

UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO  
CENTRO UNIVERSITÁRIO NORTE DO ESPÍRITO SANTO

## **ESTRUTURA DE DADOS 1**

Nome: Gabriel Martins Spósito

Prof.: Oberlan Christo Romao

Relatório de exercício de programação

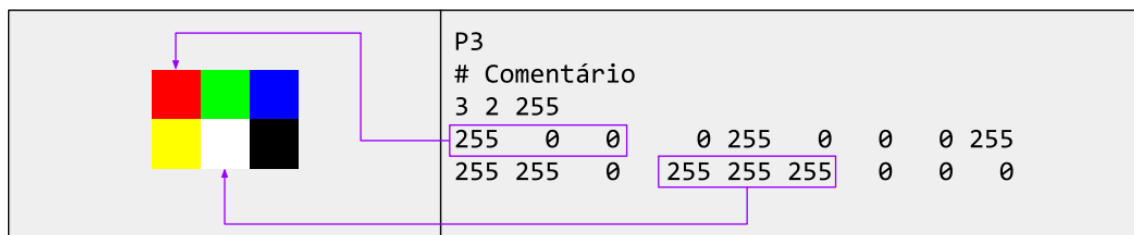
FOTOXOP

# 1.OBJETIVO

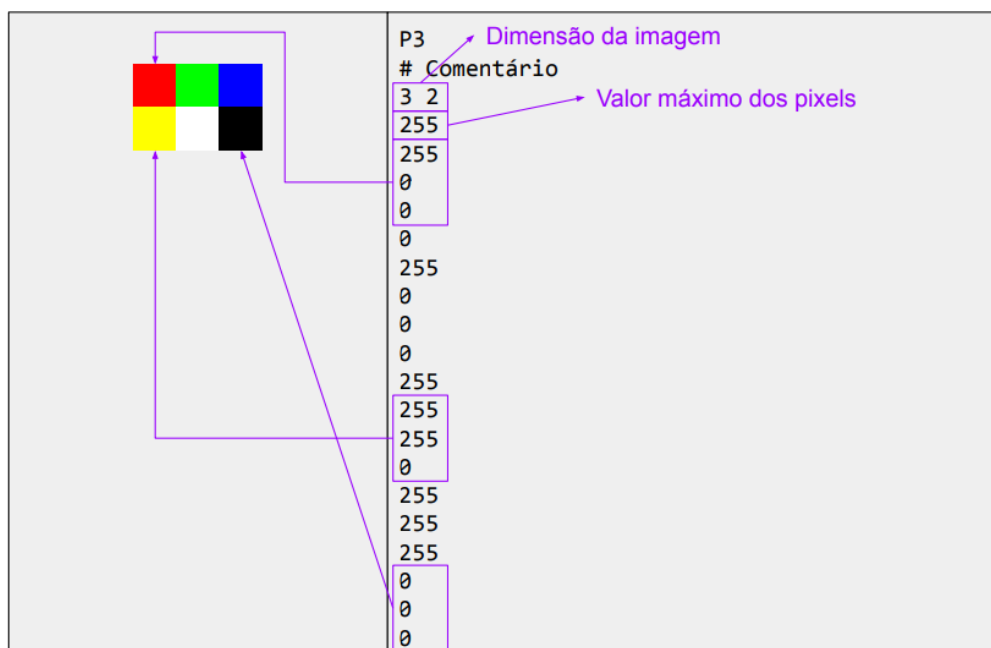
Este relatório faz parte de um exercício programa (EP) de Estrutura de Dados 1 de Eng. Da Computação com o intuito de ensinar Makefile, TAD, funções, matriz e processamento de imagens. Mostrando a aplicação de todos esses elementos em um trabalho pontuado como incentivo para melhorar o conhecimento dos alunos. Fazendo filtros para imagens do tipo PPM(Portable pixmap) que e formato bitmap, pixel por pixel, sem nenhuma compressão, pois isso gera uma matriz bidimensional com valores inteiros de 0 a 255 para fazer as mudanças necessárias para o exercício, pois cada pixel e representado por 3 valores nesse intervalo.

