# CI1026 - Visão computacional e percepção - TA1

Carlos Eduardo Cichon Henriques  Camile Nunes dos Anjos

Abril 2023

## 1 Informações Técnicas

Para a realização do trabalho e a codificação do Jupyter Lab, nós utilizamos as seguintes bibliotecas:

- OpenCV
- Numpy
- Matplotlib
- ipywidgets

Link para o projeto no Github: aqui, nele apresentamos um workflow de processamento de imagens e comparações lado a lado com a imagem original.

## 2 Imagem Original

# 3 Imagem em escala de cinza

# 4 Filtros

Nessa sessão aplicamos filtros à imagem em escala de cinza.

## 4.1 Filtro Linear

Fizemos um filtro linear aplicando um kernel 7x7 repleto de 1 e dividindo por um valor x. Ficando no formato:

$$\frac{1}{x} \times \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Dentro do .ipynb o valor de x pode ser escolhido numa barra deslizante. Abaixo temos um exemplo utilizando x = 44.

## 4.2 Sharpening

Utilizamos um kernel no formato:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

para realizar o sharpening na imagem, acentuando contornos.

## 4.3 Gaussiano

Utilizamos um kernel gaussiano como filtro na imagem, obtendo o seguinte resultado:

# 5 Threshold

Nessa sessão utilizamos a técnica de thresholding no processamento da imagem.

## 5.1 Valores variados

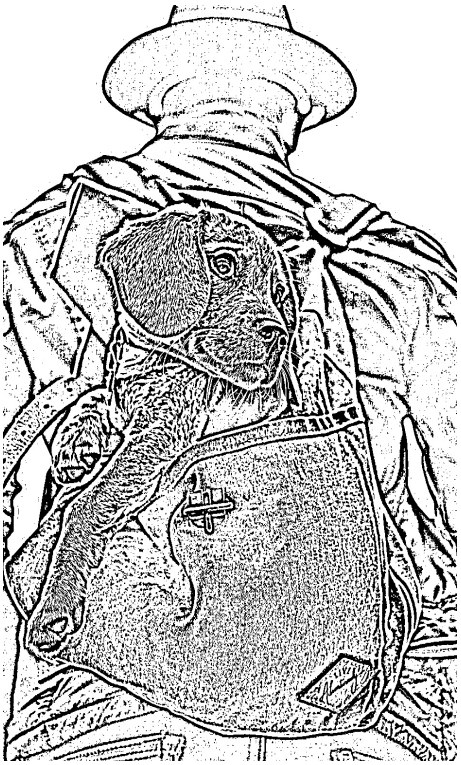
Dentro do arquivo .ipynb implementamos uma barra deslizante que muda o valor do threshold e obtivemos resultados como estes:

### 5.1.1 Threshold 127

### 5.1.2 Threshold 168

## 5.2 Threshold adaptativo

Também utilizamos o threshold adaptativo do OpenCV, obtendo o seguinte resultado:



# 6 Detecção de bordas

Nessa sessão implementamos diferentes técnicas para a detecção de bordas na imagem escolhida.

## 6.1 Sobel

Utilizamos o kernel x e y nos formatos:

$$x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

$$y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

e geramos esta imagem com as bordas detectadas:

## 6.2 Canny

Aplicamos a técnica de Canny para detectar as bordas com threshold1 = 100 e threshold2 = 200 e obtivemos o seguinte resultado: