PSI-3471: Fundamentos de Sistemas Eletrônicos Inteligentes
Primeiro semestre de 2021 Exercício-programa Prof. Hae
Data de entrega: 14/07/2021 (quarta-feira) até 23:59 horas. Enunciado revisado: 26/06/2021
Obs: Este EP deve ser resolvido de preferência em duplas.

Este problema foi sugerido pela Samsung SDS Logistics, para que os alunos tenham um contato mais próximo com os problemas reais das empresas:

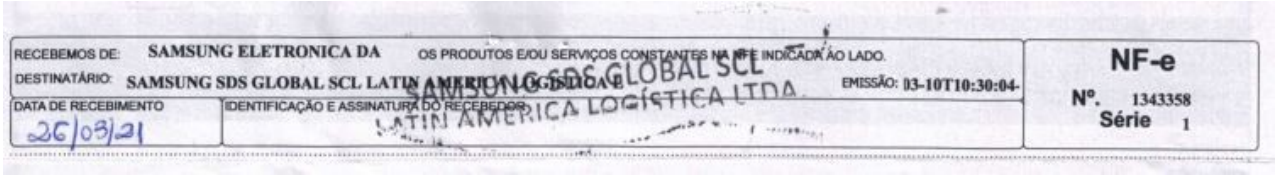
https://www.samsungsds.com/en/index.html
https://www.samsungsds.com/la/index.html

O banco de dados abaixo contém imagens de POD (Proof Of Delivery ou canhoto de recebimento de mercadoria) utilizadas pela Samsung:

http://www.lps.usp.br/hae/psi3471/ep1-2021/POD.zip

Há três modelos de canhoto, numerados de 2 a 4:

m2_001.jpg



m3_002.jpg

m4_001.jpg

Figura 1

Os nomes dos arquivos mA_BBB.jpg no banco de dados significam:

- A: O modelo do POD (número de 2 a 4).
- BBB: O número sequencial das imagens, começando por 001.

A Samsung tem que olhar cada canhoto para verificar se deve aprová-lo ou não. O objetivo é escrever um programa que facilite este trabalho.

Neste exercício, vamos fazer alguns pré-processamentos necessários para resolver este problema. Vamos fazer um programa que:

1) Dada a imagem de um canhoto, reconhece o modelo (2 a 4) do canhoto.
2) Alinha a imagem do canhoto com a imagem do canhoto "padrão", sobrepondo as duas.

A figura abaixo mostra os 3 modelos de canhotos "padrões". Eles foram gerados manualmente a partir de canhotos reais e contém somente pixels com valores 0, 255 e 128, sendo que 128 indica que os valores nesses pixels são indiferentes (don't care). Se você preferir, pode gerar os seus próprios canhotos "padrões" ou modificar as imagens dadas. É possível usar este canhotos "padrões" e fazer casamento de modelos para resolver este problema. O programa deve testar diferentes escalas e rotações dos padrões, procurando a transformação geométrica que resulta no melhor casamento. Vamos ignorar as distorções em perspectiva.