## 2.FUNDAMENTAÇÃO TEÓRICO BÁSICA

Esses 3 valores representam respectivamente a intensidade de Vermelho, Verde e Azul do pixel (padrão RGB). Veja o exemplo abaixo: 255 0 0 indica R=255, G=0, B=0, ou seja, vermelho puro; 0 0 0 indica preto, 255 255 255 branco, 255 255 0 indica amarelo, etc. Na figura, a parte da esquerda é o resultado visto ao abrir o arquivo em um visualizador de imagens. A esquerda é mostrado o conteúdo ao abrirmos a imagem em um editor de texto (VSCode, por exemplo).



O arquivo PPM é escrito em formato texto, podendo abrir em qualquer editor de texto simples, com VS Code e até mesmo modificá-la. Quando aberto em um editor de imagens, como o GIMP, ele é interpretado como uma figura, e os valores indicam a cor de cada pixel, como mostrado no exemplo abaixo.



## 3.METODOLOGIA

O professor já disponibilizando parte do código e deixando só para fazer poucas partes do mesmo ficando disponível no AVA (Ambiente Virtual de Aprendizagem) da UFES, usando como complemento o OpenGL + GLUT para termos um programa mais iterativo.

Foi necessário instalar a biblioteca freeglut3-dev no Linux e testa o Valgrind movendo arquivos específicos para a pasta do Valgrind e rodando o make memcheck dentro da pasta. Sendo necessário muda o ./EP1 <NOMEDAFOTO>.ppm para poder fazer os filtros em outras imagens a escolha do aluno.

Foram passadas dicas e a ordem de implementação das partes que faltavam nos arquivos, com a ordem de seguir essa implementação: `alocalmagem (Imagem.c)`; `liberalmagem (Imagem.c)`; `salvalmagem (Imagem.c)`; `escurecerImagem`, `clarearImagem`, `escalaDeCinzalImagem (Filtros.c)`; `copialmagem (Imagem.c)`; `deteccaoBordasLaplace`, `filtroSobel (Filtros.c)`; `meuFiltro (Filtros.c)`.

## Os filtros



Imagem original

- **Escurecer imagem**

O escurecimento da imagem e uma subtração de cada banda de cor pelo valor que é pedido no terminal, não passando de 0 o menor valor.



Imagem original	Imagem escurecida por fator 50
<pre>P3 # Comentário 3 2 255 255 0 0 0 255 0 0 0 255 0 255 255 255 0 255 255 255 0</pre> 	<pre>P3 # Comentário 3 2 255 205 0 0 0 205 0 0 0 205 0 205 205 205 0 205 205 205 0</pre> 



Imagem escurecida com fator 100

- **Clarear imagem**

O clareamento da imagem é uma soma de cada banda de cor pelo valor que é pedido no terminal, não passando de 255 o maior valor.

