Universidade Estadual de Campinas

Instituto de computação

Introdução ao processamento de imagem digital

MC920

Trabalho #1

Nome: Leonardo de Alencar Lopes **RA:** 171928

1. Introdução

O objetivo deste trabalho é aplicar alguns filtros no domínio espacial, através da operação de convolução de um kernel sobre a imagem. Para isso, foi utilizada a linguagem Python 3 e a biblioteca de tratamento de imagens OpenCV.

Junto a este documento está o arquivo comprimido **171928-Trabalho-1.zip**. Nele há todos os scripts que foram implementados e serão citados ao longo deste texto, além das pastas **img**, que contém as imagens utilizadas pelos algorítmos, e **resultados**, que contém as imagens de *outputs*.

2. O programa

Para esse projeto, foi implementado o algorítmo *filtro.py*, que recebe como parâmetros uma imagem png monocromática (que deve estar presente na pasta img) e um valor inteiro **h**, $1 \leq h \leq 11$, referente ao filtro que deseja aplicar. Os filtros h1, h2, …, h11, foram definidos conforme o enúnciado proposto para este relatório e podem ser acessados através do arquivo *filtros_lib.py*. Por exemplo: *python3 filtro.py city.png 4.* Seu output será gravado na pasta de resultados, além de ser mostrado em uma nova aba, junto com a imagem original para comparação.

Visando automatização da apresentação dos filtros, também foi implementado um script para executar o programa com todos os filtros, um por vez, com imagens selecionadas. Para executá-lo, basta navegar através do terminal para a pasta Trabalho-0 e inserir o comando *sh todos_filtros.sh*. As imagens escolhidas para serem filtradas por esse script foram escolhidas como exemplos para discussões dos filtros neste documento.

Para realizar a convolução, foi utilizada a função *filter2D*. São passados como parâmetros a imagem passada como input para o programa, uma parâmetro de profundidade da imagem (constante de -1, para que a imagem de saída tenha a mesma profundidade da de entrada) e o filtro a ser aplicado.

Bordas

Como padrão, a função *filter2D* trata casos de borda fazendo um reflexo da imagem nas proximidades (g,f,e,d,c,b | a,b,c,d,e,f,g | f,e,d,c,b,a). Além dessa abordagem, a bibloteca OpenCV

provê outras formas de tratar casos de borda, bastando modificar o último argumento a ser passado para a função. Assim, visando analisar as diferenças, foram desenvolvidos outros scripts que aplicam o filtro desejado, diferindo apenas na forma como operam nas bordas. Esses scripts são: *filtro_borda_replicada.py* (a,a,a,a,a,a | a,b,c,d,e,f,g | f,f,f,f,f,f) e *filtro_borda_constante.py* (i,i,i,i,i,i | a,b,c,d,e,f,g | i,i,i,i,i,i).

Figure 1: Borda refletida (padrão)

Figure 2: Borda replicada

Figure 3: Borda constante

Analisando as imagens acima, obtidas aplicando o filtro h2 no arquivo *city.png* com diferentes abordagens para caso de borda, não foram observadas diferenças relevantes entre elas. Provavelmente isso ocorre por conta do número de pixels da imagem ser muito maior do que o número de elementos do filtro, não tendo uma boa margem para notar suas diferenças. Portanto, para o programa *filtro.py* foi definida a abordagem padrão para o caso de bordas (refletida) e a análise dos filtros será realizada com ele.

