CIÊNCIAS DA COMPUTAÇÃO

Processamento de Imagens
e
Visão Computacional

Prof. César C. Xavier

Processamento de Imagens e Visão Computacional

ROTEIRO

Pré-Processamento

- Transformações Geométricas
  - Rotação
  - Translação
  - Escala
  - Perspectiva
- Operações Aritméticas
  - Adição
  - Subtração
  - Mistura
  - Multiplicação
  - Divisão
- Ruído em Imagens
- Práticas Python – Ao longo do Conteúdo

Prof. César C. Xavier

Processamento de Imagens e Visão Computacional

Transformações Geométricas

- É uma função cujo domínio e intervalo são conjunto de pontos (R^2 ou R^3) cuja função é injetora de forma que sua inversa existe.

- São transformações geométricas:
  - Rotação
  - Escalonamento
  - Translação
  - Perspectiva

Processamento de Imagens e Visão Computacional

Rotação

```
getRotationMatrix2D(center, angle, scale)
```

PARÂMETRO	DESCRIÇÃO
center	o centro de rotação para a imagem de entrada
angle	o ângulo de rotação em graus
scale	um fator de escala isotrópico que dimensiona a imagem para cima ou para baixo de acordo com o valor fornecido

Rotação

```
import cv2
import numpy as np
imagemOriginal = cv2.imread("folha_colorida.jpg", 0)
(totalLinhas, totalColunas) = imagemOriginal.shape
matriz = cv2.getRotationMatrix2D((totalColunas/2,
                                  totalLinhas/2), 90, 1)

ImagemRotacionada = cv2.warpAffine(imagemOriginal,
                                   matriz,
                                   (totalColunas, totalLinhas))

cv2.imshow("Resultado", imagemRotacionada)
cv2.waitKey(0)
cv2.destroyAllWindows()
```


UNIP UNIVERSIDADE PAULISTA
Processamento de Imagens e Visão Computacional

Translação

- Consiste em deslocar uma imagem de posição.
- Será utilizado a função *warpAffine*.
- Necessário uma matriz de translação, que indicará para qual posição a imagem será movida. Será usada função *float32* da biblioteca Numpy.

Translação (100 pixels)

```
import  cv2
import  numpy as np
imagemOriginal=cv2.imread("folha_colorida.jpg")
totalLinhas, totalColunas = imagemOriginal.shape[:2]
matriz  =  np.float32([[1, 0, 100], [0,  1, 100]])
imagemDeslocada =  cv2.warpAffine(
          imagemOriginal,
          matriz,
          (totalColunas,  totalLinhas))
cv2.imshow("Resultado", imagemDeslocada)
cv2.waitKey(0)
cv2.destroyAllWindows()
```


UNIP UNIVERSIDADE PAULISTA
Processamento de Imagens e Visão Computacional

Escala

```
cv2.resize(src, dsize[, dst[, fx[, fy[, interpolation]]]])
```

PARÂMETRO	*DESCRIÇÃO*
src	Matriz da imagem origem
dsize	Tamanho desejado imagem de saída
dst	Matriz da imagem de saída
fx	Fator de escala horizontal
fy	Fator de escala vertical
interpolation	Método de Interpolação (INTER_NEAREST, INTER_LINEAR, INTER_AREA, INTER_CUBIC, INTER_LANCZOS4)

Processamento de Imagens e Visão Computacional

Escala

```
import  cv2
import  numpy as np
imagemOriginal  =  cv2.imread("folha_colorida.jpg")
imagemModificada=  cv2.resize(
          imagemOriginal, None, fx =  0.5,  fy =  0.5,
          interpolation =  cv2.INTER_CUBIC)

cv2.imshow("Resultado",  imagemModificada)
cv2.waitKey(0)
cv2.destroyAllWindows()
```


