Universidade Federal de Alagoas
Instituto de Computação

Processamento Digital de Imagens
Professor Tiago Vieira

Hiago Cavalcante e Luana Ferreira
Reprodução dos Resultados Obtidos

Sumário

1 Introdução

Esse relatório contém uma expansão do trabalho publicado pelos autores do artigo *Non-linear aggregation of filters to improve image denoising* como parte do trabalho que constitui a disciplina de Processamento Digital de Imagens, ministrada pelo professor Tiago Vieira. Esse projeto está em desenvolvimento e encontra-se em

https://github.com/ferreiraluana/dip-project. O artigo escolhido está disponível na *web* em: **https://paperswithcode.com/paper/non-linear-aggregation-of-filters-to-improve**.

2 O filtro BM3D

2.1 O pacote BM3D/PyPI

O framework PyPI da linguagem Python disponibiliza um algoritmo BM3D já implementado. Para utilizar, basta instalar o pacote a partir da seguinte instrução:

```
pip install bm3d
```

BM3D é um algoritmo para atenuação de ruído gaussiano estacionário aditivo espacialmente correlacionado. Este pacote fornece um *wrapper* para os binários BM3D para uso em tons de cinza, cores e outras imagens multicanal para remoção de ruído e desfoque.

Esta implementação é baseada no trabalho de [1] e encontra-se disponível em https://pypi.org/project/bm3d/

2.2 Funcionamento do modelo

O BM3D é um método de remoção de ruído baseado no fato de que uma imagem tem uma representação localmente esparsa no domínio de transformação. Essa dispersão é aprimorada pelo agrupamento de imagens 2D semelhantes patches em grupos 3D [2].

A filtragem colaborativa é o nome do procedimento de agrupamento e filtragem do BM3D. É realizado em quatro etapas:

1. encontrar os patches de imagem semelhantes a um determinado patch de imagem e agrupá-los em um bloco 3D;

2. transformada linear 3D do bloco 3D;

3. encolhimento dos coeficientes do espectro transformado;

4. transformação 3D inversa.

Este filtro 3D, portanto, filtra simultaneamente todos os patches de imagem 2D no bloco 3D. Ao atenuar o ruído, a filtragem colaborativa revela até os menores detalhes compartilhados pelas manchas agrupadas. Os patches filtrados são então devolvidos às suas posições originais. Uma vez que estes patches se sobrepõem, muitas estimativas que são obtidas precisam ser combinadas para cada pixel. Agregação é um procedimento de média específico usado para aproveitar essa redundância.

Como o artigo [3] utilizado neste projeto possui a filtragem colaborativa como objeto principal de estudo, é conveniente adicionarmos esse filtro à implementação original.

O algoritmo BM3d foi primeiramente proposto por [4] e pode ser simplificado em dois passos principais:

1. O primeiro passo estima a imagem removida de ruído usando um limiar **hard** thresholding durante o processo de filtragem colaborativa. Os parâmetros nesta etapa são indicados pelo expoente **hard**;

2. A segunda etapa é baseada tanto na imagem ruidosa original quanto na estimativa básica obtida no primeiro passo. Ele usa filtragem Wiener. A segunda etapa é, portanto, denotada pelo expoente **wiener**.

Uma representação gráfica do modelo BM3D é exemplificada na figura 1:

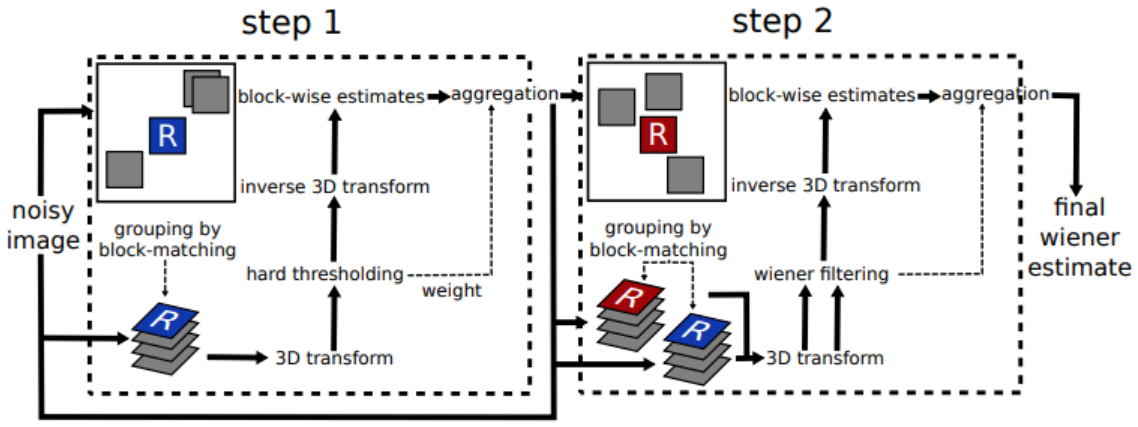

Figura 1: Esquema do algoritmo BM3D. Fonte: Lebrun, 2012.

