) como classificador. Na segunda, o sistema conseguiu uma taxa de 79.9% de detecções e cinco falsos positivos, utili-

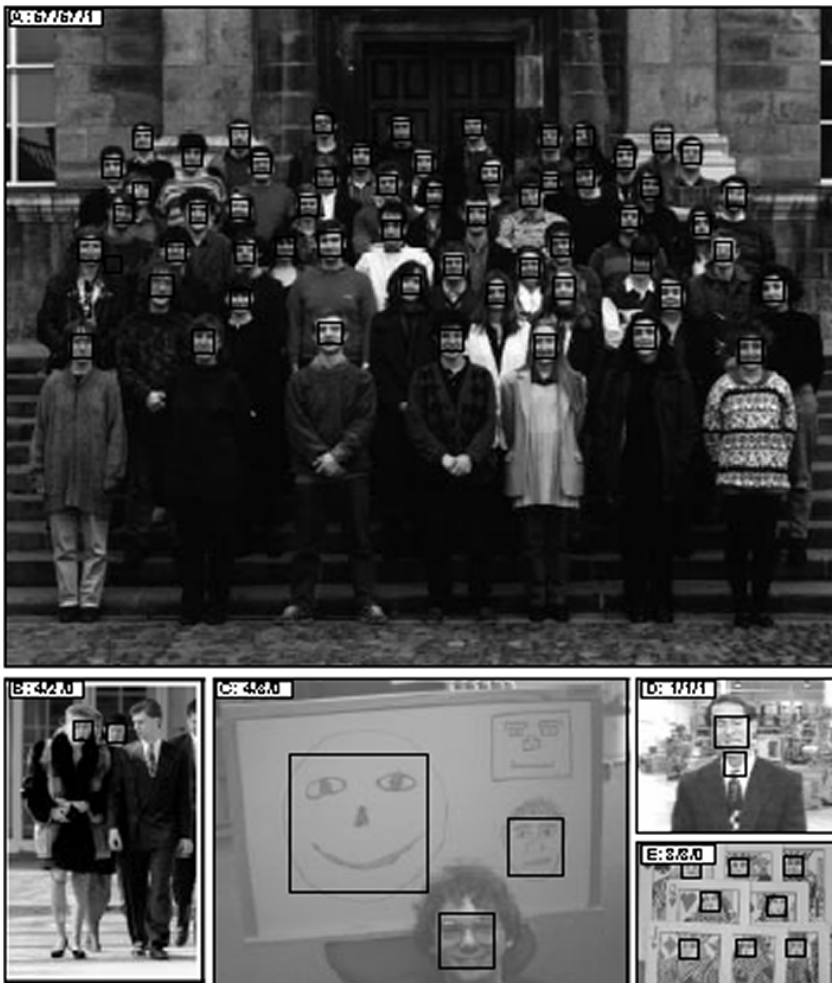


Figura 2.1: *Resultados obtidos por Rowley et al. (1998).*

zando a mesma rede neuronal.

Na figura 2.2, ilustra-se algumas imagens utilizadas no teste do sistema de Sung e Poggio (1998). Na imagem 1, o sistema detectou todas as faces com quatro falsas detecções. Na imagem 2, o sistema errou na detecção de uma face e as outras faces foram todas detectadas, com uma falsa detecção. Na imagem 3, ocorreram três erros devido à sombra.

Figura 2.2: *Resultados obtidos por Sung e Poggio (1998).*

Os métodos de Viola e Jones (2004) são sistemas de detecção de faces capaz de processar imagens de uma forma rápida [24], alcançando uma taxa de detecção bastante aceitáveis. Este método é um dos mais usados e citados pela comunidade cientifica.

Há três contribuições fundamentais a introdução de uma nova representação de imagem, chamada imagem integral, que permite que as características usadas pelo detector sejam processada através de uma única passagem pela imagem. A segunda é um classificador simples e eficiente que é construído usando o *AdaBoost*. Este algoritmo selecciona um pequeno número de características visuais críticas de um conjunto muito grande de potenciais características. A terceira consiste num método para combinar os classificadores numa cascata, que permite descartar as regiões de fundo da imagem de forma rápida, focando-se assim apenas nas regiões de maior interesse.

Estes autores descrevem dois tipos de classificadores. O primeiro é um classificador de uma única camada, fazendo com que não rejeite nenhuma sub-janela durante todo o processo de classificação, isto é, todas as sub-janelas serão processadas por todas as características seleccionadas pelo

algoritmo de treino.

O segundo classificador é baseado numa cascata, o que levará a que as sub-janelas classificadas como não faces, processadas numa dada camada da cascata, sejam logo retiradas do processo de classificação. As sub-janelas com possíveis faces irão passar para o patamar a seguir.

No primeiro classificador, os autores obtiveram uma taxa de 5% de erro para um classificador com 200 características e um tempo de processamento de 0.7 segundos em imagens de 384 x 288 *pixels* num Pentium III a 700 MHz. Para o segundo classificador, o erro aumentou um pouco, devido às imagens que eram faces, e que foram sendo rejeitadas ao longo da cascata, obtendo-se mesmo assim uma taxa de sucesso de 79.9%, para uma cascata com 38 patamares usando 6060 características. No entanto o tempo de processamento passou para 0.067 segundos no mesmo computador.

Figura 2.3: *Resultados obtidos usando um conjunto de teste com imagens da base de dados do MIT e CMU.*

2.1.2 Arestas

Para detectar faces humanas através do mapa de arestas, primeiramente terá que se detectar as arestas e, a seguir, relacioná-las com um modelo de face para verificar as correctas detecções [15].

Wang e Tan (2000) desenvolveram um método baseado na informação do formato da face [26]. A imagem de entrada é realçada por meio da equalização do histograma, de seguida, é realiza uma detecção de arestas utilizando o filtro da mediana. As arestas extraídas são ligadas, usando um método baseado numa função de energia. O contorno da face é depois extraído utilizando a informação da direcção da ligação das arestas. O algoritmo foi testado usando a base de dados do MIT. Esta contém fotografias de 16 pessoas, às quais foram tiradas 27 fotografias, dentro de um escritório, com várias orientações, iluminações e zoom da máquina fotográfica. A resolução é de 128 X 120 e o fundo das imagens é simples, pode ver exemplos desses testes na figura 2.4. Foi também usado um outro conjunto, de 90 imagens com fundo considerado complexo. Estas imagens provêm da base de dados CMU, contendo fotografias tiradas no escritório e digitalizadas, com apenas uma face em cada imagem.

Figura 2.4: *Resultados das detecções na base de dados MIT.*

Para o teste foram criados dois subconjuntos: o primeiro contendo apenas 50 imagens com um fundo simples e o outro 40 imagens com um fundo complexo. No primeiro conjunto foram detectadas 100% das faces, mas 16% destas com inexactidão. Isto acontece devido ao facto das faces estarem inclinadas. O número de falsas caras detectadas foi de 0%. Alguns

exemplos com fundos simples são mostrados na figura 2.5. No segundo conjunto, foram detectadas 87.5% das faces; 12.5% são consideradas inexactas e 12.5% falsas detecções.

Figura 2.5: *Detecções nas imagens com fundos simples.*

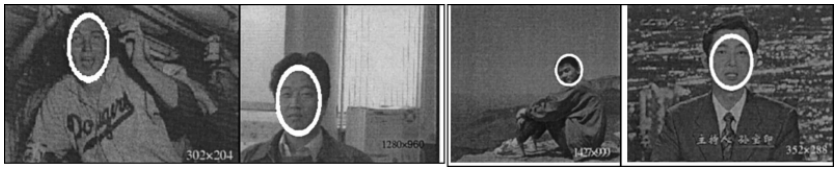


Figura 2.6: *Detecções nas imagens com fundo complexo.*

2.1.3 Cor

Um número considerável de técnicas utiliza a informação da cor para detectar faces. Essas técnicas, primeiro, seleccionam as regiões de imagem mais prováveis de serem faces e só depois as tentam detectar nas regiões seleccionadas, usando padrões faciais.

Daí e Nakano (1996) preferiram isolar a região próxima do laranja no espaço de cor YIQ como região semelhante à pele humana e eliminaram as regiões remanescentes [9]. A partir daí, empregaram características de textura em imagens com nível de cinza para identificar faces nas regiões da pele. No primeiro teste foram seleccionadas aleatoriamente 10 pessoas diferentes numa base de dados de faces e foram realizados um conjunto de cinco ou seis testes por pessoa. Nestes seus testes, estavam incluídas faces com rotação, inclinação e diferentes expressões. As taxas de correctas detecções verificadas foram de 98%. Contudo, este sistema não pode

detectar faces com oclusão parcial, faces usando óculos e faces de perfil.

O algoritmo proposto por Cai e Goshtasby (1999) [4] realiza a detecção das possíveis faces através de processamentos realizados no espaço de color da *Commission Internationale dÉclairage Lab* (CIE Lab). Transformam cada cor no nível de cinza correspondente e utilizam essa informação com uma função de distribuição de probabilidades para determinar as regiões que podem ser faces humanas numa imagem. A partir daí, empregam um modelo de face para a identificação final. Nestes testes, realizados pelos autores, quando foi usado um limiar de 0.5, 13% das faces foram perdidas e 8.7% das faces detectadas de forma errada não eram faces.

O método proposto por Yachida et al. (1999) [6] descreve um método para detectar faces em imagens coloridas, baseado na teoria *fuzzy*. Esse método trabalha com dois modelos *fuzzy*: um para descrever a cor da pele e outro para descrever a cor do cabelo, utilizando um espaço de cor percentual para aumentar a precisão do método. Foi criado um modelo para extrair as regiões da cor da pele e outro para extrair as regiões da cor do cabelo. Comparando estes dois modelos com um modelo de *head-shape*, e utilizando o método de fusão do modelo baseado na teoria *fuzzy*, tentam-se detectar as faces candidatas. No teste realizado pelos autores foi usada uma base de dados com 233 faces, em que 186 eram faces asiáticas e as outras caucasianas. O tamanho das faces vária entre 20 x 24 a 200 x 240 *pixels*. O índice de acerto nas detecções foi de 97% em imagens com tamanho de faces superiores a 50 x 60 *pixels*. As falhas desse método ocorrem devido a factores, tais como: a variação da iluminação, a oclusão facial, as faces adjacentes (se duas ou mais caras estiverem muito próximas, os modelos que descrevem a cor da pele e do cabelo podem ser fundidos, resultando numa forma bem diferente de uma única cabeça) e o estilo do cabelo (por exemplo, faces com um penteado especial, faces de pessoas carecas ou usando chapéus). Alguns resultados obtidos pelo método de Yachida et al. (1999) são apresentados na figura 2.7.

O método apresentado por Kim et al. (2000) é um método de detecção de faces humanas baseado num objecto [17]. Esse método possui dois passos: segmentação e detecção da região facial. No primeiro passo, a imagem de entrada é segmentada por um algoritmo genético dentro de

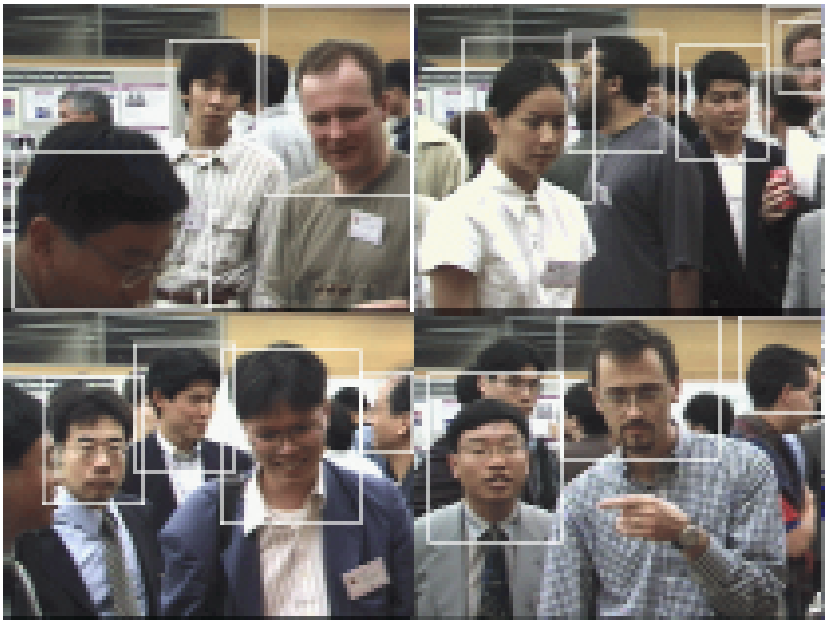


Figura 2.7: *Resultados obtidos por Yachida et al. (1999).*

algumas regiões iniciais. A seguir, as regiões são unidas de acordo com uma similaridade espacial, pois as regiões de formação de um objecto partilham de algumas características espaciais comuns. No segundo passo, as regiões faciais são identificadas a partir dos resultados do primeiro passo, utilizando um modelo da cor da pele. A taxa de sucesso nas detecções de faces desse método é de 82% e a taxa de falsas detecções é de 17%. Essa considerável taxa de falsas detecções é deve-se a erros que ocorrem na segmentação da imagem.

