Soft Computing (2021) 25:8941–8956
https://doi.org/10.1007/s00500-021-05783-5

METHODOLOGIES AND APPLICATION

Image convolution: a linear programming approach for filters design

Giovanni Capobianco[1] · Carmine Cerrone[2] · Andrea Di Placido[1] · Daniel Durand[1] · Luigi Pavone[3] · Davide Donato Russo[1] · Fabio Sebastiano[3]

Accepted: 30 March 2021 / Published online: 26 April 2021

Abstract
Image analysis is a branch of signal analysis that focuses on the extraction of meaningful information from images through digital image processing techniques. Convolution is a technique used to enhance specific characteristics of an image, while deconvolution is its inverse process. In this work, we focus on the deconvolution process, defining a new approach to retrieve filters applied in the convolution phase. Given an image I and a filtered image $I' = f(I)$, we propose three mathematical formulations that, starting from I and I', are able to identify the filter f' that minimizes the mean absolute error between I' and $f'(I)$. Several tests were performed to investigate the applicability of our approaches in different scenarios. The results highlight that the proposed algorithms are able to identify the filter used in the convolution phase in several cases. Alternatively, the developed approaches can be used to verify whether a specific input image I can be transformed into a sample image I' through a convolution filter while returning the desired filter as output.

1 Introduction

In recent years, numerous works on image processing have dealt with image denoising and feature extraction problems. On signal analysis problems, denoising algorithms are fundamental to improve the quality of the data; in particular, in image analysis, the denoising process is essential to improve the image quality (Chen and Fomel 2015). Feature extraction problems are closely related to artificial vision, finding applications in object detection and recognition, motion tracking, identity recognition, and numerous other problems related to automatic photograph and video processing (Bovik 2010, Alpaydin 2009).

One of the basic techniques used for image processing is the convolution and its inverse, the deconvolution (Nussbaumer 2012). Convolution is a mathematical operation that, given two functions f and g, produces a third one that expresses how the shape of one is modified by the other. In the convolution, for real-valued functions, contrary to the cross-correlation operator, $f(x)$ or $g(x)$ is reflected about the y-axis. In addition to the problems mentioned above, convolution has applications that include probability (Harikrishna and Amuthan 2020, Shrivastava 2019), statistics (Wang et al. 2020, Castro et al. 2019), natural language processing (Behera et al. 2019, Afreen et al. 2019), etc. Otherwise, deconvolution is an algorithm-based process used to reverse the effects of convolution on recorded data. As well as convolution, it is used in signal processing and image processing, but it also has several different applications. In general, the objective of deconvolution is to find the solution of a convolution equation $f \times g = h$. To solve this equation, the convolution theorem (Weisstein 2014) proposes an approach:

Theorem 1 *The Fourier transform of a convolution of two functions is equal to the product of the Fourier transform of the two functions.*

Therefore, the approach is not applicable since the function g could not be unique, the function h can contain zeros, and real data could be affected by noise. According to these, while the convolution can always be calculated, this is not possible for deconvolution, due, in particular, to the loss of information that occurs during the process. Several methods have been developed in order to calculate the best possible inverse convolution, such as Van-Cittert deconvolution (Chengqi et al. 1994); Wiener deconvolution (Dhawan et al. 1985); blind deconvolution (Michailovich and Tannenbaum

✉ Carmine Cerrone
carmine.cerrone@unige.it

[1] Department of Biosciences and Territory, University of Molise, via Francesco De Sanctis, 86100 Campobasso, CB, Italy

[2] Department of Economics & Business Studies, University of Genoa, Via Vivaldi 5, 16126 Genoa, Italy

[3] IRCCS NEUROMED, via Atinense, 18, 86077 Pozzilli, IS, Italy

2007); Gaussian elimination (Zhao and Desilva 1998); singular value decomposition (SVD) (Sadek 2012); truncated singular value decomposition (TSVD) (Wu et al. 2017).

