FGV - Fundação Getulio Vargas

Visão Computacional - Lista 1

Professor: Moacyr Alvim Horta Barbosa da Silva
Monitor: Tulio Koneçny

A lista deverá ser entregue tanto no formato .pdf e em .py ou .ipynb
Entrega: 22/03/23

1 Rotação de uma Imagem

Uma opção seria utilizar a função de rotação do OpenCV. A ideia do exercício é não utilizar esta função, mas sim implementá-la manualmente (para compreender o que está ocorrendo nos bastidores). Para realizar este exercício, você deve preencher as lacunas da função escrita no código em anexo. Esta função tem como entrada:

(i) uma imagem (img),

(ii) o ângulo de rotação (angulo),

(iii) o centro de rotação (centro),

e devolve a imagem rotacionada.

```
def my_rotation(img, angulo, centro):

    # nessa primeira parte, vamos definir a transformação que leva a posicao dos
    # pixels da imagem original para a posicao dos pixels do imagem rotacionada.

    # a primeira matriz de translação muda a origem das coordenadas do canto da
    #imagem para o centro da imagem
    matriz_translacao = # Complete aqui
    # a matriz de rotacao aplica a rotacao em torno da origem
    matriz_rotacao = # Complete aqui
    # a composicao coloca todas as matrizes em uma só: aplica a translacao
```

```
    #(muda a origem), rotaciona, volta para a origem anterior
    matriz_composicao = # Complete aqui

    # criar imagem rotacionada em preto, com mesmas dimensões da original
    height, width = img.shape[:2]
    rotated_image = np.zeros((height,width,3), np.uint8)
    # o próximo passo é percorrer cada pixel da nova imagem e verificar qual é o
    #pixel correspondente na imagem original
    m_comp_inv = matriz_composicao.I
    for linr in range(height):
        for colr in range(width):
            pos_rot = np.matrix([linr, colr, 1]).T
            pos_orig = # Complete aqui
            lin = round(pos_orig[0,0]); col = round(pos_orig[1,0]);
            if (lin >=0 and lin < height) and (col >= 0 and col < width):
                #opa, é um pixel pertencente à imagem original...
                rotated_image[linr, colr] = # Complete aqui

    return rotated_image
```

Você pode comparar seu resultado com o função do Open CV.

```

# Obter o centro da imagem
height, width = img.shape[:2]
center = (width/2, height/2)

# Definir a matriz de rotação
M = cv.getRotationMatrix2D(center, 30, 1)

# Aplicar a rotação na imagem
rotated_img = cv.warpAffine(img, M, (width, height))

#cv.imshow('Display window', rotated_img)
#cv.waitKey(0); cv.destroyAllWindows()
```

2 Transformação Projetiva de uma Imagem

Escreva uma função que tenha uma como entrada:

(i) uma imagem (img),

(ii) uma matriz de uma transformação projetiva (T),

e devolva a imagem transformada.

```
def my_transform(img, T):
# Escreva seu código aqui
return transform_image
```

A ideia é semelhante ao exercício 1, mas ao invés da rotação, temos uma transformação projetiva (homografia).

Teste a sua função, faça a transformação projetiva com apenas um ponto de fuga no eixo x (digamos, o ponto $(2000, 0)$ em coordenadas x, y), da imagem do Palazzo Farnese Fassade (em anexo).

Você pode usar outra imagem de sua preferência, caso queira. Não se esqueça de trabalhar com coordenadas homogênas.

3 Estimação da Transformação Projetiva

3.1 Crie uma função

Escreva uma função que tenha uma como entrada:

(i) quatro pontos "fonte" no plano projetivo, possivelmente impróprios,

(ii) quatro pontos "destino" no plano projetivo, possivelmente impróprios.

e devolva a transformação projetiva $T : \mathbb{RP}^2 \to \mathbb{RP}^2$ que leva os pontos "fonte" nos pontos "destino".

```
def my_estimation(lst1, lst2):
# Escreva seu código aqui
return estimation_T
```

3.2 Teste a função

Encontre a transformação projetiva que leva o campo de futebol da imagem do gol no jogo Vasco contra Flamengo (em anexo) em um retângulo correspondente ao campo de futebol visto de cima em projeção ortogonal (em anexo, mas fique a vontade para usar outra). Use esta projeção para desenhar o campo de futebol visto de cima junto com a posição do Léo Pelé, do goleiro e de um jogador do Flamengo.

4 Extra

4.1 You are Fake News ... or maybe not.

Muitas vezes, os torcedores apaixonados pelo time desconfiam do resultado do VAR e das decisões de impedimento, como foi o caso do vídeo `https://www.instagram.com/papodeboleiros/reel/CvAhQazvuhu/`. Teste sua função na imagem da partida entre Union e Boca Juniors (em anexo) e verifique se o método mostrado no vídeo produz o mesmo resultado de impedimento que a função implementada (basta fazer a transformação na imagem com as "retas paralelas" do vídeo e verificar se de fato são paralelas). Comente sobre o resultado da comparação. Você acredita que este método seja sempre correto? Caso contrário, forneça um contraexemplo (se possível, com uma imagem).

4.2 My VAR-lidation

Utilizando suas funções *my_estimation* e *my_transform*, faça uma função que tem como entrada:

(i) duas imagens, uma do jogo e uma do campo (img1,img2),

(ii) uma lista de pontos "fonte" e "destino"(lst1,lst2)

(iii) os pixels relativos aos dois jogadores que estão sendo analisados (p1,p2)

e devolve a imagem do campo com uma linha marcada de ambos os jogadores, semelhante com as análises do VAR.

Perceba que basta desenhar uma linha (pintando todos os pixels com a mesma coordenada em x da mesma cor) quando a imagem do jogo estiver em projeção ortogonal e depois usar a inversa da matriz de projeção para voltar ao formato original.

Observação: a imagem provavelmente sofrerá perda de qualidade neste processo. Se desejar manter a qualidade, você poderá realizar esse processo apenas em uma imagem com a linha inserida e adicioná-la depois de transformada na imagem original.