PSI5796: Processamento e Análise de Imagens e Vídeos
Segundo período de 2017 2º exercício-programa Prof. Hae
Data de entrega: 10/09/2017 (domingo) até 24:00 horas
Nota: Na classe tínhamos combinado data de entrega para 3/9. Resolvi dar mais uma semana.
Obs. 1: Cada dia de atraso acarreta uma perda de 1 ponto no exercício.
Obs. 2: Este EP deve ser resolvido individualmente. EPs iguais receberão nota zero.

Considere o bancos de dados "Yale face database B cropped". Este banco de imagens contém 2414 imagens (com 192x168 pixels) de 38 indivíduos, alinhadas e recortadas manualmente. Há mudanças grandes de condição de iluminação.

Original: http://vision.ucsd.edu/~leekc/ExtYaleDatabase/ExtYaleB.html
Cópia local: http://www.lps.usp.br/hae/psi5796/ep2-2017/cropped_yale.zip

1. Nota: As imagens "Ambient" não fazem parte do banco de dados.
2. Nota: Os indivíduos estão numerados de 1 a 39, porém não há indivíduo número 14, resultando em 38 indivíduos.
3. Nota: Há 64 fotos de cada indivíduo, mas há 18 fotos marcadas como "bad", resultando em 38x64-18=2414 fotos.

O objetivo deste exercício é fazer dois programas C++ ep50.cpp e ep05.cpp para reconhecer as faces usando alguma técnica de extração de características e aprendizagem de máquina. Os dois programas devem ser praticamente iguais, diferindo somente na escolha dos parâmetros. Para isso, separei as imagens em dois conjuntos disjuntos "treino" e "teste" (aproximadamente 50% para cada conjunto) e armazenei seus nomes nos arquivos treino50.csv e teste50.csv. Em cada linha desses arquivos há a informação:

"subDiretório/nomeImagem;classificação".

Você deve usar esses arquivos para ler as imagens de treino e de teste com as suas classificações corretas.

Também criei arquivos "treino05.csv" e "teste05.csv" com a divisão treino/teste em aproximadamente 5% e 95% (resultando 3 imagens de treino por pessoa). Fazer o reconhecimento de face fica bem mais difícil quando há poucas imagens de treino.

Para facilitar a correção, os seus programas devem se chamar obrigatoriamente ep50 e ep05. O programa ep50 deve ter os parâmetros otimizados para maximizar os acertos usando treino50.csv e teste50.csv. O programa ep05 deve ter os parâmetros otimizados para usar treino05.csv e teste05.csv. A forma de executá-los deve ser obrigatoriamente:

```
diretorio>ep?? diretorio_do_banco_de_dados treino??.csv teste??.csv
```

Por exemplo:

```
c:\win_diret>ep50 c:\diretorio\cropped_yale treino50.csv teste50.csv
~/linux_diret$ ep05 ~/facerec/cropped_yale treino05.csv teste05.csv
```

Os programas devem imprimir os nomes dos arquivos classificados incorretamente, a classificação correta e a classificação atribuída pelo algoritmo. Também deve imprimir a taxa de erro e o tempo de execução. Por exemplo:

```
~/pos2017$ crono ep50 ~/facerec/cropped_yale treino50.csv teste50.csv
Arquivo yaleB01/yaleB01_P00A+130E+20.pgm. Correta=1. Classificada=16.
(...)
Arquivo yaleB39/yaleB39_P00A+095E+00.pgm. Correta=39. Classificada=15.
totalErros=27 nTestes=1209 mediaErros=2.23%
Tempo gasto     11.8 segundos
```

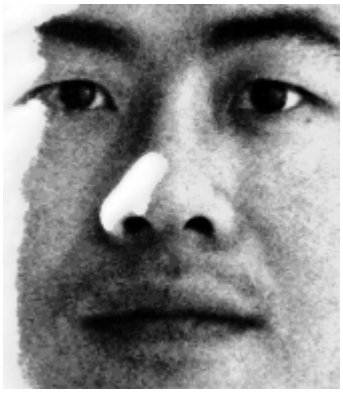
Nota 1: Em OpenCV há três classes prontas para reconhecer faces:

```
http://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_api.html
http://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_tutorial.html
```

Vamos fazer este exercício ***sem usar essas classes*** ou qualquer outra biblioteca ou função pronta reconhecer faces. Neste exercício, é permitido:

- Usar as classes prontas para aprendizagem de máquina de OpenCV ou de qualquer outra biblioteca.
- Ler artigos sobre reconhecimento de face e implementá-las.

Nota 2: A qualidade perceptual das imagens melhora substancialmente se você fizer equalização de histograma.

 Imagem original.	 Depois da equalização de histograma.