2.1.4 Geometria da face

Em muitas técnicas de detecção de face, o conhecimento da geometria da face tem sido empregue para caracterizar e, posteriormente, verificar várias características faciais nos seus estados de incerteza [15].

Jeng et al. (1998) detectam as faces baseado-se no modelo da geometria

da face [16]. Nesse sistema, tentam-se estabelecer as possíveis localizações dos olhos em imagens binarizadas. Para cada possível par de olhos, o algoritmo irá fazer uma busca à procura de um nariz, uma boca e sobrancelhas. Cada característica facial tem uma função de avaliação associada, que é utilizada para determinar a face candidata. O sistema apresentou uma taxa de detecção de 86% numa base de dados de 114 imagens, não sendo capaz de fazer uma correcta detecção quando existem múltiplas faces, ou nenhuma, na imagem.

Um método desenvolvido por Yow e Cipolla (1997) usa filtros Gaussianos para localizar características semelhantes a barras horizontais nas imagens [29]. Assim, características alongadas, tais como olhos e boca, são localizadas. Comparando as suas relações com as de um modelo de face, as possíveis faces são localizadas. Mudando-se o tamanho do filtro Gaussiano, este método pode detectar faces de diferentes tamanhos numa imagem. Os autores encontraram uma taxa de sucesso na detecção de faces de 85% sobre uma base de dados de 110 imagens de faces com diferentes escalas, orientações e ângulos. Algumas imagens utilizadas nos testes com o algoritmo de Yow e Cipolla (1997) são mostradas nas figuras 2.8 e 2.9.

Figura 2.8: *Resultados da detecção de faces em imagens de faces com diferentes escalas e ângulos.*

Alguns casos ineficazes são mostrados na figura 2.10. Na primeira imagem, as sobrancelhas da pessoa estão realmente muito perto dos olhos, e, nesse ponto de vista, é indistinguível dos olhos. Entretanto, o algoritmo agrupa dois pontos na região do cabelo, formando uma falsa evidência que conduz a uma identificação errada das características faciais. Na segunda imagem, a sobrancelha esquerda da pessoa (a sobrancelha direita

Figura 2.9: *Resultados de detecção de faces em imagens com diferentes orientações.*

na imagem) coincide bem com uma faixa horizontal escura no fundo. Em consequência, a sobrancelha é agrupada com a característica do fundo, numa característica longa, não sendo classificada como uma característica facial e resultando numa configuração geométrica incorrecta da face. Na terceira imagem, a face rodou além do ângulo que o algoritmo pode detectar.

Figura 2.10: *Resultados obtidos por Yow e Cipolla (1997).*

Lin e Fan (2001) propõem a detecção de faces humanas utilizando as relações geométricas do triângulo [18]. Este sistema possui duas partes principais. A primeira consiste na procura de regiões que possam possuir faces. A segunda executa a verificação da face.

A primeira parte do processo é constituída por quatro passos:

1. Ler a imagem e convertê-la para uma imagem binária;

2. Rotular os quatro componentes conectados na imagem para formarem vários blocos e encontrar o centro de cada um deles;

3. Detectar-se quaisquer três centros, de três diferentes blocos, para formarem um triângulo isósceles (imagem frontal) ou um triângulo rectângulo (imagem de perfil);

4. Agrupar os blocos que satisfazem o critério de triângulo como possível face.

A segunda parte é constituída de três etapas:

1. Normalizar o tamanho de todas as possíveis regiões;

2. Fornecer a cada região de possível face normalizada um peso na função máscara;

3. Executar a verificação por *thresholding* do peso obtido na etapa anterior.

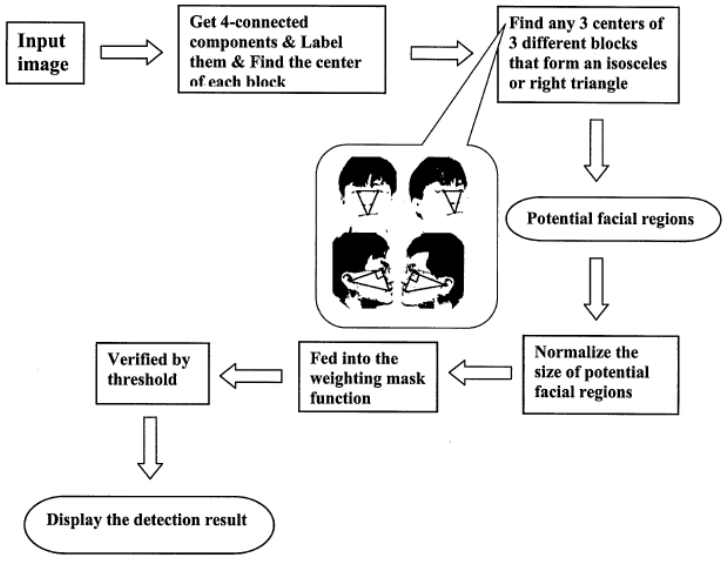


Figura 2.11: *Visão geral do sistema de Lin e Fan (2001).*

Na figura 2.11 é mostrado um diagrama com todas as etapas que o sistema desenvolve, desde da entrada da imagem até o resultado da detecção.

Este método pode lidar com diferentes tamanhos de faces nas imagens, diferentes condições de iluminação, ruído, problema de desfocagem, variação de pose e de expressão. O sistema pode também detectar faces de perfil, faces com problema de oclusão parcial da boca e faces com óculos de sol.

Para validar a eficiência do sistema foram usadas 500 imagens de teste, de 450 pessoas diferentes, resultando no total de 600 faces. A taxa de sucesso deste sistema é de 98%. Na figura 2.12 são mostrados alguns resultados obtidos pelo método proposto por Lin e Fan (2001).

Figura 2.12: *Alguns resultados do sistema proposto por Lin e Fan (2001).*

Algumas falhas do método de Lin e Fan (2001) são apresentadas nas figura 2.13 e 2.14.

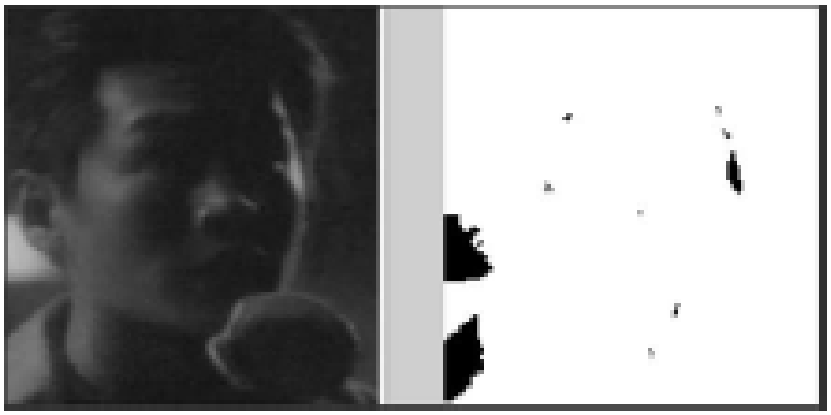


Figura 2.13: *A face está muito escura para ser detectada.*

Figura 2.14: *Face com oclusão do olho direito pelo cabelo preto.*

Capítulo 3

Trabalho desenvolvido

3.1 Método base

O método que foi usado como base neste trabalho foi desenvolvido por Viola e Jones em 2004 [24]. As motivações para esta escolha residem no elevado número de sistemas que utilizam este método e o facto do ser provavelmente o método de detecção de faces humanas mais citado na literatura especializada, com pode se verifica em [7], [3], [28], [25], [22], [19], [5], [11], [8], entre outros.

Nesta secção serão descritos os dois algoritmos propostos pelos autores. Apenas será usado o primeiro deles visto que é o que apresenta melhores resultados. A única desvantagem é que apresenta uma maior lentidão relativamente à segunda variante, mas computadores com maior velocidade de processamento.

Para estes autores, o sistema de detecção de faces apresenta três grandes contribuições para a obtenção dos resultados descritos.

A primeira é a criação de um novo tipo de imagens, designada por imagem integral, que permite uma avaliação muito mais rápida das características. Uma imagem integral pode ser obtida com um único varrimento à imagem original.

A segunda contribuição é o uso de um classificador, simples mas eficaz, que é construído a partir de um pequeno conjunto de características importantes. Estas são provenientes de um conjunto enorme de potenciais características, seleccionadas através de um algoritmo de treino denominado *AdaBoost*. Existe a necessidade de um algoritmo eficaz para o processo de selecção dos melhores classificadores, devido ao facto de que, com apenas os quatro tipos de características apresentadas pelos autores, ser possível criar 160 000 possíveis características numa imagem com uma base de análise de 24 x 24.

A terceira contribuição é a conjugação dos classificadores numa estrutura em cascata, aumentando assim a velocidade de detecção. Este apenas se irá focar nas regiões onda a probabilidade de existir uma face seja bastante significativa. A medida de avaliação deste método é a sua taxa de falsos negativos, pois aqui qualquer sub-imagem que contenha uma face, e que seja rejeitada por um dado patamar da cascata, jamais poderá ser recuperada. As sub-imagens que passam ao patamar seguinte irão sofrer uma nova avaliação um pouco mais complexa que a anterior.

3.1.1 Imagem integral

O conceito de imagem integral é crucial no método proposto. Num dado ponto (x,y), esta imagem contém a soma dos *pixels* acima e à esquerda de x, y, inclusive:

$$ii(x, y) = \sum_{x'=1}^{x} \sum_{y'=1}^{y} i(x', y') \tag{3.1}$$

onde ii(x,y) é a imagem integral e i(x,y) é a imagem original (ver imagem 3.1).

$$s(x, y) = s(x, y - 1) + i(x, y) \tag{3.2}$$

$$ii(x, y) = ii(x - 1, y) + s(x, y) \tag{3.3}$$

onde s(x,y) é a soma da coluna, s(x,-1)=0 e ii(-1,y)=0. Com estas duas fórmulas é possível calcular a imagem integral através de uma única passagem pela imagem original.

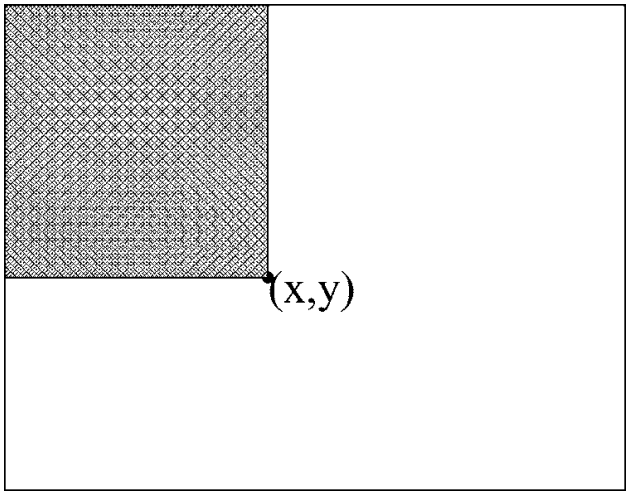


Figura 3.1: *O valor da imagem integral no ponto (x,y) é a soma dos pontos acima e à esquerda.*

Na imagem 3.2, pode se ver que a soma dos *pixels* do rectângulo D pode ser encontrada com o acesso a apenas quatro pontos da imagem. O valor da imagem integral no ponto 1 é a soma do valor dos *pixels* do rectângulo A. O valor na posição 2 é a soma dos valores dos *pixels* dos rectângulos A e B, na posição 3 o dos rectângulos A e C e na posição 4 a soma dos quatro rectângulos. A soma dos *pixels* do rectângulo D é dada pelos valores em 4 + 1 - (2 + 3).

3.1.2 Características

Este sistema de detecção de faces classifica as imagens baseando-se no valor das características simples. Os autores afirmam que existem inúmeras razões para que sejam usadas características, em vez dos valores

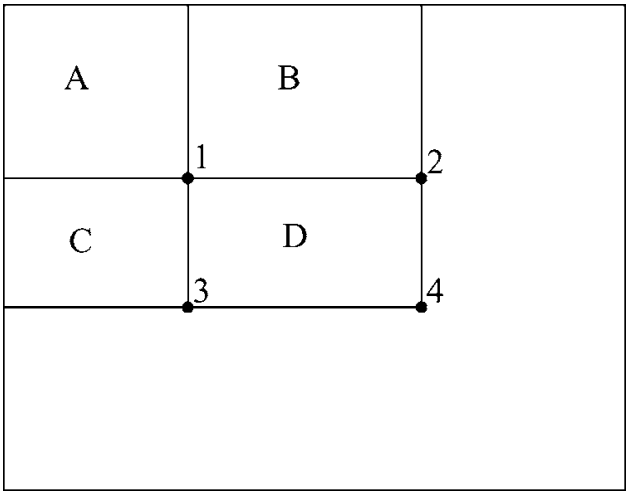


Figura 3.2: *Imagem que representa os pontos necessários para o cálculo do valor da soma dos pixels para um dado rectângulo.*

dos *pixels* directamente. A mais comum é que estas podem servir para codificar especificamente um dado domínio, que é difícil de treinar usando uma quantidade finita de dados de treino, e que um sistema baseado em características é mais rápido do que os baseados nos valores dos *pixels*.

