Visão Computacional: reconhecimento de texto com OCR e OpenCV

cursos.alura.com.br/course/visao-computacional-reconhecimento-texto-ocr-opencv/task/112869

Já importamos a fonte *Calibri* para o nosso projeto e podemos começar a construir uma nova função chamada `escreve_texto`.

```
def escreve_texto(texto, x, y, img, fonte, tamanho_texto=32):
  fonte = ImageFont.truetype(fonte, tamanho_texto)
  img_pil = Image.fromarray(img)
  draw = ImageDraw.Draw(img_pil)
  draw.text((x, y - tamanho_texto), texto, font = fonte)
  img = np.array(img_pil)
  return img
```

Vamos analisá-la passo a passo! Começaremos definindo `def escreve_texto()`. Primeiro, passaremos o texto que estamos recebendo, `texto`, e que vem da função `for`:

```
img_copia = rgb.copy()
for i in range(len(resultado['text'])):
  confianca = int(resultado['conf'][i])
  if confianca > min_conf:
    x, y, img = caixa_texto(resultado, img_copia)

    texto = resultado['text'][i]
    cv2.putText(img_copia, texto, (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0,100,255))

cv2_imshow(img_copia)
```

Depois, passaremos: `x` e `y`, que se referem às posições; a imagem, `img`; a `fonte`, que será a Calibri; e o tamanho do texto que, por enquanto, será 32, `tamanho_texto=32`, mas, a depender da imagem ou de sua disposição, podemos diminuir ou aumentar o tamanho.

Na próxima linha, definiremos a `fonte`, que será igual a `ImageFont`, e que vem do módulo `PIL`:

```
from PIL import ImageFont, ImageDraw, Image

fonte = '/content/text-recognize/Imagens/calibri.ttf'
```

Seguindo, escreveremos `truetype()`, passaremos a `fonte` e o `tamanho_texto`. Na próxima linha, faremos `img_pil`, isto é, a imagem PIL, igual a `Image.fromarray()`, passando a imagem, `img`. Com isso, transformaremos uma imagem em *array* para PIL.

```
def escreve_texto(texto, x, y, img, fonte, tamanho_texto=32):
  fonte = ImageFont.truetype(fonte, tamanho_texto)
  img_pil = Image.fromarray(img)
```

Em seguida, faremos o desenho da nossa escrita. Lembrando que, anteriormente, começamos pelo desenho da caixa, *bounding box*, e depois fizemos o texto. Desta vez, começamos com a `img_PIL` e depois passamos para o desenho das fontes, `draw = ImageDraw.Draw(img_pil)`. Então, estamos fazendo o desenho, `ImageDraw.Draw()`. Essa etapa nos permitirá desenhar na `img_pil`, que é a imagem transformada em PIL.

Na próxima linha, faremos um `draw.text()` no `x,y`, menos o tamanho do texto, `tamanho_texto`, que será 32. Depois, `texto` e `font = fonte`. Esses são os parâmetros que passamos para o `draw.text`, eles farão o desenho do nosso texto. Por fim, passamos outra vez `img = np.array(img_pil)`.

```
def escreve_texto(texto, x, y, img, fonte, tamanho_texto=32):
  fonte = ImageFont.truetype(fonte, tamanho_texto)
  img_pil = Image.fromarray(img)
  draw = ImageDraw.Draw(img_pil)
```

Basicamente, passamos a imagem como um *array*, ele transformou em PIL e escreveu o texto com a fonte Calibri. Depois disso, pegamos a imagem, em PIL, e transformamos para *array* de novo, assim ela poderá ser utilizada no OpenCV. O processo é um pouco demorado e custoso, mas fontes externas, como a Calibri, nos permitem escrever termos de caracteres especiais, o que não é possível com as fontes que são originalmente do OpenCV.

Agora, vamos rodar a função!

```
def escreve_texto(texto, x, y, img, fonte, tamanho_texto=32):
  fonte = ImageFont.truetype(fonte, tamanho_texto)
  img_pil = Image.fromarray(img)
  draw = ImageDraw.Draw(img_pil)
  draw.text((x, y - tamanho_texto), texto, font = fonte)
  img = np.array(img_pil)
  return img
```

A próxima etapa é copiar a função `caixa_texto` que fizemos anteriormente e colar na célula abaixo. Além disso, vamos comentar o `#cv2.putText()` e adicionar uma nova linha para escrevermos o texto:

```
img_copia = escreve_texto(texto, x, y, img_copia, fonte)
```

Então, depois do texto, queremos que ele faça um `img_copia` igual a `escreve_texto()`, função que acabamos de criar, passando `texto`, `x` e `y`, a imagem de cópia, `img_copia`, e a `fonte`. Vamos rodar e conferir o que ele vai retornar.

```
img_copia = rgb.copy()
for i in range(len(resultado['text'])):
  confianca = int(resultado['conf'][i])
  if confianca > min_conf:
    x, y, img = caixa_texto(resultado, img_copia)

    texto = resultado['text'][i]
    #cv2.putText(img_copia, texto, (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0,100,255))
    img_copia = escreve_texto(texto, x, y, img_copia, fonte)

cv2_imshow(img_copia)
```

Ele retornou uma imagem que já conhecemos, com o logo do Google, "G" maiúsculo e, abaixo, a frase "Tesseract OCR".

A fonte Calibri que escolhemos para a escrita do texto é um pouco mais arredondada que a utilizada anteriormente, do OpenCV, e até mais simples de ser lida, por já estarmos habituados. Além disso, as letras têm cor branca e isso ajuda na visibilidade do texto. Nós não temos nenhum caractere especial, mas, nestes casos, é bastante recomendável a utilização de fontes externas.

Concluímos duas tarefas importantes: colocar *bounding box* e adicionar fontes externas. Nos encontramos na próxima aula para outros desafios!