2.3 Exemplo de implementação

```
import numpy as np
import skimage.data
from skimage.measure import compare_psnr
import pybm3d

noise_std_dev = 40
img = skimage.data.astronaut()
noise = np.random.normal(scale=noise_std_dev,
                         size=img.shape).astype(img.dtype)

noisy_img = img + noise
out = pybm3d.bm3d.bm3d(noisy_img, noise_std_dev)
noise_psnr = compare_psnr(img, noisy_img)
out_psnr = compare_psnr(img, out)
print("PSNR of noisy image: ", noise_psnr)
print("PSNR of reconstructed image: ", out_psnr)
```

Disponível em: https://github.com/ericmjonas/pybm3d

3 Expansão do trabalho

Adição do método de remoção de ruído BM3D através da função a seguir:

```
import bm3d

def bm3d(self):
        """BM3D denoising algorithm"""
        self.Ibm3d = bm3d.bm3d((self.Inoisy*255).astype(np.int), self.
                                                    bm3d_std)
        self.Ibm3d = np.clip(self.Ibm3d/255., 0, 1)
        self.Ilist[self.str2int['bm3d']] = self.Ibm3d
        if self.verbose :
              print('BM3D :', self.Ibm3d)
        return()
```

4 Resultados

Observa-se que o método de remoção de ruídos BM3D possui desempenho satisfatório, porém não obteve nenhuma métrica superior aos outros métodos. Na figura 2, os melhores resultados obtidos para a remoção de ruído gaussiano foram: TV Chambolle para a métrica MAE, KSVD para a métrica RMSE e Richardson Lucy para a métrica PSNR.

Imagem com ruído Gaussiano

Denoise Method	MAE	RMSE	PSNR
Bilateral	0,1415	0,1674	60,9731
NL-means	0,1455	0,1716	58,2049
Gaussian	0,1508	0,1786	59,6858
Median	0,1529	0,1819	60,1439
TV Chambolle	**0,1398**	0,1647	59,7792
Richardson Lucy	0,1475	0,1807	**62,9905**
Inpainting	0,1522	0,1844	62,8125
KSVD	0,1402	**0,1605**	61,0919
Lee	0,1629	0,1936	60,3352
BM3D	**0,1400**	**0,1651**	**59,5500**

Figura 2: Métricas com remoção de ruído em imagem com ruído Gaussiano. Fonte: autores, 2022.

Jà na figura 3, é possível inferir que o método com maior desempenho para remoção de múltiplos ruídos em uma imagem é o TV Chambolle, pois obteve os melhores valores para as métricas MAE, RMSE e PSNR.

Imagem com múltiplos ruídos

Denoise Method	MAE	RMSE	PSNR
Bilateral	0,1092	0,1534	64,4468
NL-means	0,1050	0,1491	64,7647
Gaussian	0,1244	0,1614	64,2234
Median	0,1149	0,1594	64,1028
TV Chambolle	**0,1009**	**0,1454**	**64,9880**
Richardson Lucy	0,1131	0,1605	64,0194
Inpainting	0,1018	0,1564	64,2427
KSVD	0,1000	0,1463	64,8205
Lee	0,1308	0,1689	64,0201
BM3D	**0,1021**	**0,1462**	**64,8288**

Figura 3: Métricas com remoção de ruído em imagem com ruído Gaussiano. Fonte: autores, 2022.

5 Conclusões

Observamos que o método de remoção de ruídos BM3D pode apresentar desempenhos satisfatórios, no entanto, não superiores aos métodos clássicos de remoção de ruídos. Além disso, o método COBRA não se mostrou muito eficiente computacionalmente, apresentando problemas de esgotamento de recursos como memória e processamento.

Como trabalho futuro, poderia ser implementada uma rede neural baseada na ideia do método COBRA para superar essas dificuldades de processamento.

Referências

[1] Ymir Mäkinen, Lucio Azzari, and Alessandro Foi. Collaborative filtering of correlated noise: Exact transform-domain variance for improved shrinkage and patch matching. *IEEE Transactions on Image Processing*, 29:8339–8354, 2020.

[2] Marc Lebrun. An analysis and implementation of the bm3d image denoising method. *Image Processing On Line*, 2012:175–213, 2012.

[3] Benjamin Guedj and Juliette Rengot. Non-linear aggregation of filters to improve image denoising. In *Science and Information Conference*, pages 314–327. Springer, 2020.

[4] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on image processing*, 16(8):2080–2095, 2007.