O valor, de uma dada característica constituída por dois rectângulos (rectângulos A e B da imagem 3.3), é a diferença entre a soma dos *pixels* dentro do rectângulo da zona branca a soma dos *pixels* do rectângulo da zona preta. Estes rectângulos têm o mesmo tamanho, forma e são horizontalmente ou verticalmente adjacentes. Para as características baseadas em três rectângulos, o valor é dado pela diferença entre a soma dos *pixels* dos rectângulos exteriores e o do rectângulo interior. Finalmente, as características que usam quatro rectângulos, com as mesmas proporções, é processada a diferença entre os pares diagonais de rectângulos.

Estas características são um pouco primitivas quando comparadas com outros tipos de características, como por exemplo características baseadas em filtros dirigíveis [12] [14]. Estes filtros são muito mais eficazes para a análise detalhada de limites, compressão de imagens e a análise de texturas. Enquanto as características baseadas em rectângulos são sensíveis

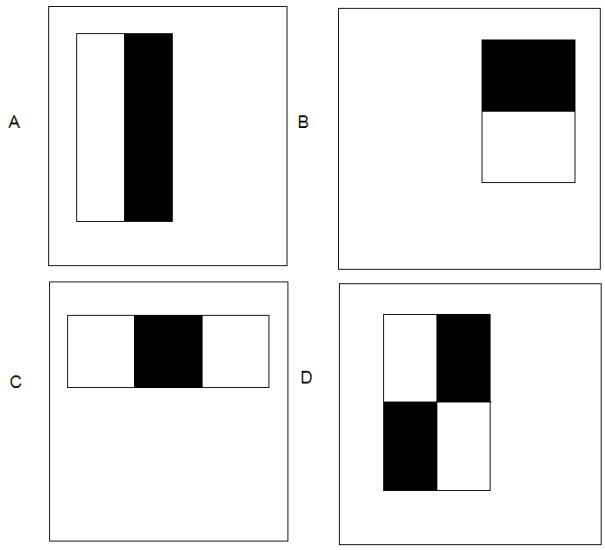


Figura 3.3: *Exemplos de características baseadas em rectângulos numa posição relativa da janela de detecção.*

à presença de extremidades, barras ou outras estruturas de imagens simples, estas são bastantes mais grosseiras na análise de imagens. As únicas orientações possíveis para os filtros dirigíveis são verticais, horizontais ou diagonais. Considerando que uma dada característica não é central no plano ortogonal, tem de ser gerado um grande conjunto de características baseadas em rectângulos com aspecto mais variado possível para que o algoritmo de treino seja capaz de determinar se uma dada característica é melhor de que as outras todas do conjunto. A eficiência computacional destas características compensa as suas limitações em termos de detalhes.

Para aproveitar a vantagem computacional da técnica da imagem integral, considere-se uma aproximação mais convencional na qual é processada uma pirâmide de imagens. Como a maioria dos sistemas de detecção de faces, este detector processa muitas sub-janelas em diferentes escalas. Nesta abordagem, inicialmente, o detector baseia-se em sub-janelas com um tamanho de 24 x 24 *pixels*, indo aumentando este tamanho segundo uma escala de 1.25 até ao tamanho da imagem original de 384 x 288 *pixels*. Esta variação na escala irá fazer com que seja processada uma pirâmide de

12 imagens, cada uma delas 1.25 vezes superior à anterior. Isto pode ser aplicado devido ao facto de que o custo computacional ser o mesmo tanto para janelas de 24 x 24 do que para 100 x 100 *pixels*. Isto é, o processo é invariante à escala.

3.1.3 Detecção com classificador sequencial

Dado um conjunto de características e um conjunto de treino contendo imagens de faces e não faces, qualquer processo de treino pode ser usado para criar um classificador.

Recorde-se que se podem extrair cerca de 160 000 características baseadas em rectângulos em imagens de 24 x 24, um número bastante maior que o número de *pixels* desta imagem. Mesmo que cada característica possa ser processada muito eficazmente, processar todo o conjunto torna-se muito moroso. A hipótese é combinar um número muito pequeno destas características de forma a criar um classificador forte.

Neste sistema é usado uma variante do algoritmo *AdaBoost* para seleccionar as melhores características e treinar o classificador [13]. Na sua forma original, o *AdaBoost* é utilizado para impulsionar o desempenho da classificação de um algoritmo de treino simples (por exemplo poderia ser usado para impulsionar o desempenho de um perceptron simples). Faz isso combinando uma colecção de classificadores fracos para formar um classificador forte.

O treino do algoritmo do perceptron tenta procurar, num dado conjunto de possíveis perceptrons, aquele que possui o menor erro. É denominado por classificador fraco pois não se espera a melhor função de classificação, nem sequer que seja capaz de classificar bem todos os dados de treino, isto é, para uma determinada fase do treino, este classificador fraco poderá classificar bem apenas 51% dos dados. Depois de ser encontrado o primeiro classificador fraco, os pesos das imagens de treino são actualizados para que, no passo seguinte, seja encontrado outro classificador fraco. O classificador forte final leva a forma de um perceptron, sendo esta

uma combinação de classificadores fracos pesados e seguidos de um limiar.

O procedimento convencional do *AdaBoost* pode ser interpretado como um processo de selecção de características ganancioso. Este processo associa um peso superior às melhores funções de classificação e um peso mais baixo às mais fracas. O *AdaBoost* é um mecanismo agressivo porque selecciona um pequeno conjunto de bons classificadores de um conjunto enorme de possíveis classificadores.

Estabelecendo uma analogia entre classificadores fracos e características, o *AdaBoost* é um procedimento efectivo para a procura de um pequeno número de boas características entre uma variedade significativa. Um método prático para completar esta analogia é restringir o estudante fraco ao conjunto de funções de classificação, cada uma das quais dependem de uma única característica. Em defesa desta meta, o algoritmo de aprendizagem é projectado para seleccionar a única característica baseada em rectângulos que melhor separa os exemplos positivos dos negativos. Para cada característica o estudante fraco determina o óptimo limiar da função de classificação. Um classificador fraco (h(x,f,p,θ)) consiste numa sub-janela da imagem (x), uma característica (f), um limiar (θ) e uma polaridade (p), sendo que esta indica a direcção da desigualdade:

$$h(x, f, p, \theta) = \begin{cases} 1 & \text{se pf(x)} < \text{p}\theta \\ 0 & \text{caso contrário} \end{cases} \tag{3.4}$$

O algoritmo 1 é usado para seleccionar os melhores classificadores fracos de um conjunto enorme destes. Forma um conjunto grande devido à que existência de um classificador fraco para cada característica distinta/combinação de limiar. Há efectivamente, KN classificadores fracos, onde K é o número de características e N o número de exemplos de treino. Para demonstrar a dependência existente em N, suponha-se que os exemplos são ordenados pelo valor da característica, sendo este denominado por limiar. Com isto, existe o mesmo número de limiares possíveis por características e exemplos de treino, sendo assim, que para cada tarefa com $N = 21000$ e $K = 160000$, há cerca de $3.36 * 10^9$ classificadores fracos

Algoritmo 1 Este é responsável por construir um classificador forte, combinando os T melhores classificadores fracos. Esta combinação é feita através de uma combinação linear pesada em que os pesos são o inverso dos erros no conjunto de treino.

Para todas as imagens de treino $(x_1,y_1) \ldots (x_n,y_n)$, onde $y_i = 0, 1$ para os exemplos não faces e face, respectivamente.

Inicializar os pesos $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ para $y_i = 0, 1$ respectivamente, onde m e l são os números de exemplos negativos e positivos, respectivamente.

for $t = 1$ to T **do**

Normalizar os pesos: $w_{t,i} = \frac{w_{t,i}}{\sum_{j=1} n}$.

Com base no erro, seleccionar a melhor característica:

$$e_t = min_{f,p,\theta}(\sum_i w_i \mid h(x, f, p, \theta) - y_i \mid).$$

Definir $h_t(x) = h(x, f_t, p_t, \theta_t)$, onde f_t, p_t e θ_t minimizam o erro.

Actualizar os pesos:

$$w_{t+1,i} = w_{t,i}\beta_t^{1-e_i},$$

onde $e_i = 0$ se o exemplo x_i for classificado correctamente, $e_i = 1$ caso contrário e $\beta_t = \frac{e_t}{1-e_t}$.

end for

O classificador forte final é dado por:

$$C(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t \geqslant \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{caso contrário} \end{cases}$$

possíveis de onde serão escolhidos os T melhores classificadores fracos.

A principal vantagem do *AdaBoost*, como mecanismo de selecção de características, é a velocidade de aprendizagem, pois este algoritmo tem uma complexidade de $O(TNK)$, bastante mais reduzida do que a de outros algoritmos de selecção de características. Uma outra vantagem é que, pelo facto dos exemplos de treino serem pesados, a selecção de uma característica depende sempre da característica seleccionada anteriormente.

Para cada característica, os exemplos são ordenados pelo valor do limiar. O limiar óptimo para cada característica a ser processada pode ser encontrado através de uma única passagem pela lista ordenada. Para cada elemento na lista ordenada, são actualizadas quatro somas, pois são estas que irão definir qual o melhor erro, sendo a soma total dos pesos dos exemplos positivos T^+, a soma total dos pesos das imagens negativas T^-, a soma dos pesos dos exemplos positivos abaixo do exemplo actual S^+ e a soma dos pesos dos exemplos negativos abaixo S^-. O erro para um limiar que divide a lista ordenada é obtido através da aplicação da seguinte fórmula:

$$e = min(S^+ + (T^- - S^-), S^- + (T^+ - S^+)) \tag{3.5}$$

O erro mínimo será aquele que consegue dividir melhor os exemplos positivos dos negativos. As somas são facilmente actualizadas visto que sempre que se anda de elemento em elemento, na lista, sabemos de ante mão se ela é ou não uma face.

3.1.4 Detecção com classificador em cascata

Esta secção descreve um algoritmo para construir uma cascata de classificadores. Em relação ao algoritmo 1, há uma taxa de erro superior mas uma velocidade de processamento bastante superior também. A vantagem fundamental para a obtenção de uma velocidade superior de processamento é que este tenta produzir, inicialmente, classificadores mais simples, que tentam rejeitar o maior número de sub-janelas negativas possível, para que os

classificadores mais complexos analisem um menor número de sub-janelas.

Os patamares de cascata são construídos com base no algoritmo 1, visto que estes são vários classificadores sequenciais, cada um deles com um número menor de classificadores fracos. O primeiro classificador é descrito como tendo apenas dois classificadores fracos, mas é ajustado o limiar do classificador forte para minimizar o número de falsos negativos. O limiar inicial do *AdaBoost*, $\frac{1}{2}\sum_{t=1}^{T}\alpha_t$, é projectado para obter uma baixa taxa de erro no conjunto de treino. Um mais baixo limiar rende umas taxas superiores, tanto nas detecções como nos falsos positivos.

O desempenho de detecção de um classificador com apenas duas características é longe de aceitável, mas pode reduzir o número de sub-janelas que necessitam de ser analisadas nos processos seguintes, através de muitas poucas operações.

O processo de detecção tem a forma de uma árvore de decisão degenerada, o que foi denominado por cascata. Um resultado positivo no primeiro classificador activa a avaliação no segundo (ver figura 3.4), que também foi ajustado para alcançar uma alta taxa de detecção, e assim por diante. Um resultado negativo em qualquer ponto da cascata conduz à rejeição imediata da sub-janela. Esta estrutura reflecte o facto de que, numa qualquer imagem, existe uma esmagadora maioria de sub-janelas negativas. Como tal, a cascata tenta rejeitar tantas sub-janelas negativas quantas possíveis, numa fase inicial. Um exemplo positivo activa a avaliação de todo o classificador na cascata, o que será um evento sumamente raro.

A estrutura da cascata é construída com base num conjunto de detecção e de metas de desempenho. O número de fases ou patamares da cascata, e o tamanho de cada fase, deve ser o suficiente para alcançar um desempenho de detecção aceitável.