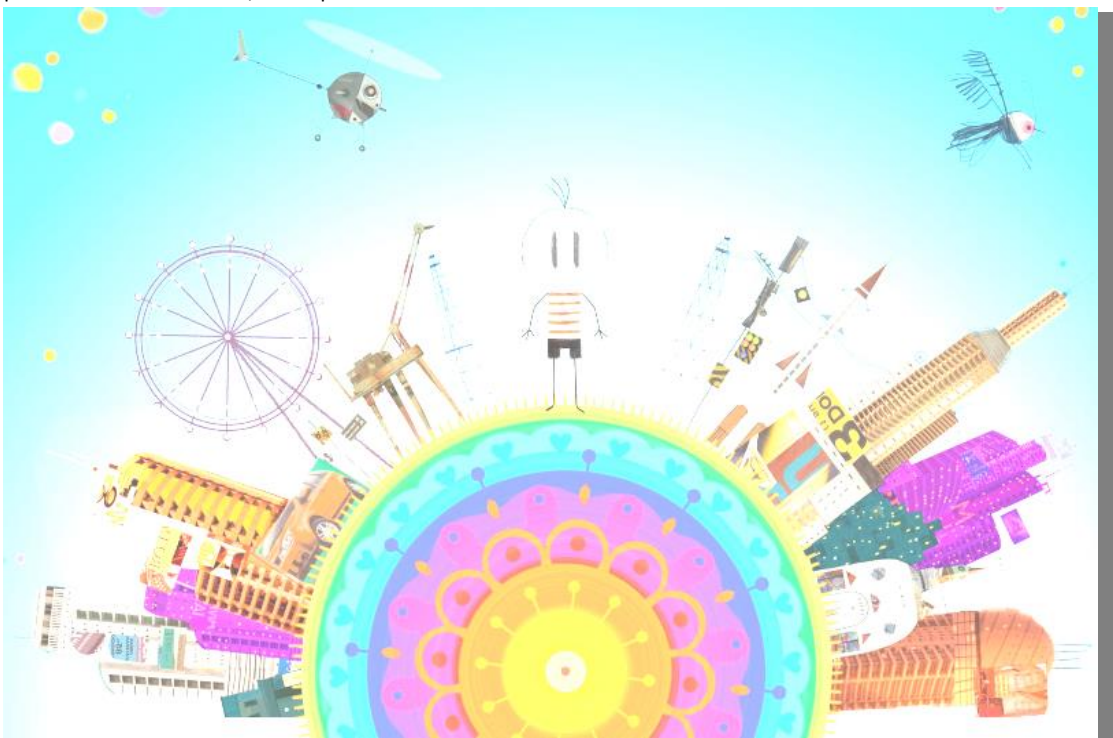




Imagem clareada com fator 100

- **Imagem em escala de cinza**

Para transforma uma imagem em escala de cinza basta pegar a média dos 3 valores do pixel e fazer cada banda de cor receber essa média porem e essa escala não pode deixar a desejar principalmente para as cores primarias que são o vermelho, verde e azul. Outra alternativa e fazer uma equação matemática para corrigir essa diferença que tem  $[0.3 \times \text{RED} + 0.59 \times \text{GREEN} + 0.11 \times \text{BLUE}]$ .

Imagem original	Escala de Cinza	
<div> P3 # Comentário 3 2 255 255 0 0 0 255 0 0 0 255 255 255 0 255 255 255 0 0 0  </div>	<div> P3 # Comentário 3 2 255 85 85 85 85 85 85 85 85 85 170 170 170 255 255 255 0 0 0  </div>	
	Com media	Com a equação



- Filtro de Sobel

O filtro de Sobel pode ser usado para detecção de bordas em imagens digitais e consiste na aplicação de duas matrizes (matriz de convolução), que detectam os contornos na vertical e na horizontal. No PDF do professor mandou utilizar essas duas matrizes:

$$G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \text{ e } G_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

Essas matrizes são aplicadas a cada banda de cor de cada pixel da imagem. Os coeficientes na matriz indicam o peso de cada pixel vizinho. Para cada pixel, são utilizados seus 8 pixels vizinhos, verificando se essas equações não ficam abaixo de 0 ou acima de 255, caso isso ocorra tem que fixar os valores que são negativos em 0 e os que estão acima de 255 em 255. Os demais devem ser calculados da mesma forma.

...	...	...	...	...	...	...	...	...	...	...	...
...	10	15	80	85	95	...					...
...	15	40	120	130	131	...					...
...	16	41	100	120	120	...					...
...	17	43	80	110	130	...					...
...	19	50	70	42	20	...					...
...	...	...	...	...	...	...					...

...	...	...	...	...	...	...	...	...	...	...	...
...											...
...											...
...											...
...											...
...											...
...											...

Para esse filtro precisou usar uma cópia da imagem original pois se usasse na imagem original afetaria os resultados pois como pega os valores vizinhos isso mudaria o resultado final fazendo um erro em cadeia mudando toda a imagem. Fazendo tudo certo a imagem resultante dará ênfase as bordas dos objetos da imagem (onde há mudanças brusca de cor).

Deve aplicar as duas matrizes e combinar os resultados podendo ser de forma simples, como somar, tirar a media, manter o maior ou menor valor entre outros. Porem sites de processamentos de imagens era usando outra fórmula de magnitude:

$$c = \sqrt{a^2 + b^2}$$

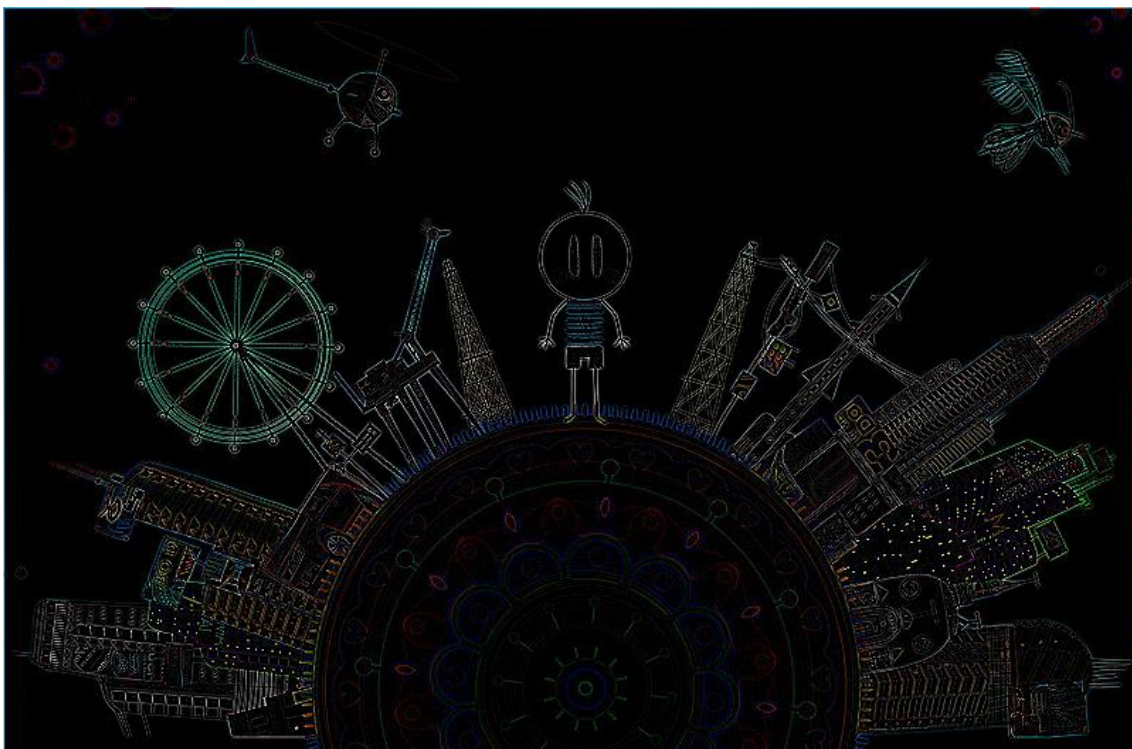




- **Detecção de bordas de Laplace**

A implementação do filtro de detecção de bordas de Laplace segue a mesma ideia do filtro de Sobel, mas agora existe apenas uma matriz de convolução (apresentada abaixo) que deve ser aplicada em cada banda de cor de cada pixel.