3. Filtros

Filtro h1

0	0	-1	0	0
0	-1	-2	-1	0
-1	-2	16	-2	-1
0	-1	-2	-1	0
0	0	-1	0	0

Table 1: Filtro h1

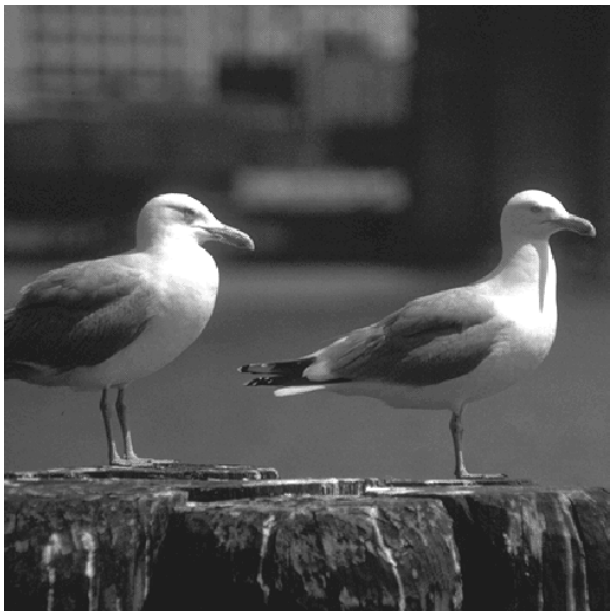

Figure 4: Original seagull.png

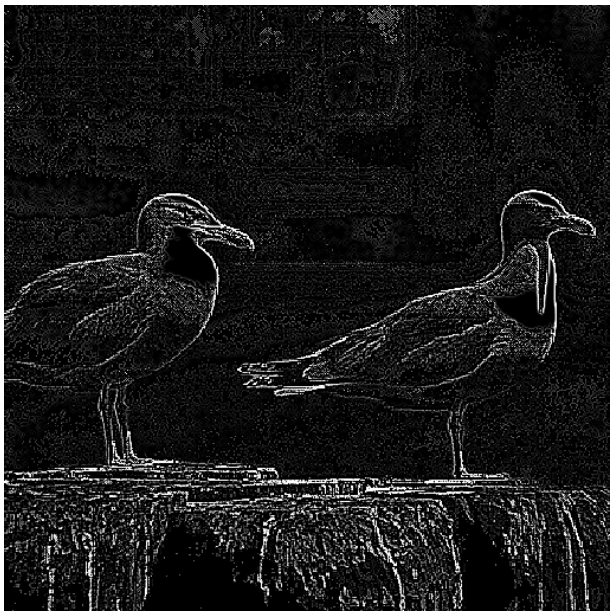

Figure 5: Filtrada com h1 seagull.png

No filtro h1, é nítido o realce sobre bordas que é aplicado sobre a imagem original. Os pássaros são precisamente contornados e destacados, elementos estes com as bordas mais bem definidas da imagem. Partes que não tem uma borda tão bem definidas são mantidos escuros, como pode ser observado na região de cima. Por outros lado, o objeto que os pássaros estão apoiados tem um número grande de pequenas bordas, que são realçadas.

Filtro h2

1/256	4/256	6/256	4/256	1/256
4/256	16/256	24/256	16/256	4/256
6/256	24/256	36/256	24/256	6/256
4/256	16/256	24/256	16/256	4/256
1/256	4/256	6/256	4/256	1/256

Table 2: Filtro h2

Figure 6: Original city.png

Figure 7: Filtrada com h2 city.png

No filtro h2, é possível identificar uma perda de nitidez ao longo de toda a imagem, muito parecido com leves borrões. Essa característica é bem notável na segunda casa, da esquerda para direita. No telhado dela as telhas são mais heterogêneas, sendo possível identificar bordas entre cada telha individual. Na imagem filtrada, as bordas da telhas desaparecem. Efeito parecido pode ser notado nas janelas e blocos da mesma casa. Esse filtro se assemelha muito com o efeito de desfoque, presente aplicativos de muitos celulares.