Dada uma cascata de classificadores, a taxa de falsos positivos da cascata é:

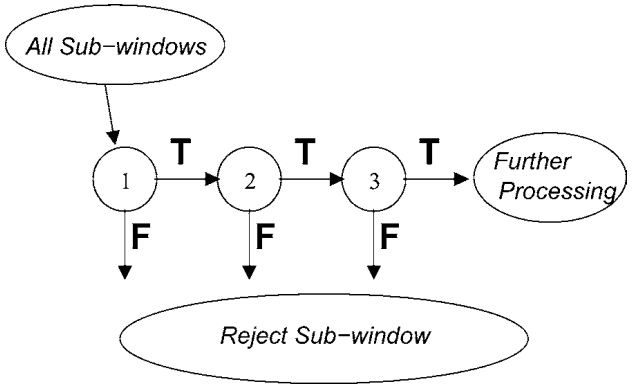


Figura 3.4: *Representação do processo de detecção em cascata.*

$$F = \prod_{i=1}^{K} f_i \tag{3.6}$$

onde F é a taxa de falsos positivos do classificador em cascata; K o número de classificadores; f_i a taxa de falsos positivos do i-ésimo classificador. A taxa de detecção é:

$$D = \prod_{i=1}^{K} d_i \tag{3.7}$$

onde D é a taxa de detecção do classificador em cascata, K o número de classificadores, e d_i a taxa de detecção do i-ésimo classificador. Determinadas metas concretas para os falsos positivos e taxas de detecção podem ser determinadas: são as taxas designadas para cada fase no processo de cascata.

Uma determinada sub-janela progredirá pela cascata, de classificador em classificador, até que seja decidido se esta é ou não uma face. O comportamento esperado deste processo é determinado pela distribuição das sub-janelas da imagem num conjunto de teste. A medida fundamental de cada classificador é a taxa de faces detectadas e a proporção de sub-janelas contendo possíveis faces perdidas. O número esperado de características

que são avaliadas é:

$$N = n_0 + \sum_{i=1}^{K} (n_i \prod_{j<i} p_j) \tag{3.8}$$

onde N é o número esperado de características avaliadas; K o número de classificadores; p_i a taxa de faces detectadas no i-ésimo classificador; n_i o número de características do i-ésimo classificador.

O processo pelo qual cada elemento da cascata é treinado requer um pouco de cuidado. O procedimento de treino baseado no *AdaBoost*, algoritmo 1, apenas tenta minimizar os erros, e não é projectado para apresentar uma alta taxa de detecção com base nas taxas de falsos positivos. Um simples, e muito convencional, esquema para estes erros consiste em ajustar o limiar do perceptron produzido pelo *AdaBoost*. Limiares mais altos produzem classificadores com menos falsos positivos e uma mais baixa taxa de detecção; limiares mais baixos rendem classificadores com mais falsos positivos e uma taxa de detecção superior.

O processo global de treino envolve dois tipos de intercâmbios. Na maioria dos casos, alcançarão classificadores com mais características, mais alta taxa de detecção e mais baixas taxas de falsos positivos. Ao mesmo tempo, classificadores com mais características requerem mais tempo de processamento. Em princípio, a pessoa poderia definir um modelo de optimização, no qual ajustaria:

- O número de fases do classificador;
- O número de características, n_i, de cada fase;
- O limiar de cada fase;

para minimizar o número de características N, dado um objectivo para F e D. Infelizmente, procurar esta optimização é um problema difícil.

Algoritmo 2 Algoritmo para a construção de um detector em cascata.

Seleccionar os valores para f, taxa máxima de falsos positivos por fase, e d, taxa minima de detecções por fase.
Seleccionar a taxa de falsos positivos global.
P = conjunto de exemplos positivos
N = conjunto de exemplos negativos
$F_0 = 1.0$; $D_0 = 1.0$
$i = 0$

while $F_i > F_{global}$ **do**

$i = i + 1$
$n_i = 0$; $F_i = F_{i-1}$

while $F_i > f * F_{i-1}$ **do**

$n_i = n_i + 1$
Usar P e N para treinar o classificador com n_i características, usando o *AdaBoost*.
Avaliar o classificador actual com um conjunto de validação para determinar F_i e D_i.
Diminuir o limiar para o i-ésimo classificador até que a taxa de detecção da corrente cascata seja the $d * D_{i-1}$ (isto também afecta F_i).

end while

N = 0

if $F_i > F_{global}$ **then**

Avaliar o corrente detector em cascata num conjunto de imagens que não contêm faces e inserir as que forem mal classificadas dentro do conjunto N.

end if
end while

Na prática, esta vigilância muito simples é usada para produzir um classificador efectivo e eficiente. O utilizador define a taxa máxima aceitável para f_i e a taxa mínima aceitável para d_i. Cada patamar da cascata é treinado pelo *AdaBoost* (descrito pelo algoritmo 1) com o número de características adequadas para a taxa de detecção designada. As taxas designadas para o detector são testadas num conjunto de validação. Se a taxa dos falsos positivos não for atingida, uma nova característica será acrescentada à cascata. Este algoritmo é descrito mais precisamente pelo algoritmo 2.

Há um benefício escondido para treinar um detector como uma sucessão de classificadores que corresponde ao número de exemplos negativos que o detector final pode ver, sendo que este poderá ser muito grande. Pode-se imaginar o treino de um único classificador, com muitas características, e tentar acelerar o seu tempo de processamento, olhando apenas para as somas parciais das características e parando o seu processamento. Então, a soma parcial será abaixo do limiar apropriado. A desvantagem desta abordagem é que, desta forma, o número de exemplos negativos no conjunto de treino teria de ser relativamente baixo para se poder fazer um treino viável.

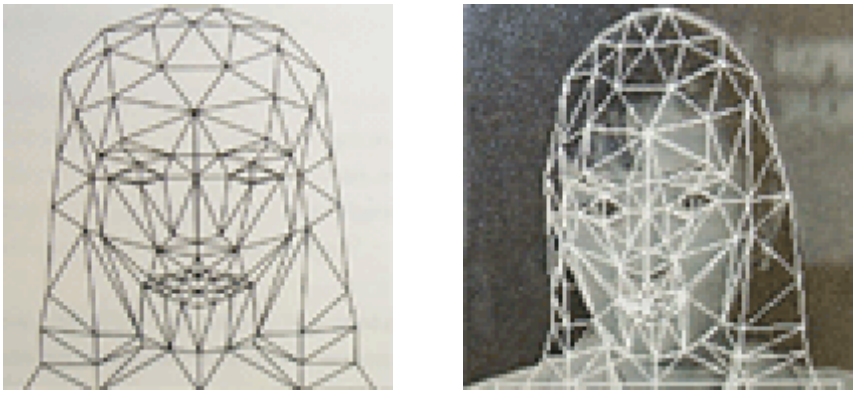
3.2 Método proposto

Nesta secção serão descritas todas as alterações feitas ao método base (proposto por Viola e Jones em 2004 [24]) na tentativa de o melhorar.

3.2.1 Características

As alterações passam pela extracção de novas características, sendo estas baseadas em triângulos. A ideia da extracção de características baseadas em triângulos provém da ideia de que todos os objectos 3D, em computação gráfica, podem ser representados através de malhas de triângulos, isto é, um conjunto de triângulos ligados entre si através das suas extremidades. Com isso achamos que poderia ser uma boa abordagem para o

problema de detecção de faces humanas, tentando que ao longo do processo de treino de um classificador forte final fosse criando uma malha de triângulos através da qual fosse possível fazer a representação de faces humanas em 2D (ver figura 3.5), através dos modelos a 3D (ver figura 3.6). É de realçar o facto de os resultados comprovarem isso mesmo. Havendo uma melhoria no erro obtido, em comparação com os do método base, foi feita a submissão de um artigo para a *9th International Conference on Signal Processing* (ICSP'08).



(a) Malha de triângulos em 2D.

(b) Malha de triângulos em 2D aplicada numa face humana.

Figura 3.5: *Representação de uma face através de uma malha de triângulos a 2D.*

Figura 3.6: *Representação de uma face através de uma malha de triângulos em 3D.*

O ICSP'08 irá decorrer na *Beijing Jiaotong University (Northern Jiaotong University)*, em Beijing na China entre os dias 26 e 29 de Outubro de 2008. A organização da conferência está a cargo da universidade e pelo *CIE Signal Processing Society*. Os artigos aceites são publicados pelo *Institute of Electrical and Electronics Engineers* (IEEE).

Na figura 3.7 estão demonstradas as características baseadas em triângulos rectangulares. Estas têm a mesma base que as características baseadas em rectângulos, isto é, todas as características são determinadas pela diferença das somas das regiões escuras e claras. Estas novas características foram denominadas por E, F, G e H, sendo a sequência aos nomes de características desenvolvidas por Viola e Jones [24]. As características descritas por E, F, G e H são triângulos equiláteros com quatro orientações diferentes, sendo superior, direita, inferior e esquerda respectivamente, estando cada uma dividida internamente em dois triângulos rectângulos, o triângulo claro e o escuro. Com base numa imagem de 24 x 24 *pixels* é possível criar cerca de 110 400 destas características diferentes, já com a aplicação de regras para que o tamanho destas tivesse alguma representação significativa, limitando a área mínima de cada tipo de triângulos ou distancia entre cada ponto. Estas limitações devem-se ao facto de que uma característica que abranja um número pixel relativamente pequeno não possui informação suficiente para poder dizer se existe ou não uma face na janela.

3.2.2 Imagens integrais triangulares

Para a integração das novas características no método base foi necessário recorrer à criação de quatro novos tipos de imagens integrais, sendo cada uma delas denominada por imagem integral triangular Nordeste (figura 3.8(a)), Noroeste (figura 3.8(b)), Sudeste (figura 3.8(c)) e Sudoeste (figura 3.8(d)). Estes nomes dependem de para onde está direccionado o ângulo recto do triângulo.

O valor de um pixel num ponto (x,y) nas imagens integrais triangulares apresentadas na figura 3.8 é dado por:

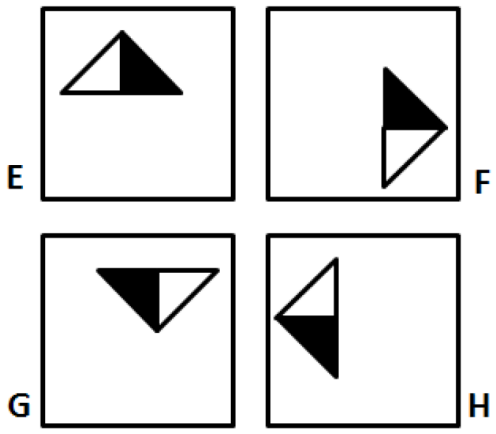


Figura 3.7: *Exemplos de características baseadas em triângulos em posições relativas da janela de detecção.*

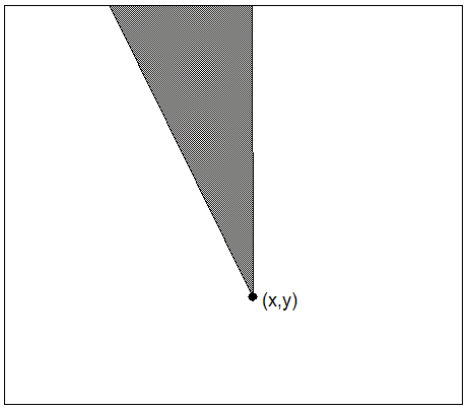
$$iit_{NE}(x,y) = \sum_{l=1}^{x}\sum_{c=1}^{l} i(c, y-x+l) \tag{3.9}$$

$$iit_{NO}(x,y) = \sum_{l=1}^{nC-x}\sum_{c=1}^{l} i(nC-c, y-x+l) \tag{3.10}$$

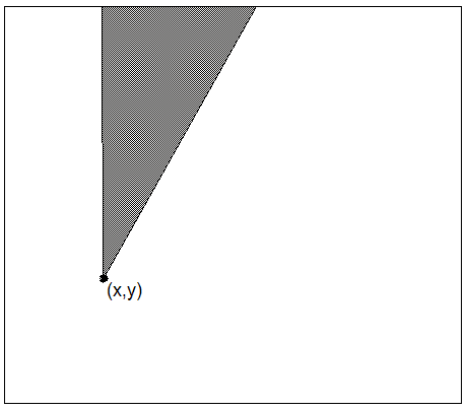
$$iit_{SE}(x,y) = \sum_{l=1}^{x}\sum_{c=1}^{l} i(c, y+x-l+1) \tag{3.11}$$

$$iit_{SO}(x,y) = \sum_{l=1}^{nC-x}\sum_{c=1}^{l} i(nC-c, y+nC-x-l) \tag{3.12}$$

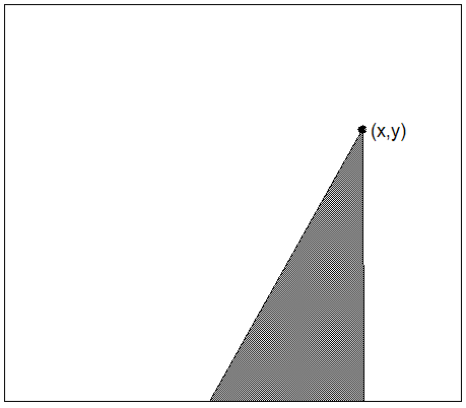
onde i é a imagem original em escala de cinza, nC e nL é o número de colunas e linhas, respectivamente, da imagem e i(x,y) é o pixel da imagem original na coluna x e linha y (considera-se que $i(x,y) = 0$ se $x \leq 0$ ou $y \leq 0$ ou $x > nC$ ou $y > nL$). Desta forma o nosso método iria perder imenso tempo só para calcular estas imagens integrais, pois necessita de fazer várias passagens pela imagem original para calcular apenas uma imagem integral. Para diminuir o tempo de computação tiveram de ser