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

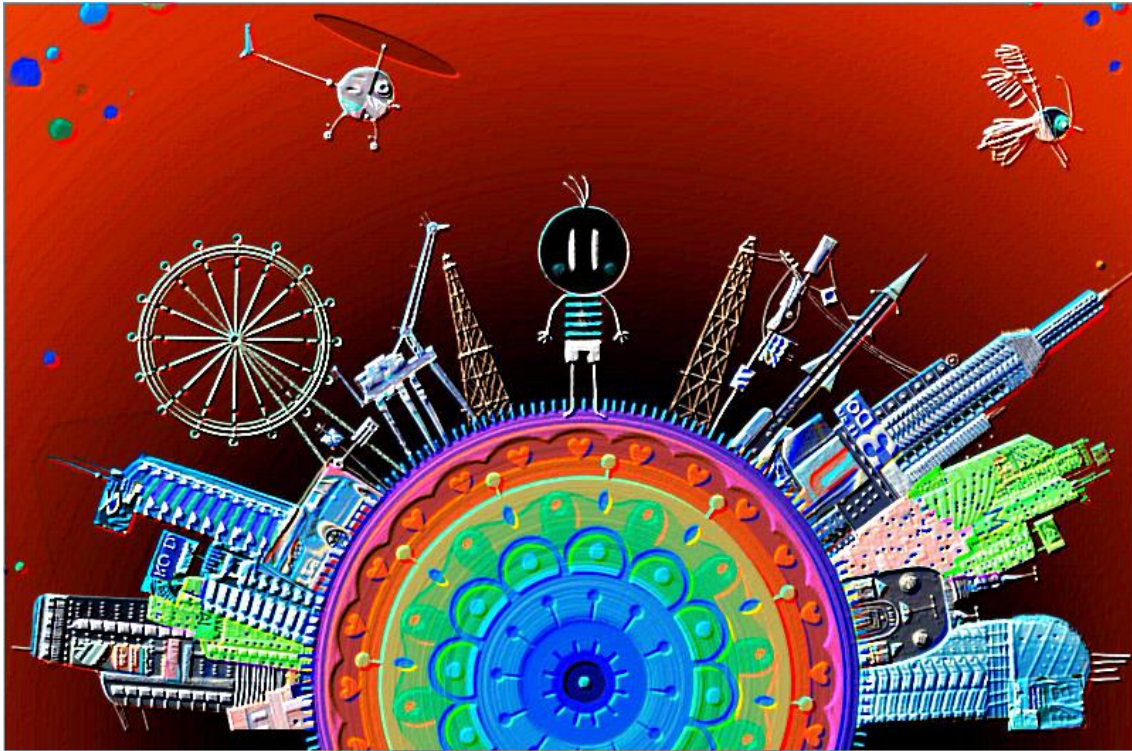


- **Meu filtro**

Usei um filtro de destaca alto-relevo semelhante ao Sobel dá uma ilusão de profundidade ao enfatizar as diferenças de pixels em uma determinada direção. Nesse caso em uma direção ao longo de uma linha da parte superior esquerda para a parte inferior direita, usando essa matriz de convolução para achar o relevo.

	-2	-1	0	
	-1	1	1	
	0	1	2	

Depois desse processamento de imagem fiz outra operação para achar o negativo da imagem resultante  $a = 255 - a$  para cada banda de cor sempre verificando para que não ficasse negativo, se ocorresse seria fixado em 0. Resultando em uma imagem que contenha alto-relevo e negativo.