In this work, we focus on the deconvolution, defining a new approach to retrieve filters applied in the convolution phase. We examined image filters, such as *Blur*, *Sobel*, *Laplace*, *Emboss*, etc., that are commonly used to enhance specific characteristics of images (e.g., *SobelX* can enhance horizontal lines, *Laplace* extracts contours, etc.). In this paper, we propose an approach able to extract the applied filter starting from the original image and a filtered version of the original one. Most of the existing techniques are able to recover the filter starting from a dataset of possible filters (Torres et al. 2008). Other approaches use linear programming (LP) or integer linear programming (ILP) (Maria and Fahmy 1974, Chottera and Jullien 1982a, b), neural network (Burger et al. 2012), or genetic algorithms (Pedrino et al. 2013, Harvey and Marshall 1996) to generate filters that can reproduce a specific analyzed transformation; some of the main papers about the *filter retrieval problem* (FRP) are the following. Maria and Fahmy 1974 proposed a useful technique that minimizes the p-error criterion in designing two-dimensional digital recursive filters. The approach utilizes an LP formulation of the problem from the frequency domain. Also, Chottera and Jullien (1982a, b) propose a novel technique, based on an LP model, which is able to describe the two-dimensional recursive filters considering the magnitude and phase of the image signal. In particular, the phase is considered linear and is specified as a desired constant group delay. The LP model tries to obtain the minimum approximation error by performing a univariant search in a range of group delay values. Furthermore, Coyle and Lin (1988), Coyle et al. (1989), Gabbouj and Coyle (1991) discussed about an LP flow formulation to solve the problem of finding an optimal stack filter that obtains the minimum *mean absolute error* (MAE) between its output and the desired signal. Another research field about filter design follows what is defined by Harvey and Marshall (1996). They defined a method of designing ad hoc morphological filters for specific tasks. Their idea was to develop an ad hoc filter for specific tasks.

Based on what defined by Coyle *et Al.*, Dellamonica et al. (2007), proposed a new algorithm for optimal MAE stack filter design based on the duality between the filter design problem and the minimum cost network flow problem. Torres et al. (2008), proposed an algorithm for filter generation and filter retrieval. Based on an original image and a filtered version of the original one, the method identifies which filter was applied to the original to obtain the filtered one from a large list of available filters. Considering that the searching phase of the filter sequence is time-consuming, (Pedrino et al. 2013), presented a new approach, namely *IFbyGP*, for constructing image filters using an evolutionary programming approach; it searches for the most efficient operator sequence for a given image processing application.

Considering the importance of the quality of images for fingerprint recognition, Wang et al. (2008) in their paper propose the design of a Log-Gabor filter for image enhancements. Furthermore, in the field of image restoration, Jaiswal and Rajesh (2009) and Khmag et al. (2015) discussed about the generation of denoising filters based on second-generation wavelet transformation. Venkatapparedy et al. (2017), proposed two methods based on threshold decomposition and stacking properties for generate median filter.

1.1 Filtering example and notations

In this section, we present the notation used in this paper and provide an example that describes how a filter is applied to a specific image and how the MAE is computed.

In this work, we will use grayscale images; even considering only one channel, trivially, our work can be extended to images with more channels as RGB. Furthermore, as shown in Figs. 1a, b, we will consider the pixels' value in the continuous range [0, 1] even if they are generally represented by the discrete integer range [0, 255].

Given an image I and given a filter $f = k \times h$, we indicate with $I' = f(I)$ the image resulting by the application of the filter f to the image I. To obtain the output image I', the filter matrix is applied pixel by pixel; in detail, the value of a generic pixel $I'_{i,j}$ is computed using the following equation:

$$f(I_{i,j}) = I'_{i,j} = \sum_{a=1}^{k} \sum_{b=1}^{h} I_{(i-\lfloor \frac{k}{2} \rfloor + a - 1)(j-\lfloor \frac{h}{2} \rfloor + b - 1)} \times f_{ab} \tag{1}$$

Figure 1 shows an application example of the *SobelX* filter f on an input image I, producing as result the image I'. In the case of h or k even, the indexes need to be changed accordingly. In the rest of the work, we assume h and k as uneven.