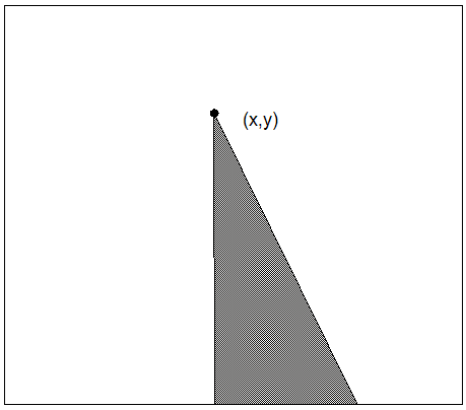
(a) Imagem integral triangular Nordeste(iit_{NE})



(b) Imagem integral triangular Noroeste(iit_{NO})



(c) Imagem integral triangular Sudeste(iit_{SE})



(d) Imagem integral triangular Sudoeste(iit_{SO})

Figura 3.8: *Representação da soma dos pixels num dado ponto (x,y) nas imagens integrais triangulares.*

revistas as equações acima para que fosse possível calculá-las apenas com uma passagem na imagem, sendo que as equações encontradas foram as seguintes:

Imagem integral triangular Nordeste

$$s_{NE}(x, y) = s_{NE}(x, y - 1) + i(x, y) \tag{3.13}$$

$$iit_{NE}(x, y) = iit_{NE}(x - 1, y - 1) + s_{NE}(x, y) \tag{3.14}$$

Imagem integral triangular Noroeste

$$s_{NO}(x, y) = s_{NO}(x, y - 1) + i(x, y) \tag{3.15}$$

$$iit_{NO}(x, y) = iit_{NO}(x + 1, y - 1) + s_{NO}(x, y) \tag{3.16}$$

Imagem integral triangular Sudeste

$$s_{SE}(x, y) = s_{SE}(x, y + 1) + i(x, y) \tag{3.17}$$

$$iit_{SE}(x, y) = iit_{SE}(x - 1, y + 1) + s_{SE}(x, y) \tag{3.18}$$

Imagem integral triangular Sudoeste

$$s_{SO}(x, y) = s_{SO}(x, y + 1) + i(x, y) \tag{3.19}$$

$$iit_{SO}(x, y) = iit_{SO}(x + 1, y + 1) + s_{SO}(x, y) \tag{3.20}$$

Com isto, estamos aptos a calcular todas as imagens integrais necessárias para que se possam aplicar todos os tipos de características apresentadas até ao momento. Este cálculo é possível através do algoritmo 3. Este algoritmo tem uma complexidade bastante baixa ($O(N^2)$) para o número de operações que são feitas. Só desta forma se pode contribuir para que o sistema de detecção de faces mantenha as características de um sistema em tempo real.

O cálculo da soma dos *pixels* definidos pela região "A", nas sub-figuras da figura 3.9, estão respectivamente baseadas na imagem integral triangular Nordeste (figura 3.9(a)), Noroeste (figura 3.9(b)), Sudeste (figura 3.9(c)) e Sudoeste (figura 3.9(d)). Com isto podemos calcular a soma dos valores dos *pixels* de uma dada região "A" usando um número muito limitado de referencias à imagem, isto é, com apenas 4 ou 6 referencias à imagem é

Algoritmo 3 Cálculo de todas as imagens integrais (imagem integral original proposta por Viola e Jones (ii) e as nossas (iit_x)) com uma complexidade de $O(N^2)$.

```
(...)
for l = 1 to nL do
   for c = 1 to nC do

      ii(c, l) = ... {imagem integral original}
      iit_NE(c, l) = ... {imagem integral triangular Nordeste}
      iit_NO(nC − c, l) = ... {imagem integral triangular Noroeste}
      iit_SE(c, nL − l) = ... {imagem integral triangular Sudeste}
      iit_SO(nC − c, nL − l) = ... {imagem integral triangular Sudoeste}

   end for
end for
(...)
```

possível saber a soma exacta dos *pixels* da pretendida região, como pode constatar nas seguintes equações:

$$SA_{NE}(x,y) = iit_{NE}(1) + ii(3) - (iit_{NE}(3) + ii(2)) \tag{3.21}$$

$$SA_{NO}(x,y) = iit_{NO}(1) + ii(2) - (iit_{NO}(3) + ii(3)) \tag{3.22}$$

$$SA_{SE}(x,y) = iit_{SE}(1) + ii(4) + ii(2) - (iit_{SE}(3) + ii(5) + ii(3)) \tag{3.23}$$

$$SA_{SO}(x,y) = iit_{SO}(1) + ii(4) + ii(2) - (iit_{SO}(2) + ii(5) + ii(3)) \tag{3.24}$$

onde SA_x representa a soma dos *pixels* da região "A" usando a imagem integral triangular x para calcular essa soma, iit_x a respectiva imagem integral triangular e ii a imagem integral proposta por Viola e Jones [24].

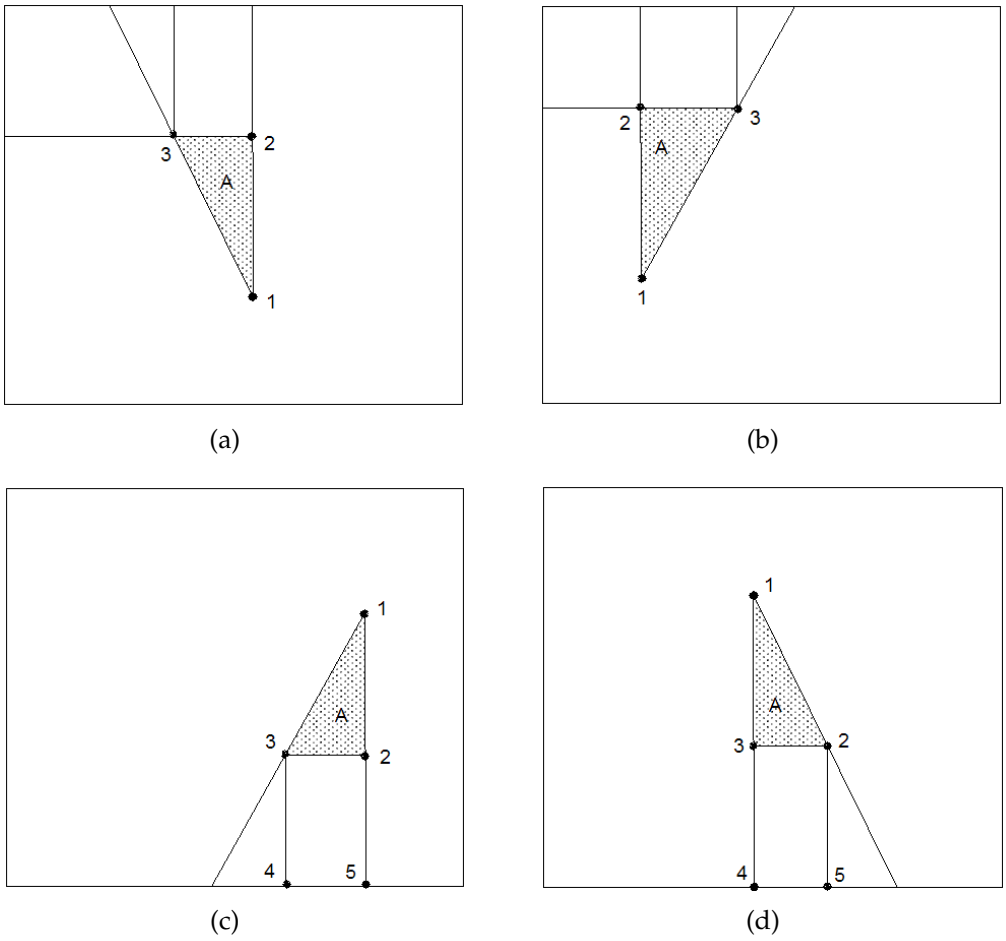


Figura 3.9: *Representação dos pontos necessários para o calculo da soma dos pixels contidos na região "A".*

Capítulo 4

Resultados

4.1 Conjunto de treino e teste

Para o treinar e testar o método proposto por Viola e Jones [24], bem como o nosso, foi criado um conjunto de treino / teste com imagens retiradas da internet. Estas imagens foram escolhias para que tivessem a mais variada complexidade de fundo e que as faces tivessem diferentes focos de iluminação, só com as mais variadas condições é que é possível verificar a eficácia de um método, pois se um método for capaz de se portar bem com imagens retiradas em diferentes condições, ainda melhor se irá portar quando for preparado para estar a funcionar numas determinadas condições específicas, isto é, treinado para funcionar num local fixo.

Destas imagens foi cortado e redimensionado, para 24 x 24 *pixels*, um conjunto de sub-imagens significativamente grande (30 000 sub-imagens no total e que está dividido em 10 000 faces e 20 000 não faces). Deste foram retiradas 1000 faces e 2000 não faces para o conjunto de teste, as outras foram para o conjunto de treino. Um método como este, baseado em características, necessita de ser treinado usando um conjunto de treino bastante grande e de preferência que o número de imagens de treino que não contenham faces seja o dobro do número das imagens de treino contendo faces, por forma a reflectir de alguma forma as probabilidades à priori. Esta quantidade de dados é necessária para que o algoritmo de treino consiga ajustar o melhor possível o parâmetro dos nossos classifi-

cadores fracos, isto é, ajustar os valores dos limiares, bem como os dos erros.

Figura 4.1: *Exemplos de imagens usadas nas experiências.*

Na figura 4.1 pode ver algumas das imagens com faces contidas no conjunto de treino e de teste. Estes conjuntos, o de treino e teste, contêm apenas imagens frontais e com uma muito pequena inclinação ou rotação.

Como se nota não só se procuraram imagens com várias iluminações, complexidades de fundo (contendo prédios, ruas, natureza e muitos outros tipos de padrões), mas também tentou-se ter uma variedade muito grande de tons de pele, desde caucasianos, índios ou orientais. Essencialmente tentou-se arranjar um conjunto de imagens que fossem capazes de conter um pouco de tudo o que existe à nossa volta no nosso dia-a-dia.

4.2 Resultados

Com base nas imagens do conjunto de treino e nas características disponíveis (características propostas por Viola e Jones [24] e as nossas) foram treinados cerca de 180 classificadores fortes, por forma a alcançar relevância estatística para os resultados obtidos. No processo de teste variou-se alguns parâmetros, sendo que estes:

- O tipo de características usadas, podendo ser as A, B, C e D (desenvolvidas por Viola e Jones [24]), E, F, G e H (propostas por nós) e ainda a conjugação de todas elas.
- A percentagem de características seleccionadas aleatoriamente (variando este número entre 10 a 50% de 10 em 10%). Posteriormente, teve de se reduzir o número de características devido ao tempo que demora a ser produzido cada um dos classificadores fortes.
- Número de classificadores fracos pretendidos no classificador forte (foram produzidos classificadores fortes com 100, 200 e 300 classificadores fracos).

Para cada um dos parâmetros foram gerados cinco testes para que se pudesse verificar a estabilidade das características deste método. Com isto foram produzidos 75 testes contendo apenas as características apresentadas por Viola e Jones [24] e outras tantos para as que conjugam as características deles e as nossas. Mesmo obtendo erros superiores para os testes realizados com apenas as características dos E, F, G e H, quando estas são integradas nas características iniciais (apresentadas por Viola e Jones [24]) passam a ter melhores erros. A percentagem de características do tipo E, F, G e H seleccionadas na integração dos dois tipos de características baseadas em rectângulos e em triângulos é apresentada na tabela 4.1.

Sendo as características do tipo E e G as mais seleccionadas durante o processo de treino, pode ver na figura 4.2 as posições mais frequentes destes tipos de características em relação às janelas de detecção.