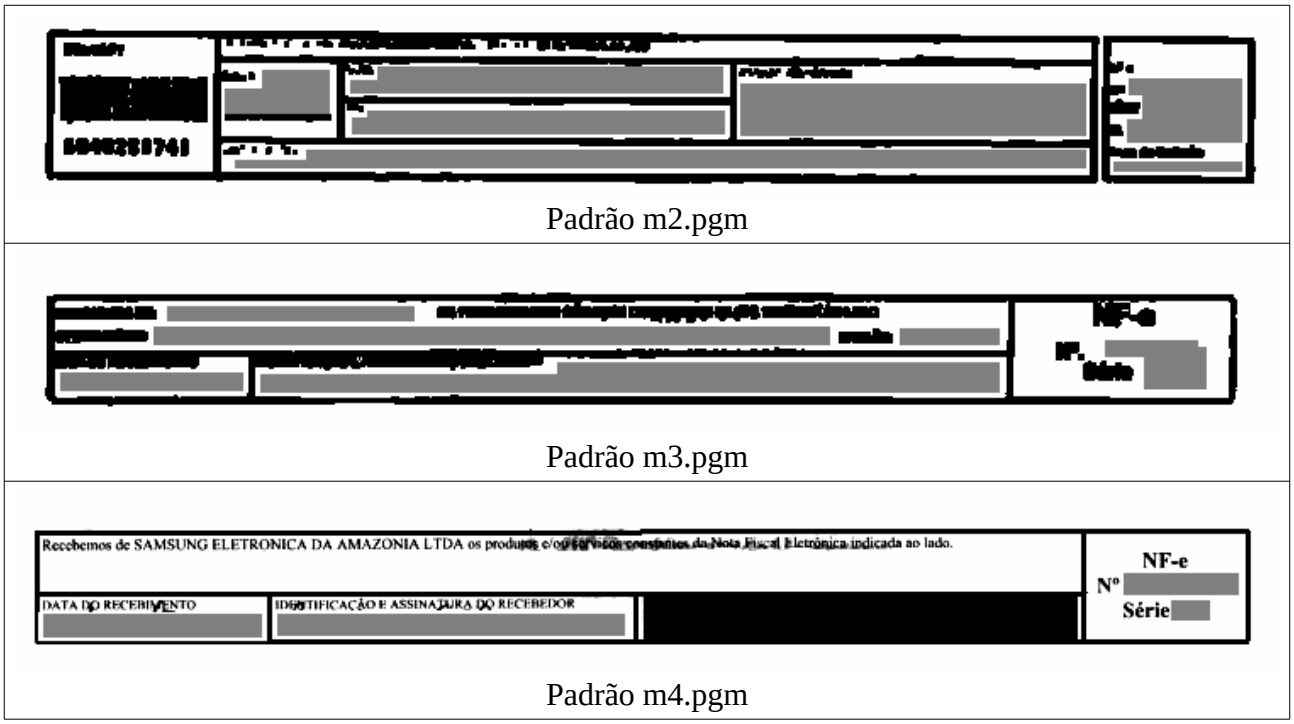


Figura 2. Os três canhotos “padrões”, onde cinza (nível 128) indica pixels cujos valores são indiferentes (“don’t care”).

Faça um programa C/C++ ep.cpp (ou Python ep.py) que, dada a imagem de um canhoto, reconhece o modelo e gera a imagem do canhoto sobreposta ao canhoto “padrão”, como na figura 3.

A forma de chamar o programa deve ser “ep entrada.jpg saida.png”, por exemplo:

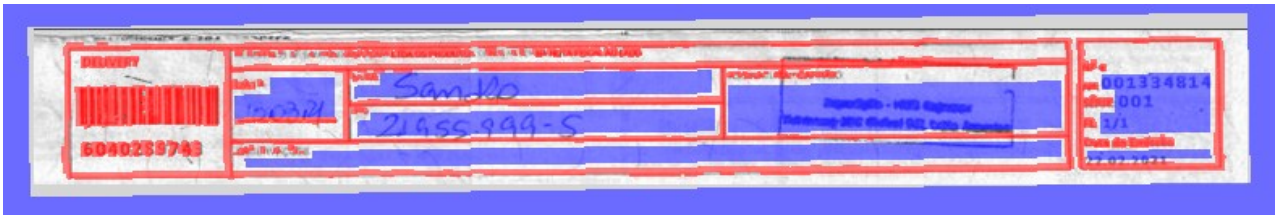
```
diretorio> ep m2_001.jpg m2_001.png
diretorio> python3 ep.py m2_001.jpg m2_001.png
```

O programa deve responder algo como:

```
Maior correlacao entre m2_001.jpg e m2.pgm: 0.147
Maior correlacao entre m2_001.jpg e m3.pgm: 0.064
Maior correlacao entre m2_001.jpg e m4.pgm: 0.078
melhorModelo=2 corr= 0.147 graus= 0.750 fator= 1.006 desloc(x,y)=[ -2,  -2]
```

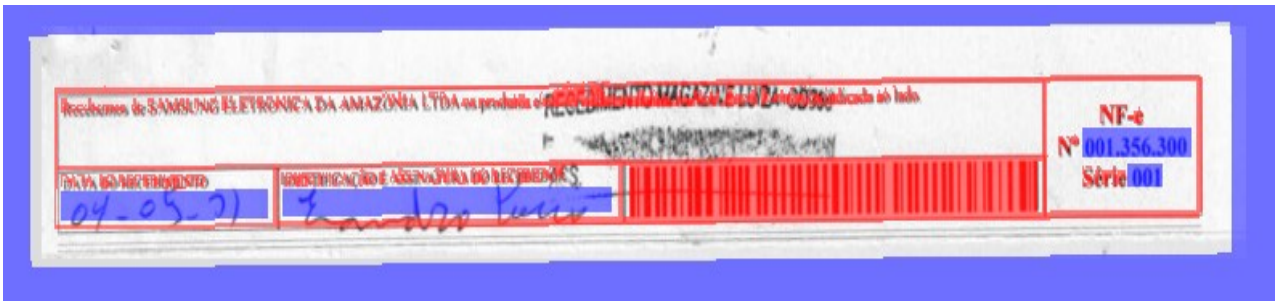
Significando que a maior correlação entre o canhoto m2_001.jpg e os padrões m2, m3 e m4 foram respectivamente 0.147, 0.064 e 0.078. A maior correlação foi obtida com o padrão m2, portanto o canhoto deve ser classificado como sendo modelo 2. Além disso, para obter o casamento ótimo, o programa teve que rotacionar o padrão 2 em 0.750 graus, ampliá-lo pelo fator 1.006 e deslocá-lo de (-2, -2) pixels.