Filtros h3 e h4

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

Table 3: Filtro h3

Figure 8: Original city.png

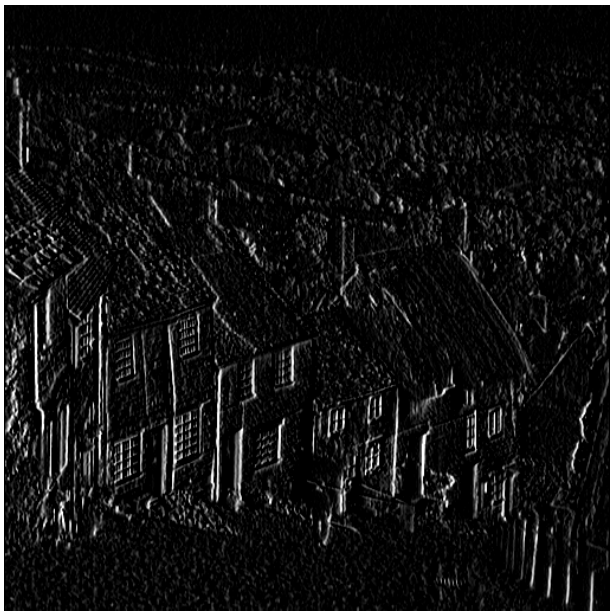

Figure 9: Filtrada com h3 city.png

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

Table 4: Filtro h4

Figure 10: Original city.png

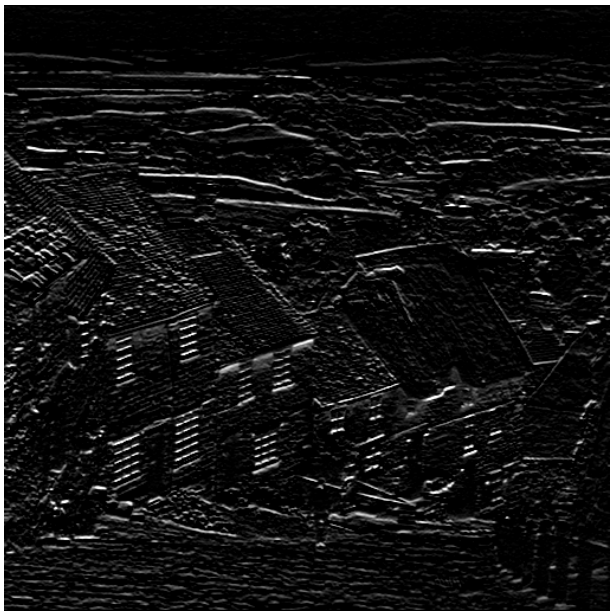

Figure 11: Filtrada com h4 city.png

Os filtros 3 e 4 são muito semelhantes. Ambos tem a mesma característica do filtro h1 de realçar bordas. Porém, o h3 apenas realça bordas na vertical (como pode ser observado nas divisões entre as casas na imagem) e o h4 apenas bordas na horizontal (como pode ser observado nos telhados e nas colinas na parte de cima da imagem).

A combinação entre esses dois filtros realizada de acordo com o enunciado para esse relatório tem como *output* uma sobreposição dos realces verticais e horizontais. Assim, o resultado dessa combinação se assemelha muito com o do filtro h1.