Processamento de Imagens e Visão Computacional

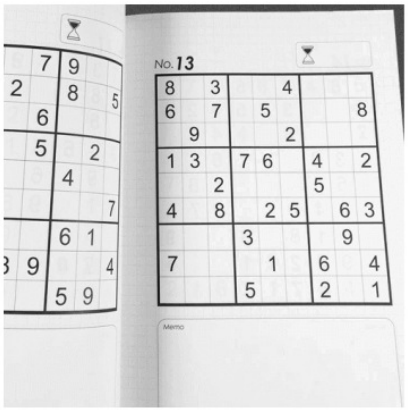
Perspectiva

- O posicionamento incorreto, a lente ou até mesmo o balanço da câmera podem interferir na perspectiva da fotografia, pode causar inclinação ou distorção do objeto representado na imagem.
- Em fotografias que possuem linhas horizontais, verticais ou formas geométricas, as distorções tornam-se ainda mais perceptíveis.

Processamento de Imagens e Visão Computacional

Perspectiva

- Para corrigir as distorções de perspectiva, podemos usar a função *warpPerspective* da biblioteca OpenCV.
- O ajuste é realizado tendo como referência uma matriz predefinida de pontos, gerada pela função *getPerspectiveTransform*.



Processamento de Imagens e Visão Computacional

Perspectiva


```
import cv2
Import numpy   as np
imagemOriginal =  cv2.imread("sudoku.jpg")
pontosIniciais =  np.float32(
            [[189,87],  [459,84],   [192,373],  [484,372]])
pontosFinais   =  np.float32(
            [[0,0],  [500,0], [0,500], [500,500]])
matriz   =  cv2.getPerspectiveTransform(
            pontosIniciais,   pontosFinais)
imagemModificada  =  cv2.warpPerspective(
            imagemOriginal,   matriz,  (500, 500))
cv2.imshow("Imagem   Original",  imagemOriginal)
cv2.imshow("Imagem   Modificada",   imagemModificada)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

UNIP UNIVERSIDADE PAULISTA

Processamento de Imagens e Visão Computacional

Operações Aritméticas

• Permite realizar operações entre imagens:

- Soma / Subtração / Mistura / Multiplicação / Divisão

- Operações entre imagens podem ser realizadas quando as imagens possuem a mesma dimensão (largura x altura) e tipo (8 bits, p.ex.).

- Atenção para possibilidade de Overflow / Underflow
  - Considerando cores de 8 bits, valores entre 0 e 255, as operações de soma e subtração não podem ultrapassar estes limites
  - Tipicamente, versões recentes do OpenCV automaticamente limita a 255.

UNIP UNIVERSIDADE PAULISTA

Processamento de Imagens e Visão Computacional

Operação de Adição

• cv2.add(): soma os valores dos pixels de uma imagem com outra, resultando em uma nova imagem.

```
import cv2

imagemFichasVermelhas= cv2.imread("fichas_vermelhas.jpg")
imagemFichasPretas = cv2.imread("fichas_pretas.jpg")
imagem = cv2.add(imagemFichasVermelhas, imagemFichasPretas)

cv2.imshow("Resultado", imagem)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