Essas informações de transformações geométricas podem ser usadas para sobrepor as duas imagens (canhoto e padrão). Uma vez que se descobre a transformação geométrica para alinhar as duas imagens, será possível (no futuro, não neste EP) verificar se o canhoto está assinado, se a data foi preenchida, etc.



m2_001.png

m3_002.png



m4_001.png

Figura 3

Teste o seu programa para todos os 9+3+8 canhotos do BD e informe no vídeo e no relatório se o seu programa consegue alinhar e reconhecer corretamente os modelos de todos esses canhotos.

Evidentemente, o seu programa não pode usar as informações que constam nos nomes das imagens para gerar as respostas.

Se chamar o programa com número de parâmetros incorretos, o programa deve imprimir alguma mensagem amigável:

```
diretorio> ep
sintaxe: ep canhoto.jpg
Erro: Numero de argumentos invalido
```

Se o arquivo a ser processado não existe, o programa também deve imprimir alguma mensagem amigável:

```
diretorio> ep lixo.ext
Erro: Nao encontrei o arquivo lixo.ext.
```

Obs. 1: Entregue o programa-fonte (ep1.cpp ou ep1.py) e outros arquivos necessários para executar o seu programa (por exemplo, m2.pgm, m3.pgm e m4.pgm se você os alterou). Entregue um vídeo no máximo 3 minutos e um relatório de no máximo 4 páginas (veja o anexo) mostrando o programa e explicando o seu funcionamento. O envio do vídeo e do relatório são obrigatórios. Pode haver desconto na nota se enviar vídeo de mais de 3 minutos ou relatório de mais de 4 páginas.

(a) Se você fez o programa no ambiente usado na classe (C++/Cekeikon/OpenCV ou Python3/OpenCV), basta entregar o programa-fonte para poder corrigir o seu programa.

(b) Se você quiser usar alguma biblioteca diferente ou programar em outro ambiente, converse antes com o professor.

Obs. 2: Compacte todos os arquivos como nome1_sobrenome1_nome2_sobrenome2.zip e envie o arquivo através de edisciplinas. Dentro do prazo, você pode substituir o arquivo anterior por um novo. Só o último arquivo entregue será corrigido.

A minha solução:

Para ajudar os alunos na resolução deste exercício, vou explicar abaixo como resolvi este problema. Vocês podem usar soluções completamente diferentes.

Resolvi o problema usando casamento de modelos (template matching). Além do casamento de modelos, outras possíveis técnicas que podem ser usadas para resolver este problema incluem:

1. Usar transformada de Hough para detectar as retas.
2. Usar algum detector de pontos-chaves (como SIFT), fazer casamento dos pontos-chaves e utilizar transformada de Hough generalizada para fazer o casamento global.
3. Usar rede neural convolucional.

As saídas do meu programa estao em:

http://www.lps.usp.br/hae/psi3471/ep1-2021/saida.zip

A minha solução reconheceu corretamente o modelo de todos os canhotos. Também conseguiu alinhar todos os canhotos, exceto o canhoto m4_002.jpg que está um pouco desalinhado. Descrevo a minha solução abaixo. Essa descrição só será compreensível para os alunos que tiverem acompanhado as minhas aulas:

1) Note que os três canhotos “padrões” possuem dimensão 750x110.

2) Estou implicitamente supondo que os canhotos foram fotografados por inteiro, isto é, sem pedaços faltando. Também estou supondo que a quantidade de colunas antes e depois da imagem do canhoto não é muito grande.

3) O programa começa lendo a imagem do canhoto como matriz ponto flutuante (0=preto, 1=branco). Usando a suposição 2, redimensiono a imagem do canhoto para que tenha 750 colunas, para que fique numa escala semelhante às dos canhotos “padrões”. O número de linhas é calculado para manter “aspect ratio”.

4) Como o número de colunas tanto de A (canhoto redimensionado) quanto Q (canhoto padrão) é 750, não é possível deslocar Q horizontalmente para produrar o melhor casamento. Para que Q possa se deslocar, coloco A centralizada dentro de uma “imagem moldura”. Usei moldura 310x1150, mas possivelmente poderia usar moldura menor. A moldura é preenchida com o nível de cinza médio de A para que perturbe minimamente o casamento de modelos.

5) Para cada um dos 3 canhotos padrões faço os passos 6-8 abaixo.