Nota 3: Há várias formas de cronometrar o tempo de processamento em C++. Se você instalou Cekeikon, o programa "crono" pode ser usado para cronometrar o tempo de processamento:

```
diretorio>crono seu_programa argumentos
```

Em Linux, há o comando "time":

```
diretorio$ time seu_programa argumentos
```

Se quiser colocar cronômetro dentro do programa C++, em Cekeikon pode usar:

```
TimePoint t1=timePoint();
(processamento)
TimePoint t2=timePoint();
double t=timeSpan(t1,t2);
//Aqui, t contém o número de segundos decorridos no processamento.
```

Nota 4: Para se ter uma ideia, copio abaixo as taxas de erros que obtive usando um algoritmo simples de aprendizagem e usando as 3 classes prontas do OpenCV.

taxa de erros / tempo de proc.	50% de imagens de treino	5% de imagens de treino
Algoritmo simples	2,2% / 12s	44,9% / 2,7s
EigenFaces	1,9% / 3,5min	37,3% / 9,0s
FisherFaces	0,17% / 2,3min	7,1% / 3,1s
LBPH	2,07% / 27s	29,9% / 5,6s

Se alguém quiser implementar o método FisherFaces, veja:

Belhumeur, P. N., Hespanha, J., and Kriegman, D. Eigenfaces vs. Fisherfaces: Recognition Using ClassSpecific Linear Projection. IEEE Transactions on Pattern Analysis and Machine Intelligence 19, 7 (1997),711–720.

Se alguém quiser implementar o método LBPH, veja:

Ahonen, T., Hadid, A., and Pietikainen, M. Face Recognition with Local Binary Patterns. Computer Vision - ECCV 2004 (2004), 469–481.

Obs. 1: Pode usar (se quiser) a biblioteca Cekeikon/OpenCV.

Obs. 2: Entregue os programas-fontes (ep50.cpp e ep05.cpp) e um documento PDF (relatorio.pdf) ou DOC (relatorio.doc) com os comentários descrevendo o funcionamento do programa. Você deve entregar obrigatoriamente o relatório (veja o anexo). Destaque as taxa de erros obtidas nos dois casos.

(a) Se você fez o programa no ambiente usado na classe (Cekeikon/OpenCV em Windows ou Linux), não é necessário enviar o programa executável. Se você fez o programa usando só OpenCV, também não é necessário enviar o programa executável.

(b) Se você quiser usar alguma biblioteca diferente, converse antes com o professor.

Obs. 3: Compacte todos os arquivos como SeuNome_Sobrenome.ZIP e envie um email para:

- hae@lps.usp.br

Se você enviar mais de um email, considerarei somente o último email enviado, apagando os anteriores.

Anexo: Relatórios dos exercícios programas

A primeira regra para uma comunicação clara é escrever para a sua audiência (http://www.plainlanguage.gov/howto/guidelines/FederalPLGuidelines/think.cfm). No caso do relatório, a sua audiência será o professor ou o monitor que irá corrigir o seu exercício. Assim, você não precisa escrever informações que são óbvias para professor/monitor. Por outro lado, precisa escrever aqueles dados que ajudarão professor/monitor a avaliar corretamente a sua solução. Seja breve. Passse as informações necessárias usando menos palavras/páginas possível.

Identificação

Seu nome, número USP, nome da disciplina, etc.

Breve enunciado do problema

Apesar do enunciado do problema ser conhecido ao professor/monitor, descreva brevemente o problema que está resolvendo. Isto tornará o documento compreensível para alguma pessoa que não tem o enunciado do EP à mão.

Técnica(s) utilizada(s) para resolver o problema

Descreva quais técnicas você usou para resolver o problema. Se você mesmo inventou a técnica, descreva em português a sua ideia, deixando claro que a ideia foi sua. Se você utilizar oração com sujeito indeterminado ou na voz passiva, o leitor pode não entender quem foi o responsável (Ex: "Criou-se um novo algoritmo" - Quem criou? Você? Ou algum autor da literatura científica? http://www.plainlanguage.gov/howto/guidelines/FederalPLGuidelines/writeActive.cfm). Se você utilizou alguma técnica já conhecida, utilize o nome próprio da técnica (por exemplo, filtragem Gaussiana, algoritmo SIFT, etc.) juntamente com alguma referência bibliográfica onde a técnica está descrita. Use elementos gráficos como imagens intermediárias e diagramas, pois ajudam muito a compreensão. Não "copie-e-cole" código-fonte, a não ser que seja relevante. Use preferencialmente o pseudo-código.

Ambiente de desenvolvimento utilizado

Em qual plataforma você desenvolveu o programa? Como o professor/monitor pode compilar o programa? Você utilizou que bibliotecas?

Operação

Como o professor/monitor pode executar o programa? Que argumentos são necessários para a execução do programa? Há parâmetros que devem ser configurados? Quais arquivos de entrada são necessários? Quais arquivos de saída são gerados?

Resultados Obtidos

Descreva os resultados obtidos. Qual é o tempo de processamento típico? O problema foi resolvido de forma satisfatória?

Referências

Descreva o material externo utilizado, como livros/artigos consultados, websites visitados, etc.