Operação de Adição

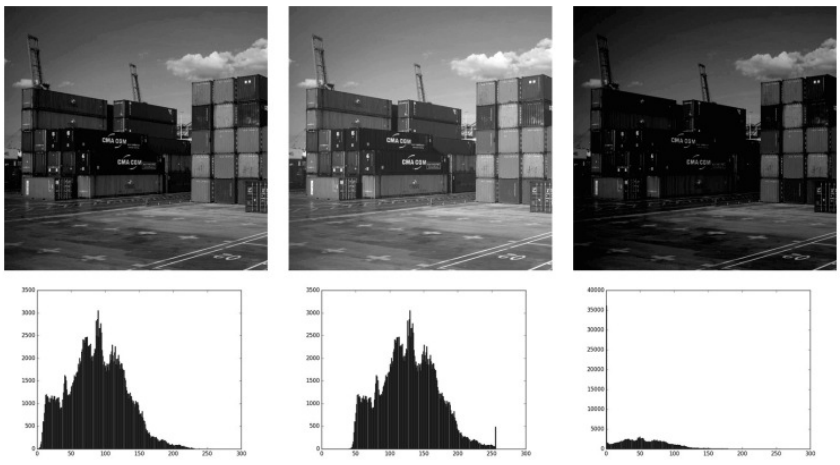
- Alteração contraste por meio da operação de soma:

```
import cv2
import numpy as np
from matplotlib import pyplot as grafico
imagemOriginal = cv2.imread("containers.jpg", 0)
imagemClara = cv2.add(imagemOriginal, 40)
imagemEscura = cv2.add(imagemOriginal, -40)
cv2.imshow("Imagem Original", imagemOriginal)
cv2.imshow("Imagem Clara", imagemClara)
cv2.imshow("Imagem Escura", imagemEscura)
grafico.hist(imagemOriginal.ravel(), 256,[0,256])
grafico.figure();
grafico.hist(imagemClara.ravel(), 256,[0,256])
grafico.figure();
grafico.hist(imagemEscura.ravel(), 256,[0,256])
grafico.show()
cv2.waitKey(0)
cv2.destroyAllWindows()
```


UNIP UNIVERSIDADE PAULISTA
Processamento de Imagens e Visão Computacional

Operação de Adição

- Alteração contraste por meio da operação de soma:



Prof. César C. Xavier

UNIVERSIDADE PAULISTA

Processamento de Imagens e Visão Computacional

Operação de Subtração

- cv2.subtract(): subtrai os valores dos pixels de uma imagem com outra, resultando em uma nova imagem.

```
import cv2

imagemFichaPosicao1 = cv2.imread("ficha-posicao-1.bmp")
imagemFichaPosicao2 = cv2.imread("ficha-posicao-2.bmp")

imagem = cv2.subtract(imagemFichaPosicao1, imagemFichaPosicao2)
cv2.imshow("Resultado", imagem)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

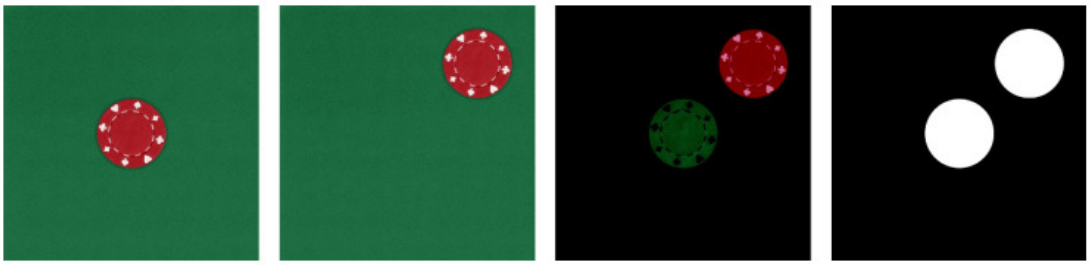

UNIVERSIDADE PAULISTA

Processamento de Imagens e Visão Computacional

Operação de Subtração

- cv2.subtract(): subtrai os valores dos pixels de uma imagem com outra, resultando em uma nova imagem.

(a): posição inicial; (b) posição final; (c) cv2.subtract(); (d) imagem binária.



(a) (b) (d) (e)

Operação de Mistura

- cv2.addWeighted(): mescla as informações, com perda de dados, de duas imagens resultando em uma nova imagem.

Parâmetro	Descrição
`src1`	Matriz referente à primeira imagem.
`alpha`	Intensidade da primeira imagem.
`src2`	Matriz referente à segunda imagem.
`beta`	Intensidade da segunda imagem.
`gamma`	Valor escalar adicionado a cada soma.

UNIP UNIVERSIDADE PAULISTA
Processamento de Imagens e Visão Computacional

Operação de Mistura

- cv2.addWeighted(): mescla as informações, com perda de dados, de duas imagens resultando em uma nova imagem.

Operação de Multiplicação e Divisão

- cv2.multiply(): faz o produto escalar entre os valores dos pixels de uma mesma posição nas duas imagens, resultando em uma nova imagem.

- cv2.divide(): pode ser usada para descriminar uma imagem da outra. Se as imagens são idênticas, com a operação da divisão de uma pela outra, os valores que representam cada pixel são iguais a 1. Dividir uma imagem pela outra consome mais recursos que a operação de subtrair.

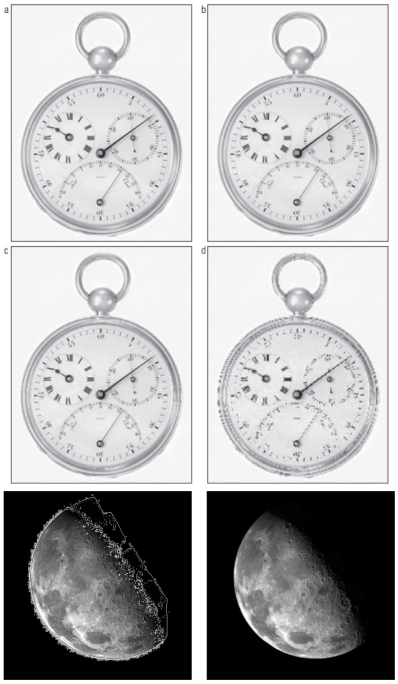
UNIP UNIVERSIDADE PAULISTA
Processamento de Imagens e Visão Computacional

Ruído em Imagens

- Ruídos reduzem a eficiência de algoritmos durante o processamento
- São variações aleatórias do sinal que difucultam a leitura do valor real.
- Tipos de Ruído:
  - Captura
  - Amostragem
  - Processamento
  - Codificação de Imagem
  - Oclusão de cena
  - Sal e Pimenta
  - Gaussiano

Ruído em Imagens

- Captura
  - Variações indesejadas provocadas por poeira no ambiente, vibração da câmera, distorção da lente, iluminação inadequada e ruído elétrico no sensor.
- Amostragem
  - Quando a taxa de amostragem e da quantização de intensidade de cor não são suficientes para proporcionar uma representação verdadeira da imagem analógica.
- Processamento
  - Quando há limitações na precisão numérica (*oveflow* de inteiros ou em representações em ponto flutuante com aproximações matemáticas).



UNIP UNIVERSIDADE PAULISTA
Processamento de Imagens e Visão Computacional

Ruído em Imagens

- Codificação de Imagem
  - Ocorre durante técnicas de compressão de imagens com perda (ex. jpeg).

- Oclusão de cena
  - Quando o objeto de interesse é obscurecido por outro.

Processamento de Imagens e Visão Computacional

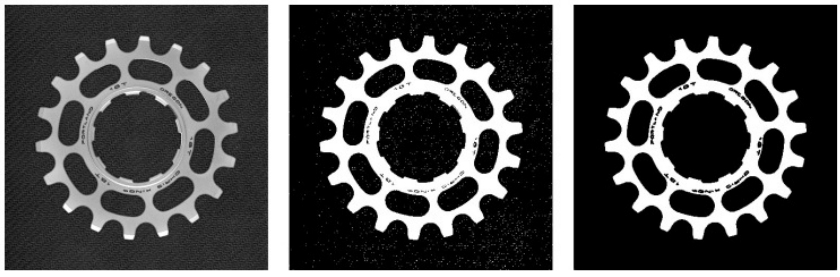
Ruído em Imagens

- Sal e Pimenta
  - Caracterizado pela adição aleatória de pixels pretos e brancos, intensos ou fracos, na imagem
- Gaussiano
  - Conhecido como ruído aditivo, ocorre quando a variação aleatória do sinal segue a distribuição gaussiana.

Processamento de Imagens e Visão Computacional

Ruído em Imagens

- Ruído em imagens binárias



(a) (b) (c)

(a): arquivo original

(b) conversão para imagem binária

(c) imagem processada