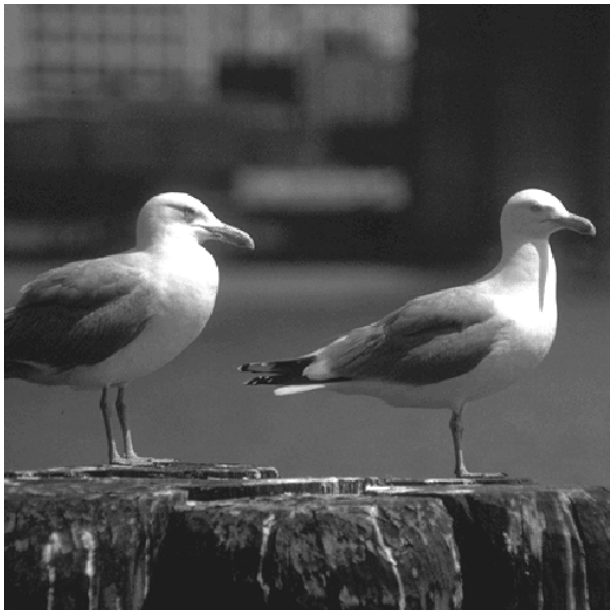

Figure 12: Original seagull.png

Figure 13: Combinação h3 e h4 seagull.png

Filtro h5

-1	-1	-1
-1	8	-1
-1	-1	-1

Table 5: Filtro h5

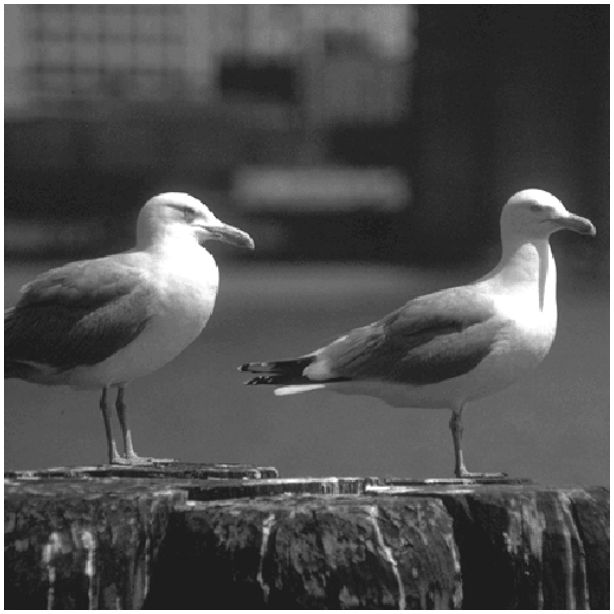

Figure 14: Original seagull.png

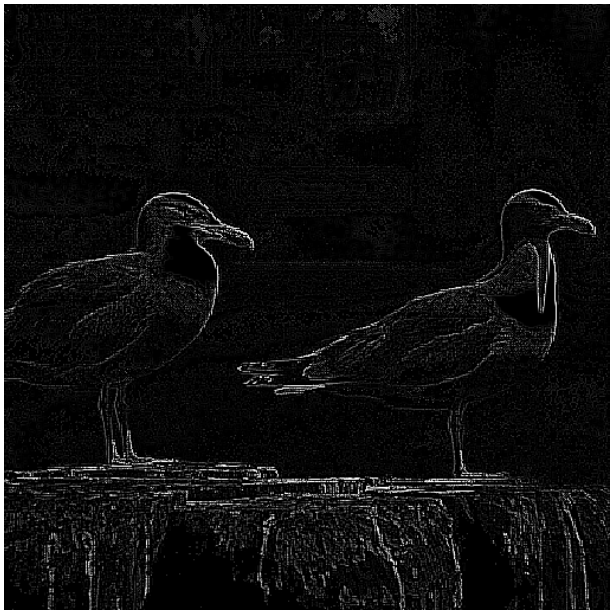

Figure 15: Filtrada com h5 seagull.png

O filtro h5 é muito parecido com a parte central do filtro h1, gerando imagens muito parecidas. Por conta do h5 ser menor, interage com menos pixels em volta das bordas, sendo mais sensível a bordas como diferenças mais abruptas. Isso pode ser notado na diferença de quantidade de bordas encontradas, já que o h1 identificou um número maior.

Filtro h6

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

Table 6: Filtro h6

Figure 16: Original city.png

Figure 17: Filtrada com h6 city.png

O filtro h6 gerou um output extremamente parecido com o h2, não sendo possível detecção de diferenças relevantes entre as imagens. Ambos os filtros fazem uma média do pixel central com seus vizinhos, porém, no h6 os pesos são todos iguais.