## 4.RESULTADOS

### IMAGENS DE TESTE 1



Imagem original



Imagem clareada fator 100

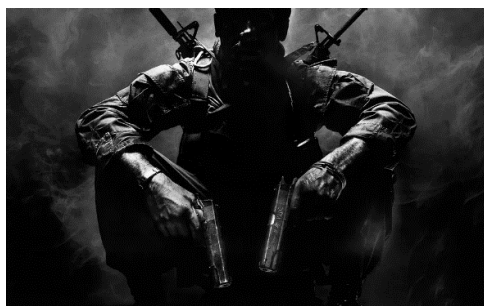


Imagem escala de cinza



Imagem escurecida fator 100



Imagem bordas Laplace



Imagem meu filtro



Imagem filtro Sobel

## IMAGENS DE TESTE 2



Imagem original



Imagem escurecida fator 100



Imagem clareada fator 100



Imagem escala de cinza



Imagem bordas Laplace



Imagem meu filtro



Imagem filtro Sobel



## IMAGENS DE TESTE 3



Imagem original

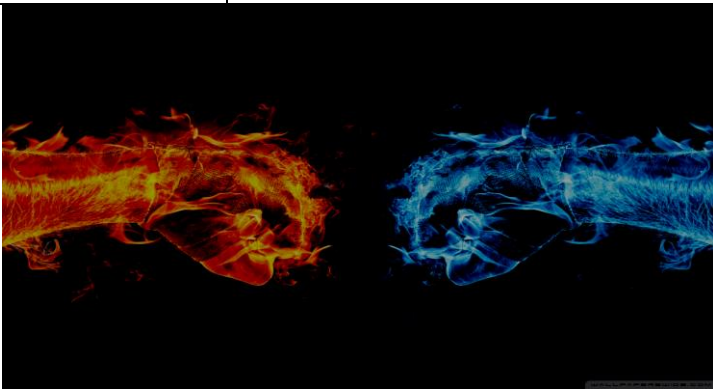


Imagem escurecida fator 100



Imagem clareada fator 100



Imagem escala de cinza



Imagem bordas Laplace

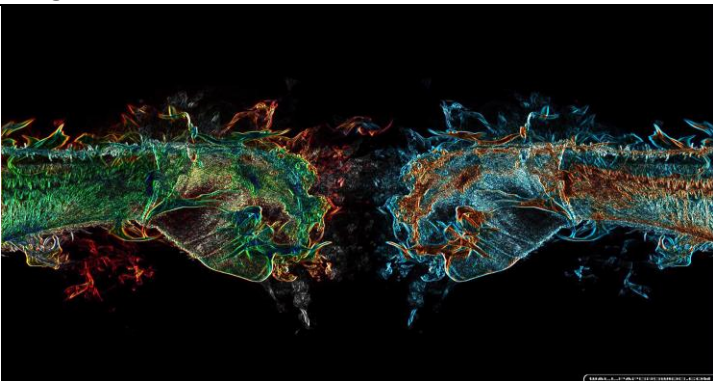


Imagem filtro Sobel

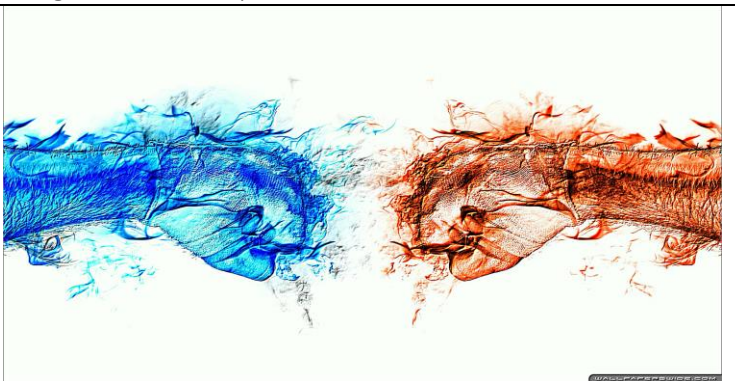


Imagem meu filtro

## IMAGENS DE TESTE 3 Valgrind

```
==2744== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==2744== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==2744== Command: ./EP1 fire.ppm
==2744==
Digite o fator de escurecimento: 100
salvaImagem: A imagem foi salva no arquivo: 'imagemEscurecida.ppm'
Digite o fator de clareamento: 100
salvaImagem: A imagem foi salva no arquivo: 'imagemClareada.ppm'
salvaImagem: A imagem foi salva no arquivo: 'imagemEscalaDeCinza.ppm'
salvaImagem: A imagem foi salva no arquivo: 'imagemFiltroSobel.ppm'
salvaImagem: A imagem foi salva no arquivo: 'imagemBordasLaplace.ppm'
salvaImagem: A imagem foi salva no arquivo: 'imagemMeuFiltro.ppm'
salvaImagem: A imagem foi salva no arquivo: 'imagemOriginal.ppm'
==2744==
==2744== HEAP SUMMARY:
==2744==      in use at exit: 0 bytes in 0 blocks
==2744==    total heap usage: 11,920 allocs, 11,920 frees, 68,562,608 bytes alloc
ated
==2744==
==2744== All heap blocks were freed -- no leaks are possible
==2744==
==2744== For lists of detected and suppressed errors, rerun with: -s
==2744== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
gabriel@gabriel-VirtualBox:~/Área de Trabalho/EP1/en/Valgrind$
```

## 5.CONCLUSÃO

Esse exercício programa proporcionou dores de cabeças, pois no Windows as programações dos programas para fazê-lo são simples porem qualquer coisinha não faz rodar ou dos erros inesperados e, portanto, foi necessário fazer um VirtualBox com Linux Mint para poder usar o VS Code e compilar e programa para ficar mais fácil, contudo tirando esses problemas o EP mostrou de um jeito pratico como utilizar TAD e alocação dinâmica em uma coisa que faz parte do dia a dia de várias pessoas que são filtros de imagens.



## 6. referências bibliográficas

<https://setosa.io/ev/image-kernels/>

<https://wenyaraujo.github.io/Projetos/negativo/negativo.html>

[https://pt.wikipedia.org/wiki/Filtro\\_Sobel](https://pt.wikipedia.org/wiki/Filtro_Sobel)

<https://capiwararex.wordpress.com/2016/04/25/dip03-matriz-de-convolucao-e-deteccao-de-bordas/>

[https://docs.gimp.org/2.8/pt\\_BR/plugin-in-convmatrix.html](https://docs.gimp.org/2.8/pt_BR/plugin-in-convmatrix.html)