6) Leio o modelo m?.pgm como matriz ponto flutuante. Coloco essa matriz centralizada dentro de uma “imagem moldura” de dimensão 210x1050 preenchida com 128.0/255.0 (valor do pixel don’t care), obtendo imagem modelo Q. Note que moldura de Q é menor que A, permitindo deslocar Q dentro de A.

7) Dentro de um vetor de matrizes ponto flutuante, coloco todas as combinações de Q redimensionadas nas escalas de 0,88 a 1,1 e rotacionadas de -3 a +3 graus. Usei 21 passos de escala em progressão geométrica e 17 passos de rotação. Todas as transformações geométricas de Q devem ser feitas usando interpolação “vizinho mais próximo”. Pense por quê. É possível distorcer Q e logo em seguida calcular “template matching” sem ter que armazenar a imagem distorcida. Se quiser armazenar as imagens de Q distorcidas, você pode usar vetor de matrizes ponto flutuante:

```
vector< Mat_<FLT> >
```

Outra possibilidade é usar Mat_ ou Mat tridimensional (veja manual de OpenCV).

8) Calculo template matching “CC” no modo “same” entre A e cada Q distorcido. É necessário pré-processar Q fazendo correção de média e obrigando a soma absoluta dar 2. Calculo a maior correlação e os índices (escala, rotação, deslocamentos x e y) que resultaram na maior correlação.

9) Para acelerar o cálculo de template matching, sugiro que use OpenCV v3:

```
compila ep -c -v3
```

Template matching é consideravelmente mais rápido no OpenCV v3 (comparado com OpenCV v2). Outra possibilidade de acelerar template matching é usar processamento paralelo, mas esta técnica não foi vista nesta disciplina (para alunos de Eletrônica e Sistemas Computacionais, veremos na disciplina PSI3422 Lab. Sistemas Eletrônicos).

10) Após processar os 3 canhotos padrões (linhas 6-8), tenho os 3 maiores valores de correlação obtidos usando os 3 padrões. Calculo o maior entre essas 3 correlações. Com isso, obtenho o modelo do canhoto, juntamente com o valor da maior correlação e os parâmetros da transformação geométrica para alinhar o canhoto e o padrão (escala, rotação, deslocamentos x e y).

11) Imprimo os parâmetros obtidos. Imprimo a imagem do canhoto sobreposta à imagem do padrão.

Anexo: Relatórios dos exercícios programas

O mais importante numa comunicação escrita é que o leitor entenda, sem esforço e inequivocamente, o que o escritor quis dizer. O texto ficar “bonito” é um aspecto secundário. Se uma (pseudo) regra de escrita dificultar o entendimento do leitor, essa regra está indo contra a finalidade primária da comunicação. No site do governo americano [1], há regras denominadas de “plain language” para que comunicações governamentais sejam escritas de forma clara. As ideias por trás dessas regras podem ser usadas em outros domínios, como na escrita científica. Resumo abaixo algumas dessas ideias.

(1) Escreva para a sua audiência. No caso do relatório, a sua audiência será o professor ou o monitor que irá corrigir o seu exercício. Você deve enfocar na informação que o seu leitor quer conhecer. Não precisa escrever informações que são inúteis ou óbvias para o seu leitor.

(2) Organize a informação. Você é livre para organizar o relatório como achar melhor, porém sempre procurando facilitar o entendimento do leitor. Seja breve. Quebre o texto em seções com títulos claros. Use sentenças curtas. Elimine as frases e palavras que podem ser retiradas sem prejudicar o entendimento. Use sentenças em ordem direta (sujeito-verbo-predicado).

(3) Use palavras simples. Use o tempo verbal o mais simples possível. Evite cadeia longa de nomes, substituindo-os por verbos (em vez de “desenvolvimento de procedimento de proteção de segurança de trabalhadores de minas subterrâneas” escreva “desenvolvendo procedimentos para proteger a segurança dos trabalhadores em minas subterrâneas”). Minimize o uso de abreviações (para que o leitor não tenha que decorá-las). Use sempre o mesmo termo para se referir à mesma realidade (pode confundir o leitor se usar termos diferentes para se referir a uma mesma coisa). O relatório não é obra literária, não tem problema repetir várias vezes a mesma palavra.

(4) Use voz ativa. Deixe claro quem fez o quê. 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?). O site diz: “Passive voice obscures who is responsible for what and is one of the biggest problems with government writing.”

(5) Use exemplos, diagramas, tabelas, figuras e listas. Ajudam bastante o entendimento.

O relatório deve conter pelo menos as seguintes informações:

Identificação

Nomes dos integrantes do grupo, números 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 a sua ideia, deixando claro que a ideia foi sua. 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.

1 https://plainlanguage.gov/guidelines/

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.