Tipo de características	Proporção (%) de características seleccionadas
E	7.43
F	2.31
G	6.91
H	2.27
E, F, G e H	18.93

Tabela 4.1: *Proporção de características do tipo E, F, G e H seleccionadas automaticamente pelo algoritmo de treino em relação ao total de características seleccionadas.*

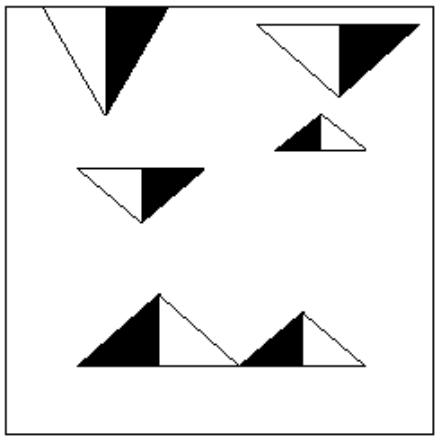


Figura 4.2: *Características do tipo E e G seleccionadas mais frequentemente pelo algoritmo de treino.*

O facto mais saliente dos testes efectuados é a selecção de um número considerável das características propostas (E, F, G e H) pelo algoritmo *AdaBoost*, o que indica a sua utilidade. Este facto veio fazer com que os resultados melhorassem nalguns testes, como pode ver na tabela 4.2 ou nos gráficos das figuras 4.3 e 4.4 onde são apresentados as curvas ROC para os dois melhores resultados para os classificadores com 200 e 300 características do tipo A, B, C e D, e A, B, C, D, E, F, G e H, respectivamente. Nesta pode-se ver quais foram os melhores EER obtidas nos testes. É de notar que todos os testes realizados usando apenas as características do tipo E, F, G e H deram erros que rondam o dobro dos outros dois conjuntos, mas quando estas são integradas nas características do tipo A, B, C e D estas tendem e ajudar na melhoria dos erros.

Tipo de características	Número de características do classificador forte		
	100	200	300
A, B, C e D	0.022152	0.016621	0.013523
E, F, G e H	0.041056	0.033763	0.029545
A, B, C, D, E, F, G e H	0.021316	0.015000	0.014200

Tabela 4.2: *Valores mínimos dos EER's nos testes realizados.*

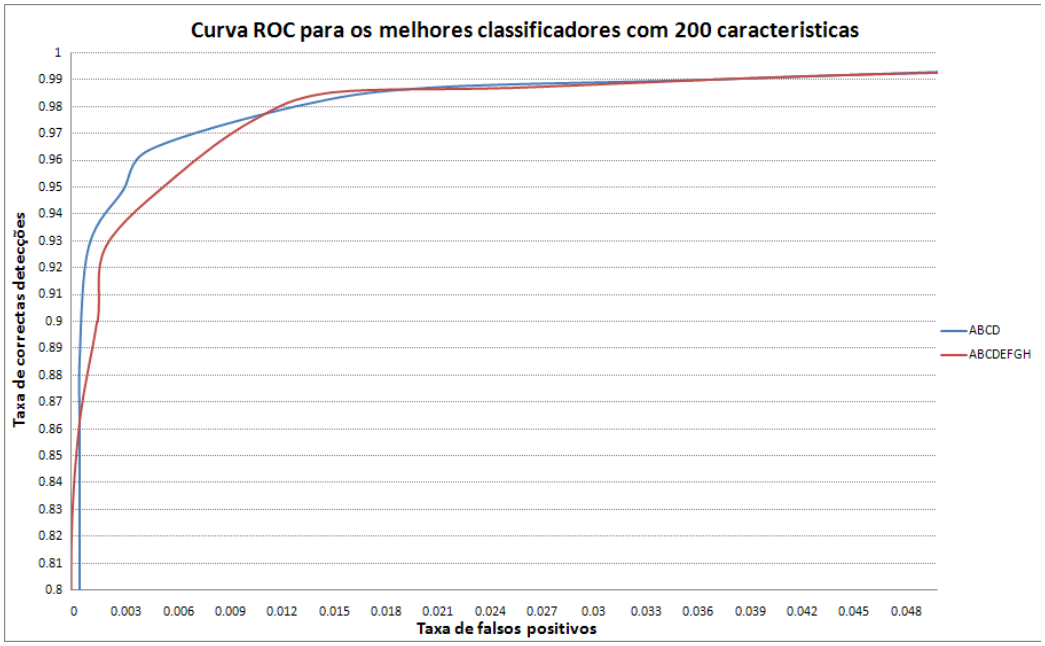


Figura 4.3: *Curva ROC dos dois melhores classificadores fortes com 200 características.*

Com base nos dados das curvas ROC, apresentadas nos gráficos das figuras 4.3 e 4.4, podem ser representados também outro tipo de gráficos (ver gráficos das figuras 4.5, 4.6, 4.7 e 4.8), sendo que estes representam a relação entre o FPR e o FNR. No processo de detecção é muito importante a análise destes gráficos, pois dependendo do erro aceite no classificador final terá que se ajustar o limiar de decisão. O valor inicial, apresentado por Viola e Jones [24], é de $\frac{1}{2}$, mas diminuindo este permitimos que a taxa de detecções e de falsos positivos aumente e vice-versa. Com a análise destes gráficos somos capazes de prever qual a taxa de erro que se irá obter

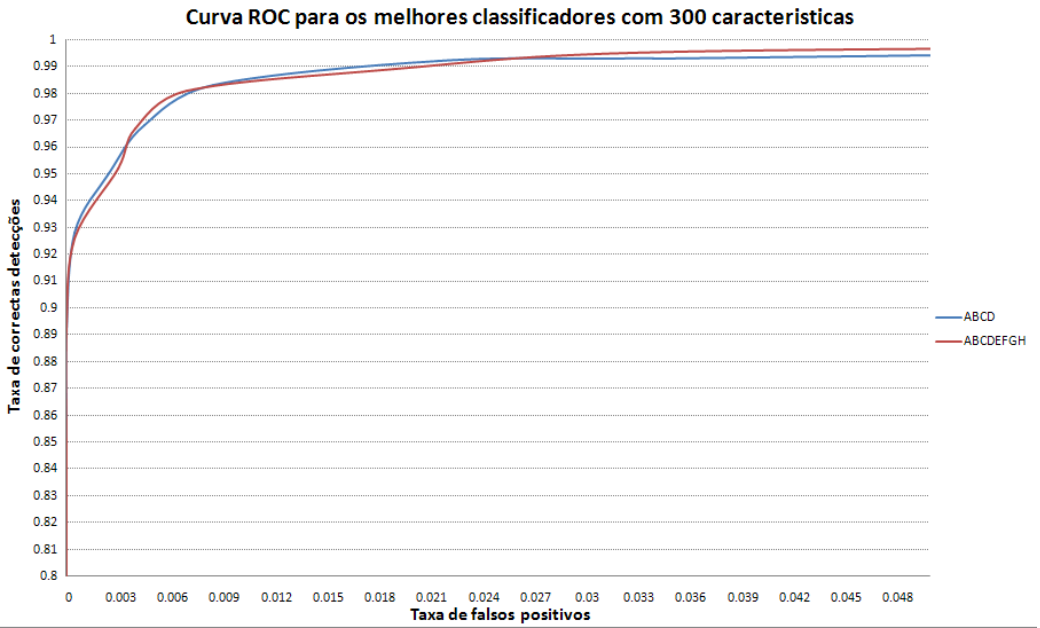


Figura 4.4: *Curva ROC dos dois melhores classificadores fortes com 300 características.*

para um dado factor.

Nos gráficos das figuras 4.5, 4.6, 4.7 e 4.8 não só são apresentadas as taxas de erro para um dado factor, mas também a área de erro (a verde nos gráficos). A área de erro pode ser uma medida tanto ou mais representativa da eficácia de um método, pois o EER poderá ser um pouco superior mas esta ser inferior, o que faz com que exista erro numa área mais reduzida do gráfico.

Na tabela 4.3 são apresentados os melhores valores das áreas de erro, podendo-se observar que estes nem sempre coincidem com os mesmos testes do que os apresentados na tabela dos EER (tabela 4.2), isto é, nem todos os valores mínimos para o EER têm a melhor área de erro.

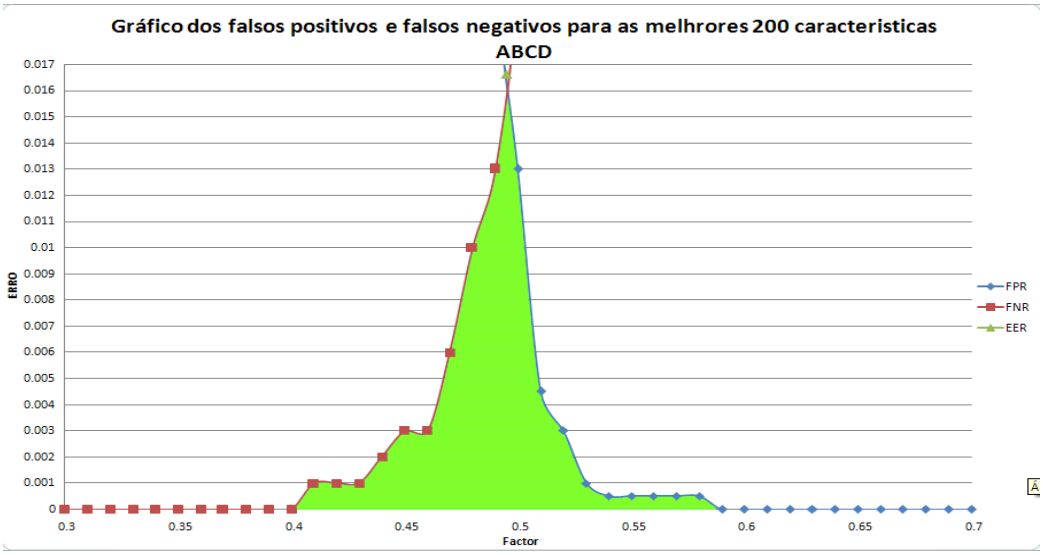


Figura 4.5: *Gráfico entre a taxa dos falsos positivos (FPR) e a dos falsos negativos (FNR) para classificador forte com 200 características do tipo A, B, C e D (Área=0.064 EER=0.016621).*

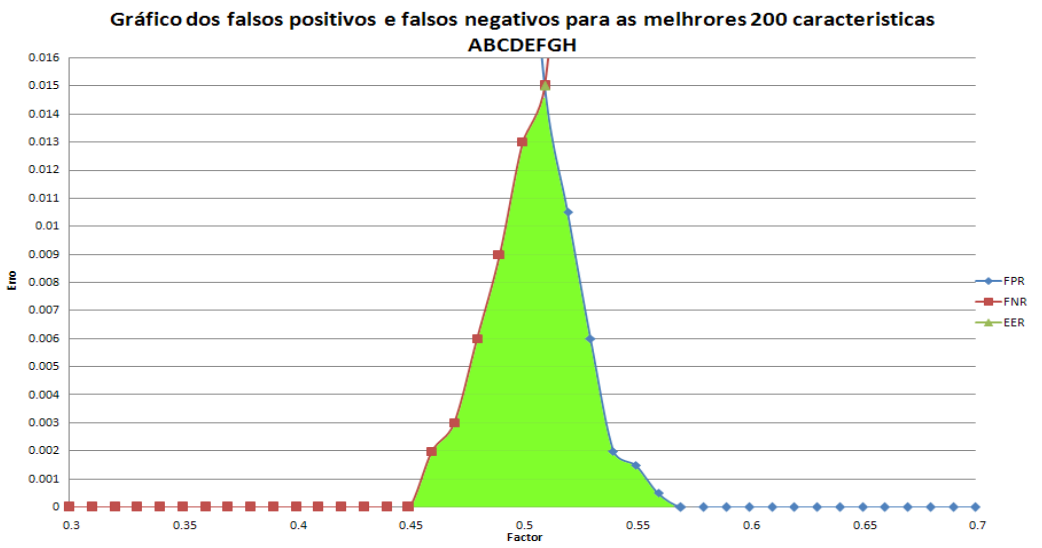


Figura 4.6: *Gráfico entre a taxa dos falsos positivos (FPR) e a dos falsos negativos (FNR) para classificador forte com 200 características do tipo A, B, C, D, E, F, G e H (Área=0.0685 EER=0.015).*

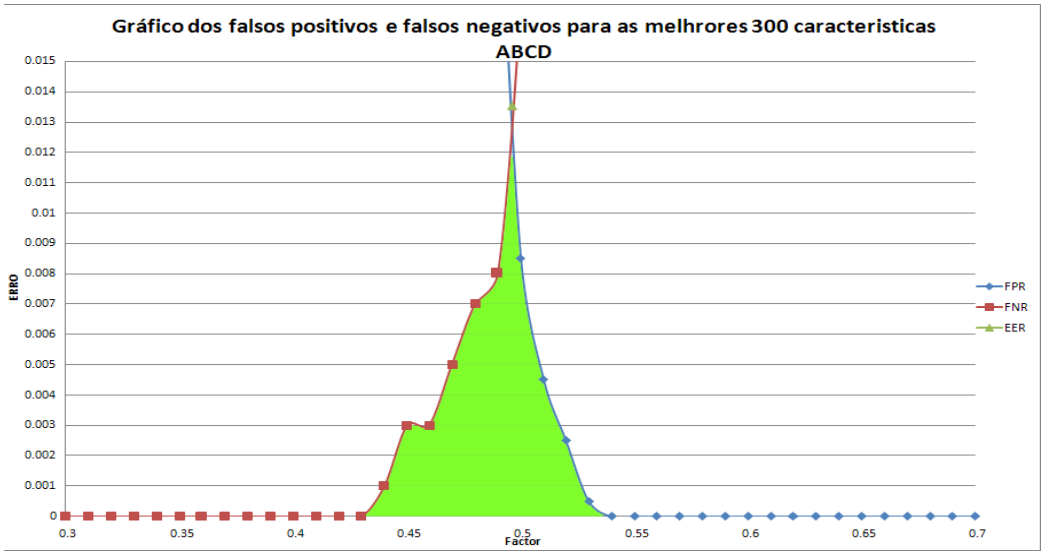


Figura 4.7: *Gráfico entre a taxa dos falsos positivos (FPR) e a dos falsos negativos (FNR) para classificador forte com 300 características do tipo A, B, C e D (Área=0.043 ERR=0.013523).*