Given the example in Fig. 1, Eq. 1 can be exploded as follows:

$$\begin{aligned} I'_{i,j} = {} & I_{i-1,j-1} \times f_{1,1} + I_{i-1,j} \times f_{1,2} + I_{i-1,j+1} \times f_{1,3} \\ & + I_{i,j-1} \times f_{2,1} + I_{i,j} \times f_{2,2} + I_{i,j+1} \times f_{2,3} \\ & + I_{i+1,j-1} \times f_{3,1} + I_{i+1,j} \times f_{3,2} + I_{i+1,j+1} \times f_{3,3} \end{aligned} \tag{2}$$

In order to calculate the difference between two images, we used the mean absolute error (MAE). It is computed as the mean of the differences pixel by pixel in absolute values of the two images. More formally, let I' and I'' be two images,

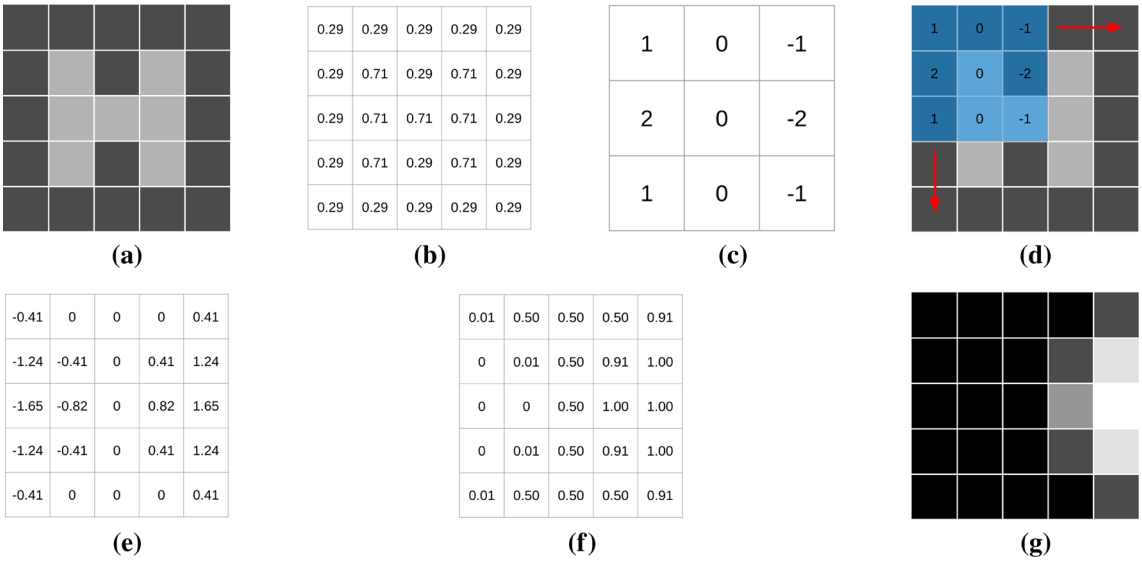


Fig. 1 Example of filter applied to an image I. **a** and **b** show the original 5×5 images and the matrix of pixels values, respectively. **c** shows the filter matrix applied on the input image in Fig. **d**. **g** shows the resulting filtered images, and **e** shows the resulting matrix of pixels values after the convolution. **f** highlights the resulting matrix after the threshold cut application

the MAE is computed according to the formula:

$$MAE(I', I'') = \frac{\sum_{i=1}^{w(I')} \sum_{j=1}^{h(I')} |I'_{i,j} - I''_{i,j}|}{w(I') \times h(I')} \tag{3}$$

1.2 Problem definition

In this paper, we propose a mathematical formulation able to generate a filter that minimizes the mean absolute error. Given an image I and a filtered image $I' = f(I)$, we define a mathematical formulation that, starting from I and I', is able to identify the filter f' that minimizes the MAE between I' and $I'' = f'(I)$. In real cases, I' is not exactly equal to $f(I)$, due to possible noise or threshold error, in particular, $I' = f(I) + \epsilon$, where ϵ is the error of the output image. Our formulation is able to produce a filter f' that could be equal or equivalent to f ($\epsilon = 0$), otherwise ($\epsilon > 0$) similar. We propose a simplified version of the formulation that does not consider any cut or threshold to the pixel values. Then, a normalized version of the model is proposed, in which the value of the pixel is bounded between 0 and 1. Finally, a third formulation considers the activation variables on the pixels for image thresholding. We clarify that all models are formulated to generate a $k \times h$ filter with each component of the filter in the range $[-\delta, \delta]$ (δ is a parameter).