Filtros h7 e h8

-1	-1	2
-1	2	-1
2	-1	-1

Table 7: Filtro h7

2	-1	-1
-1	2	-1
-1	-1	2

Table 8: Filtro h8

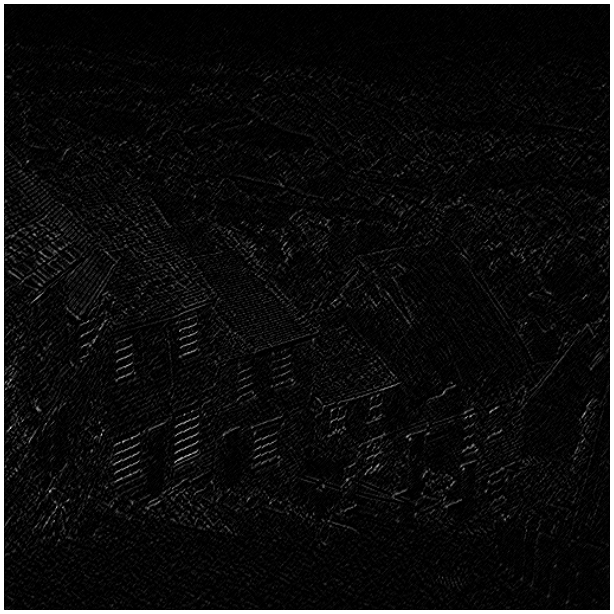

Figure 18: Filtrada com h7 city.png

Figure 19: Filtrada com h8 city.png

Os filtros h7 e h8 são muito semelhantes, ambos operam de forma diagonal sobre a imagem, diferindo apenas na direção. Testando diversas imagens com estes filtros não ficou muito claro o intuíto de sua aplicação, mas parecem realçar padrões diagonais (como pode ser observado nas diversar linhas diagonais presentes nas imagens).

Filtro h9

1/9	0	0	0	0	0	0	0	0
0	1/9	0	0	0	0	0	0	0
0	0	1/9	0	0	0	0	0	0
0	0	0	1/9	0	0	0	0	0
0	0	0	0	1/9	0	0	0	0
0	0	0	0	0	1/9	0	0	0
0	0	0	0	0	0	1/9	0	0
0	0	0	0	0	0	0	1/9	0
0	0	0	0	0	0	0	0	1/9

Table 9: Filtro h9

Figure 20: Original city.png

Figure 21: Filtrada com h9 city.png

O filtro h9, assim como os filtros h6 e h2, gera um borrão na imagem, porém de maneira vertical. Se assemelha a imagens produzidas por fotografias cuja câmera se moveu verticalmente no momento da captura.

Filtro h10

-1/8	-1/8	-1/8	-1/8	-1/8
-1/8	2/8	2/8	2/8	-1/8
-1/8	2/8	8/8	2/8	-1/8
-1/8	2/8	2/8	2/8	-1/8
-1/8	-1/8	-1/8	-1/8	-1/8