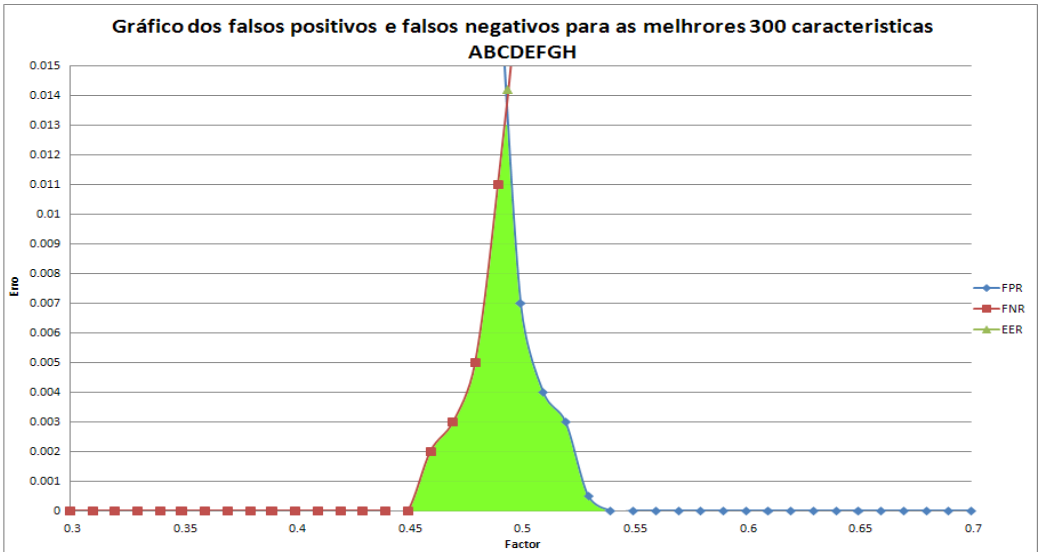


Figura 4.8: *Gráfico dos falsos positivos (FPR) e falsos negativos (FNR) para classificador forte com 300 características do tipo A, B, C, D, E, F, G e H (Área=0.0355 ERR=0.0142).*

Tipo de características	Número de características do classificador forte		
	100	200	300
A, B, C e D	0.104500	0.059500	0.043000
E, F, G e H	0.212500	0.124000	0.097000
A, B, C, D, E, F, G e H	0.099500	0.045500	0.035500

Tabela 4.3: *Valores mínimos das áreas de erro nos testes realizados.*

4.3 Testes ao detector

Nesta secção serão apresentado alguns testes realizados ao detector final. Nestes testes foram usados os dois melhores classificadores (um com apenas as características apresentadas por Viola e Jones [24], classificador usado para gerar os gráficos das figuras 4.4 e 4.7, e o outro que contêm as características deles e as nossas, classificador usado para gerar os gráficos das figuras 4.4 e 4.8) obtidos durante a fase de treino, sendo eles constituídos por 300 características cada um.

Com base no factor, descrito na secção anterior, foram realizados quatro testes para cada uma das imagens. Para cada classificador testado numa imagem foi variado o seu factor em 0.01, esta variação foi feita para se demonstrar que basta uma pequena variação para que este aumente a sua taxa de falsos negativos e falsos positivos, como pode ver nas figuras 4.9, 4.10, 4.11, 4.12 e 4.13. O factor inicial aplicado foi de 0.57, pois neste trabalho interessa ter uma taxa de detecção e falsos positivos razoável.

Nas figuras que compõem a figuras 4.9, o factor predominante no aparecimento de falsas detecções é o vestuário das pessoas. Mesmo com o aumento do factor de 0.57 para 0.58 as falsas detecções continuam a aparecer, mas com a aplicação do classificador composto pelas nossas características estas diminuíram significativamente.

Como se pode notar, nas figuras 4.10(a) e 4.10(c), ouve uma melhoria bastante significativa nas detecções ao aumentar o factor, conseguindo detectar correctamente todas as faces e obtendo apenas uma falsa detecção.

(a) Factor = 0.57 para nas características do tipo ABCD.

(b) Factor = 0.57 para nas características do tipo ABCDEFGH.

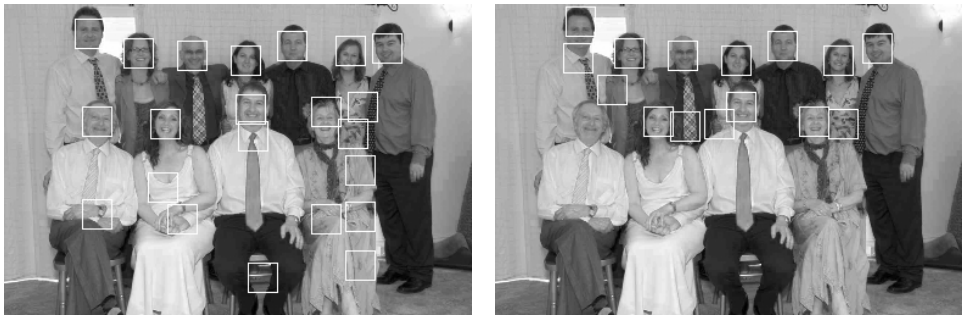
(c) Factor = 0.58 para nas características do tipo ABCD.

(d) Factor = 0.58 para nas características do tipo ABCDEFGH.

Figura 4.9: *Detecção na imagem 1.*

Nem tão boa melhoria foi obtida nas figuras 4.10(b) e 4.10(d), pois já existe uma face que foi perdida mesmo com o factor a 0.57 e continuando ainda com duas falsas detecções.

Nas figuras 4.11(a) e 4.11(c) apenas uma face não é detectada correctamente, existem algumas falsas detecções devido ao padrão que o solo da sala contêm ser composto por quadrados, enquanto que nas figuras 4.11(b) e 4.11(d) isso não é uma problema, mas o factor usado neste classificador já é demasiado elevado para que todas as faces sejam detectadas.



(a) Factor = 0.57 para nas características do tipo ABCD.

(b) Factor = 0.57 para nas características do tipo ABCDEFGH.

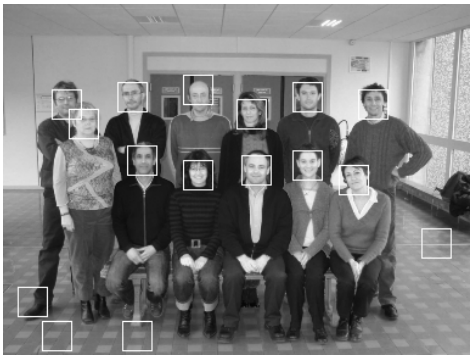
(c) Factor = 0.58 para nas características do tipo ABCD.

(d) Factor = 0.58 para nas características do tipo ABCDEFGH.

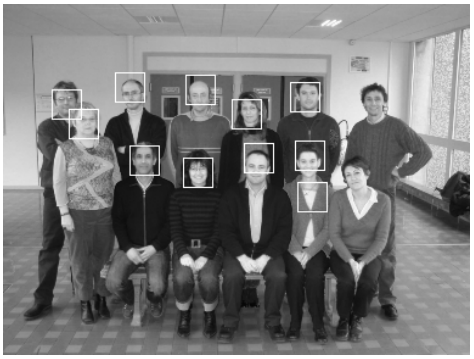
Figura 4.10: *Detecção na imagem 2.*

Nas figuras 4.12(a) e 4.11(b), as taxas de falsas detecções voltam a ser visíveis, devido ao posicionamento das mãos e do vestuário usado palas pessoas. Neste caso, ao ser aumentado o factor todas as falsas detecções, relativas ao vestuário e às mãos, são eliminadas, restando apenas uma indecisão na figura 4.12(c) onde uma face é detectada de forma parcial por duas vezes. É de notar o facto da figura 4.12(d) ser a única que obteve uma taxa de detecção de 100%, isto é, todas as faces foram detectadas com sucesso e não ouve nenhuma falsa detecção.

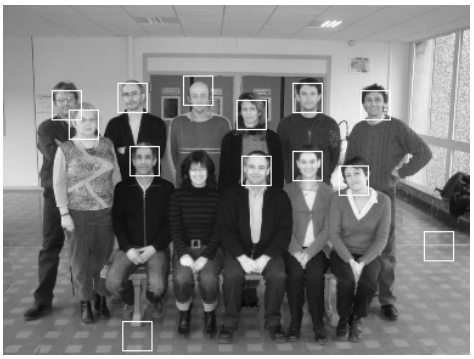
Nas figuras 4.13 podem-se ver que o factor é demasiado elevado para que todas as faces sejam detectadas. Um outro problema detectado nestas imagens é facto de que as falsas detecções não permitirem que a face seja detectada de forma central, devido a que foi imposto uma regra no de-



(a) Factor = 0.57 para nas características do tipo ABCD.



(b) Factor = 0.57 para nas características do tipo ABCDEFGH.

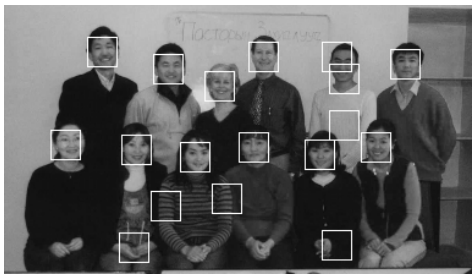


(c) Factor = 0.58 para nas características do tipo ABCD.

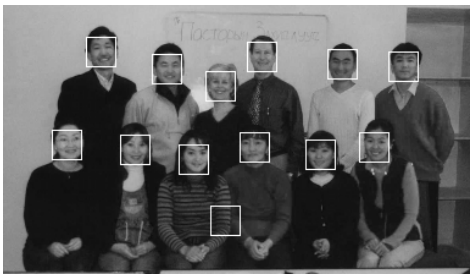
(d) Factor = 0.58 para nas características do tipo ABCDEFGH.

Figura 4.11: *Detecção na imagem 3.*

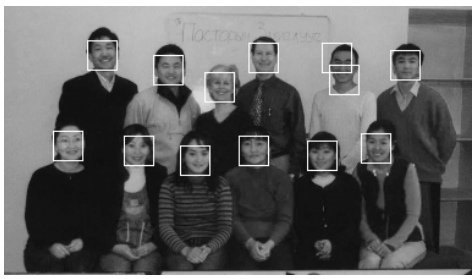
tector para que este não fizesse a sobre posição de detecções, permitindo apenas que 10% da sub-janela já contivesse uma face previamente detectada. Isto faz com que o nosso sistema não seja capaz de detectar faces que estejam muito próximas umas das outras. O ultimo problema detectado nestas imagens é posicionamento das mãos, sendo que o classificador composto pelas nossas características seja mais vulnerável a esse problema.



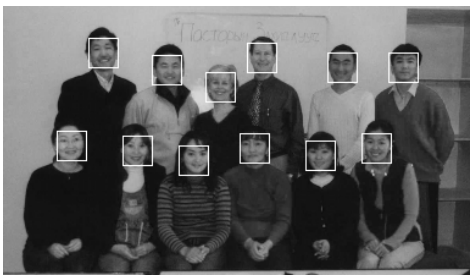
(a) Factor = 0.57 para nas características do tipo ABCD.



(b) Factor = 0.57 para nas características do tipo ABCDEFGH.

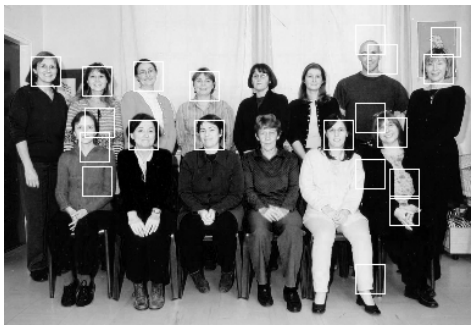


(c) Factor = 0.58 para nas características do tipo ABCD.