The importance of this approach, firstly, concerns the possibility of exactly model the FRP, and, also, allow us to certificate the goodness of the obtained filter. In this way, it is possible to certificate if a given transformation can be obtained using a $k \times h$ filter or not. The results obtained show that our approach is competitive with respect to the state of the art when the images are noise-free; in fact, we are capable to retrieve the used filter in all the considered cases.

The paper is organized as follows: Sect. 2 describes in detail the mathematical formulation of the FRP, outlining the different formulations and the possibilities of each one; Sect. 3 describes the experiments performed to evaluate the goodness of the approach and the results obtained. Finally, Sec. 4 provides a brief discussion on the FRP and proposes different possibilities for future works.

2 Mathematical formulation

2.1 LP-model

Given an input image I, we want to find

$$f' = \begin{bmatrix} x_{11} & \dots & x_{h1} \\ \vdots & \ddots & \vdots \\ x_{1k} & \dots & x_{hk} \end{bmatrix}$$

such that $MAE(I', f'(I))$ is minimum. For each $x_{ij} \in f'$, $x_{ij} \in [-\delta, \delta]$, with $\delta \in \Re$. Let $c_{i,j}$ be an auxiliary variable that represents the value of the pixel $i, j \in I$ applying the filter f' on it. Let $e_{i,j}$ be a variable equal to the absolute value of the difference between $c_{i,j}$ and $I'_{i,j}$. We want to:

$$Minimize \sum_{i,j} e_{i,j} \qquad \forall i, j \in I \tag{4}$$

s.t.

$$c_{i,j} = f'(I_{i,j}) \qquad \forall i, j \in I \tag{5}$$
$$c_{i,j} - I'_{i,j} \leq e_{i,j} \qquad \forall i, j \in I \tag{6}$$
$$I'_{i,j} - c_{i,j} \leq e_{i,j} \qquad \forall i, j \in I \tag{7}$$
$$x_{i,j} \in [-\delta, \delta] \qquad \forall i, j \in I \tag{8}$$
$$e_{i,j}, c_{i,j} \in \Re \qquad \forall i, j \in I \tag{9}$$

Constraint 5 binds the value of the pixel $i, j \in I$ to be equal to $f'(I_{i,j})$, while constraints 6 and 7 are used to impose $e_{i,j}$ to be greater than or equal to the error between $I'_{i,j}$ and $c_{i,j}$ in absolute value.

Since the model is composed of linear equations and does not have integer variables, the solution can be provided in polynomial time. Nevertheless, applying a filter on an image could produce pixel values that exceed the interval [0,1], as can be seen from Fig. 1e. This leads the LP model to produce an implicit error on filter generation, due to the truncation of the pixel values. In fact, the MAE is calculated on the thresholded images and not on the values obtained. According to this, a second mathematical formulation is defined, which considers this boundary.

2.2 ILP-model for normalization

In this model, we introduce $c'_{i,j}, c^0_{i,j}, c^1_{i,j}$ defined as follows:

$$c'_{i,j} = \begin{cases} 0 & \text{if } c_{i,j} \leq 0 \\ c_{i,j} & \text{if } c_{i,j} \in (0,1) \\ 1 & \text{if } c_{i,j} \geq 1 \end{cases}$$
$$c_{i,j} \geq 1 \implies c^1_{i,j} = 1$$
$$c_{i,j} \leq 0 \implies c^0_{i,j} = 1$$