Table 10: Filtro h10

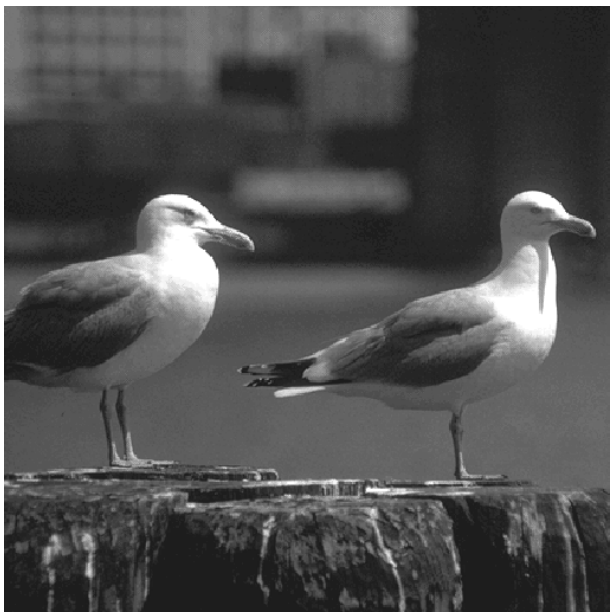

Figure 22: Original seagull.png

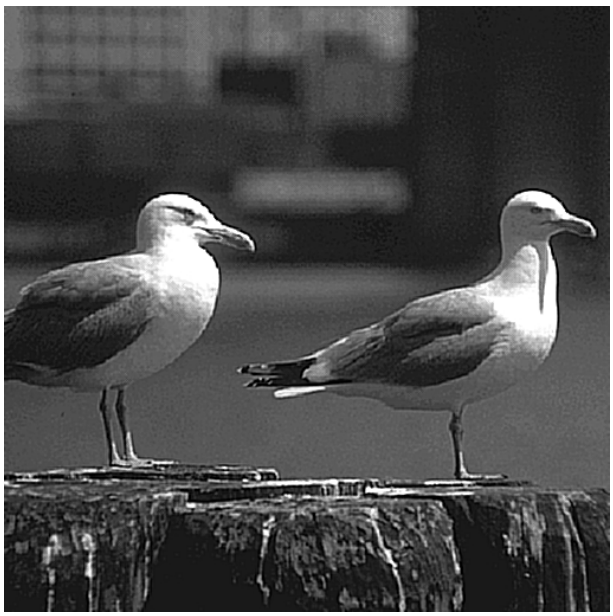

Figure 23: Filtrada com h10 seagull.png

Esse filtro parece aumentar o efeito de bordas, melhorando a nitidez dos detalhes da imagem. Esse efeito pode ser observado tanto nas penas da ave na imagem, que na original pareciam borradas, quanto no objeto que ela está apoiada. Sua matriz se assemelha muito com o h2 nas partes mais externas e com o h6 nas partes internas, provavelmente utilizando características destes dois filtros para obter o efeito desejado.

Filtro h11

-1	-1	0
-1	0	1
0	1	1

Table 11: Filtro h11

Figure 24: Original city.png

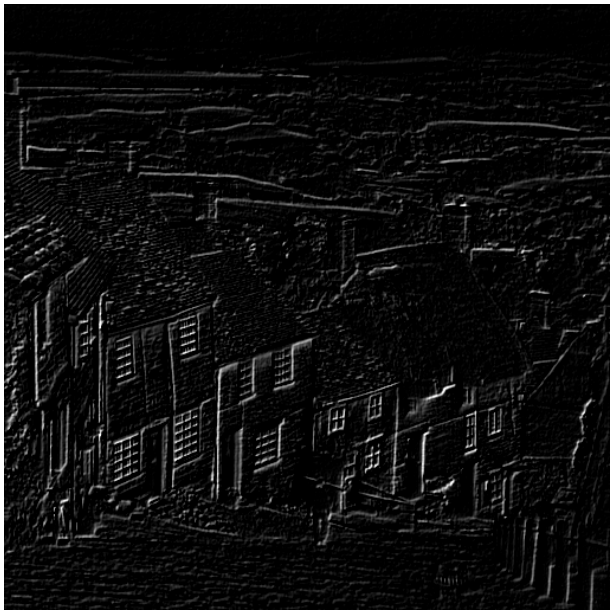

Figure 25: Filtrada com h11 city.png

O filtro h11 parece ter a mesma característica de outros filtros de encontrar bordas, porém enfatiza profundidade e relevo na imagem, efeitos que não são nítidos em filtros anteriores.

4. Conclusões

Muitos desses filtros parecem ser utilizados comercialmente em softwares de processamento de imagem, por conta da forma intuitiva que foram interpretados. Exemplo disso são os de borramento ou desfoque (h2, h6 e h9), os de identificação de bordas (h1, h3, h4, h5 e h11) e o de realce de nitidez(h10). É possível até imagina-los sendo usados em conjunto, por exemplo, em softwares de câmeras que identificam pessoas através de bordas e desfocam o fundo para tirar uma boa fotografia.

Apesar disso, nem todos os filtros foram de fácil entendimento (h7 e h8). Talvez suas aplicações sejam utilizadas apenas como intermediários de processos maiores de filtragem ou simplesmente não são tão comumente utilizados.