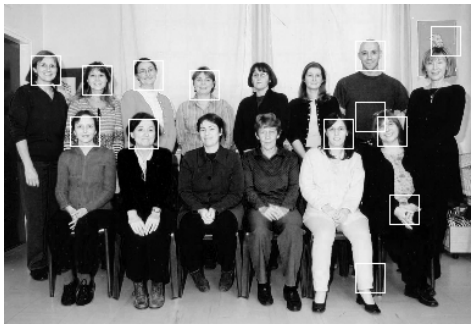
(d) Factor = 0.58 para nas características do tipo ABCDEFGH.

Figura 4.12: *Detecção na imagem 4.*



(a) Factor = 0.57 para nas características do tipo ABCD.

(b) Factor = 0.57 para nas características do tipo ABCDEFGH.



(c) Factor = 0.58 para nas características do tipo ABCD.

(d) Factor = 0.58 para nas características do tipo ABCDEFGH.

Figura 4.13: *Detecção na imagem 5.*

Capítulo 5

Conclusão e trabalho futuro

5.1 Conclusão

Os objectivos propostos para este projecto foram atingidos com sucesso, através da extracção de um conjunto de características baseadas em rectângulos e em triângulos, para a detecção de faces. Uma mais valia do método apresentado é que as características que compõem os classificadores são independentes ao tamanho das sub-janelas das imagens, proporcionando assim que o tempo de análise a uma dada característica seja constante, não variando com o tamanho da sub-janela.

Do ponto de vista científico, os métodos de detecção de faces humanas podem, num futuro muito proximo, vir a apresentar soluções práticas a nível da segurança e vigilância. O facto da criminalidade estar a aumentar faz com que cada vez mais existam pessoas a trabalhar nesta área, com o intuito de desenvolverem sistemas que sejam capazes de detectar e reconhecer um conjunto de pessoas que possam ser prejudiciais (simples ladrões, grupos organizados ou até mesmo terroristas) para a restante comunidade.

Este projecto foi dividido em duas fases, sendo que a primeira foi o estudo do método base desenvolvido por Viola e Jones [24] que usa apenas características baseadas em rectângulos e a segunda foi a substituição e

integração de novas características, estas agora baseadas em triângulos.

Os resultados obtidos para cada uma das abordagens foram satisfatórios. O método proposto não apresentou resultados significativamente melhores devido a que os resultados do método baseado nos rectângulos já serem bons. Com isto, a primeira abordagem obteve um erro de 1.35% e uma área de erro de 4.3%, enquanto que na segunda abordagem com a integração das novas características nas baseadas em rectângulos o erro foi de 1.42% e a área de erro de 3.55%. Estes erros tem por base os dois melhores classificadores, sendo que estes foram construídos a partir de uma selecção parcial das características, isto é, os classificadores apresentados não foram construídos usando o número total de características geradas, mas apenas uma dada percentagem seleccionada aleatoriamente podendo fazer com que algumas das melhores características não constem nos classificadores.

Numa perspectiva mais critica, considero que a fase mais critica deste trabalho é mesmo a fase de treino, visto que se as imagens do conjunto de treino não forem representativas de o local onde o sistema será aplicado, o algoritmo de treino não será capaz de fazer a selecção das melhores características para aquele local. Isto deve-se a que o número de sub-janelas que não contêm faces numa imagem ser muito superior às que contêm.

5.1.1 Trabalho futuro

Numa perspectiva de trabalho futuro podem ser referidas as seguintes tarefas:

- Adicionar um novo tipo de característica;
- Integração de uma câmara de video-vigilância no detector;
- Criar um novo conjunto de treino.

Novo tipo de característica

A proposta para a adição de um novo tipo de característica é uma característica baseada nos triângulos mais seleccionados por parte do algoritmo de treino (características do tipo E e G) a quando da conjugação de todos os tipos de características. Na implementação das características baseadas em triângulos esperava-se que durante o treino, se fosse produzindo algo do género da nova característica do tipo I, demonstrada pela figura 5.1.

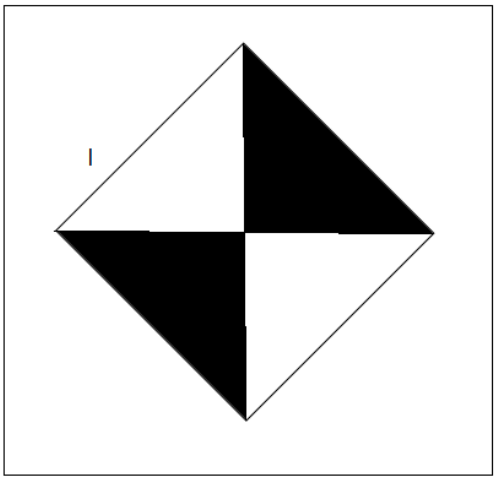


Figura 5.1: *Proposta de um novo tipo de característica, denominada por I.*

Integração de uma câmara de vídeo-vigilância no detector

Com base nos resultados apresentados acha-se que seja viável a aplicação deste método num sistema de vídeo-vigilância. Aplicando-se este método de detecção num sistema de vídeo-vigilância, será possível efectuar o reconhecimento das pessoas, as quais foram previamente detectadas por este método.

Novo conjunto de treino

A criação de um novo conjunto de treino provém da possibilidade deste sistema ser implementado num sistema de vídeo-vigilância. Ao ser criado este novo conjunto de treino optimizado para o local vigiado espera-se que o erro deste sistema seja inferior a 1%, o que não deverá ser muito difícil, visto que este sistema mesmo treinado com imagens retiradas da internet obteve um erro pouco superior a 1%.

Apêndice A

Ambiente de desenvolvimento

A.1 Plataforma de desenvolvimento

O software utilizado para o desenvolvimento do código foi o Borland C++ Builder Enterprise (versão 6.0 build 10.160), que permite ao utilizador desenvolver as mais variadas aplicações em C++. No uso deste software foi necessário ter um cuidado redobrado, visto que o compilador que vem com esta API não aceita apenas código C++ que seguem as normas da *American National Standards Institute* (ANSI) para o C/C++, o que era um requisito deste projecto. Para a certificação que o código seguia as normas foi usado o compilador da *GNU Compiler Collection* (GCC) [2] para C++.

A.2 Linguagem

Este projecto foi desenvolvido em C++, uma linguagem com base em C mas orientada a objectos. Esta linguagem é uma linguagem de programação de alto nível com facilidades do uso em baixo nível multi-paradigma e de uso geral [27]. A componente de baixo nível desta linguagem é muito importante, pois todos os projectos que pretendam ter uma resposta em tempo real terão de saber lidar com componentes muito próximas do processador para que possam dar resposta em tempo útil. Para este tipo de

problemas nunca se poderia usar uma linguagem que fosse compilada para uma máquina virtual, pois estas linguagens perdem muito tempo na interacção entre a máquina virtual e o próprio processador, não falando que se tornam muito mais pesadas.

A.3 Biblioteca de tratamento de imagens

Para a implementação do projecto utilizou-se, a biblioteca *The C++ Template Image Processing Library* (CImg Lib) [23], que é uma ferramenta open-source para o processamento de imagem consistindo num único arquivo de cabeçalho "CImg.h"que contém inúmeros métodos e classes em C++. Desta forma é possível utilizar directamente no código de cada um os métodos e classes de C++, uma vez que as suas operações são bastante variadas. A título de exemplo enuncia-se a abertura, os histogramas, as transformações, os cortes, entre outras. Com a ajuda desta poderosa ferramenta foram realizadas as várias operações de abertura, de corte, de gravação.

Apêndice B

Publicação

Neste anexo inclui-se a publicação proposta para a ICSP'08 [1]. De notar que os resultados contidos nele diferirem um pouco dos que foram apresentados neste relatório, devido ao facto de quando foi realizado ainda não teremos os testes todos realizados.

Bibliografia

[1] *9th International Conference on Signal Processing (ICSP'08)*, 2008. `http://icsp08.bjtu.edu.cn/`.

[2] *The GNU Compiler Collection*, 2008. `http://gcc.gnu.org/`.

[3] Y. Amit, D. Geman, and X. Fan. A coarse-to-fine strategy for multi-class shape detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(12):1606–1621, Dec. 2004.

[4] J. Cai and A. Ardeshir Goshtasby. Detecting human faces in color images. *Image Vision Comput.*, 18(1):63–75, 1999.

[5] M. Castrillón, O. Déniz, C. Guerra, and M. Hernández. Encara2: Real-time detection of multiple faces at different resolutions in video streams. *J. Vis. Comun. Image Represent.*, 18(2):130–140, 2007.

[6] Qian Chen, Haiyuan Wu, and M. Yachida. Face detection by fuzzy pattern matching. In *ICCV '95: Proceedings of the Fifth International Conference on Computer Vision*, page 591, Washington, DC, USA, 1995. IEEE Computer Society.

[7] Farhad Dadgostar and Abdolhossein Sarrafzadeh. An adaptive real-time skin detector based on hue thresholding: A comparison on two motion tracking methods. *Pattern Recogn. Lett.*, 27(12):1342–1352, 2006.

[8] Farhad Dadgostar and Abdolhossein Sarrafzadeh. An adaptive real-time skin detector based on hue thresholding: A comparison on two motion tracking methods. *Pattern Recogn. Lett.*, 27(12):1342–1352, 2006.

[9] Ying Dai and Yasuaki Nakano. Face-texture model based on sgld and its application in face detection in a color scene. *Pattern Recognition*, 29(6):1007–1017, 1996.

[10] B. Dawson. *Biometrics measures physical traits. Vision Systems Design.*, 2 2001. `http://www.vision-systems.com/display_article/96566/19/none/none/Feat/Biometrics-measures-physical-traits`.

[11] David A. Forsyth, Okan Arikan, Leslie Ikemoto, James O'Brien, and Deva Ramanan. Computational studies of human motion: part 1, tracking and motion synthesis. *Found. Trends. Comput. Graph. Vis.*, 1(2-3):77–254, 2005.

[12] William T. Freeman and Edward H. Adelson. The design and use of steerable filters. *IEEE Trans. Pattern Anal. Mach. Intell.*, 13(9):891–906, 1991.

[13] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *EuroCOLT*, pages 23–37, 1995.

[14] H. Greenspan, S. Belongie, R. Goodman, P. Perona, S. Rakshit, and C.H. Anderson. Overcomplete steerable pyramid filters and rotation invariance. *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR '94., 1994 IEEE Computer Society Conference on*, pages 222–228, 21-23 Jun 1994.

[15] Erik Hjelmås and Boon Kee Low. Face detection: A survey. *Computer Vision and Image Understanding*, 83(3):236–274, September 2001.

[16] Shi-Hong Jeng, Hong Y. Liao, Chin C. Han, Ming Y. Chern, and Yao T. Liu. Facial feature detection using geometrical face model: An efficient approach. *Pattern Recognition*, 31(3):273–282, March 1998.

[17] Heung Soo Kim, Eun Yi Kim, Sang Won Hwang, and Hang Joon Kim. Object-based human face detection. *Consumer Electronics, 2000. ICCE. 2000 Digest of Technical Papers. International Conference on*, pages 354–355, 2000.

[18] Chiunhsiun Lin and Kuo-Chin Fan. Triangle-based approach to the detection of human face. *Pattern Recognition*, 34(6):1271–1284, 2001.

[19] Kang Ryoung Park, Hyun-Ae Park, Byung Jun Kang, Eui Chul Lee, and Dae Sik Jeong. A study on iris localization and recognition on mobile phones. *EURASIP J. Adv. Signal Process*, 2008(1):1–7, 2008.

[20] Henry Rowley, Shumeet Baluja, and Takeo Kanade. Neural network-based face detection. In *Computer Vision and Pattern Recognition '96*, June 1996.

[21] Kah Kay Sung and Tomaso Poggio. Example-based learning for view-based human face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1):39–51, 1998.

[22] Patrick Tresset and Frederic Fol Leymarie. Aikon: the artistic/automatic ikonograph. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Research posters*, page 37, New York, NY, USA, 2006. ACM.

[23] David Tschumperlé. *The C++ Template Image Processing Library*, Fevereiro 2008. `http://cimg.sourceforge.net/`.

[24] Paul Viola and Michael J. Jones. Robust real-time face detection. *International Jornal of Computer Vision*, 57(2):137–154, 2004.

[25] Dirk Walther, Ueli Rutishauser, Christof Koch, and Pietro Perona. Selective visual attention enables learning and recognition of multiple objects in cluttered scenes. *Comput. Vis. Image Underst.*, 100(1-2):41–63, 2005.

[26] Jianguo Wang and Tieniu Tan. A new face detection method based on shape information. *Pattern Recogn. Lett.*, 21(6-7):463–471, 2000.

[27] Wikipédia. *C++*, 2008. `http://pt.wikipedia.org/wiki/C%2B%2B`.

[28] Chenyu Wu, Ce Liu, Heung-Yueng Shum, Ying-Qing Xy, and Zhengyou Zhang. Automatic eyeglasses removal from face images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(3):322–336, Mar 2004.

[29] K. Yow and R. Cipolla. Feature-based human face detection, 1996.