$c'_{i,j}$ is an auxiliary variable that represents the value of the pixel $i, j \in I$ applying the filter f' on it. $c^0_{i,j}$ and $c^1_{i,j}$ are two auxiliary variables used to normalize the values of $c'_{i,j}$. Let M be a sufficient big number[1], we want to minimize our objective function (4) s.t. constraint (5) and all the following constraints are satisfied:

$$c^0_{i,j} + c^1_{i,j} \leq 1 \qquad \forall i, j \in I \tag{10}$$
$$c_{i,j} + Mc^0_{i,j} \geq 0 \qquad \forall i, j \in I \tag{11}$$
$$c_{i,j} - Mc^1_{i,j} \leq 1 \qquad \forall i, j \in I \tag{12}$$
$$c'_{i,j} \leq c_{i,j} + Mc^0_{i,j} \qquad \forall i, j \in I \tag{13}$$
$$c'_{i,j} \geq c_{i,j} - Mc^1_{i,j} \qquad \forall i, j \in I \tag{14}$$
$$c'_{i,j} \geq c^1_{i,j} \qquad \forall i, j \in I \tag{15}$$
$$c'_{i,j} \leq 1 - c^0_{i,j} \qquad \forall i, j \in I \tag{16}$$
$$c'_{i,j} - I'_{i,j} \leq e_{i,j} \qquad \forall i, j \in I \tag{17}$$
$$I'_{i,j} - c'_{i,j} \leq e_{i,j} \qquad \forall i, j \in I \tag{18}$$
$$x_{i,j} \in [-\delta, \delta] \qquad \forall i, j \in I \tag{19}$$
$$e_{i,j}, c_{i,j}, c'_{i,j} \in [0,1] \qquad \forall i, j \in I \tag{20}$$
$$c^0_{i,j}, c^1_{i,j} \in \{0,1\} \qquad \forall i, j \in I \tag{21}$$

Constraint 10 assures that at most one variable among $c^0_{i,j}$ and $c^1_{i,j}$ is active (equal to 1). Constraint 11 imposes $c^0_{i,j}$ to be equal to 1 when $c_{i,j} \leq 0$, while constraint 12 enforces $c^1_{i,j}$ to be equal to 1 when $c_{i,j} \geq 1$. Constraint 13 ensures that the value of $c'_{i,j}$ will be less than or equal to the value of $c_{i,j}$ if $c^0_{i,j}$ is not active. Constraint 14 ensures that the value of $c'_{i,j}$ will be greater than or equal to the value of $c_{i,j}$ if $c^1_{i,j}$ is not active. Constraint 15 enforces $c'_{i,j} = 1$ if $c^1_{i,j} = 1$; contrariwise, constraint 16 imposes $c'_{i,j} = 0$ if $c^0_{i,j} = 1$.

The combination of the constraints guarantees that $c'_{i,j}$, $c^1_{i,j}$ and $c^0_{i,j}$ are set correctly: Constraints 11 and 12 do not assure that $c^1_{i,j}$ or $c^0_{i,j}$ will not be activated if $c_{i,j} \in (0,1)$. In fact, this condition is respected considering constraints 13 and 14. Furthermore, 13 and 14 do not certify that $c'_{i,j}$ value will be 0 or 1, but with 15 and 16 we enforce it.

Finally, constraints 17 and 18 are used to impose $e_{i,j}$ to be greater than or equal to the error between $I'_{i,j}$ and $c'_{i,j}$ in absolute value.

2.3 ILP-model for thresholding

A third mathematical formulation regards the thresholding of the output image. The thresholding is one of the most popular image processing techniques used, in particular in image segmentation (Arora et al. 2008, Sathya and Kayalvizhi 2011), and object detection algorithm (Park 2001). Let I be our original image and $I' = f'(I)$ be our filtered image, applying a threshold to I' corresponds to create an image I'' s.t. $I''_{i,j} \in \{0,1\}, \forall i, j \in I''$. The value $I''_{i,j}$ is equal to 1 if $I''_{i,j} \geq t$, otherwise is equal to 0. The result of this process is a binary image in which the white pixels usually highlight important features of the original image. For instance, if in

[1] The value of M was calculated considering the maximum value of the pixel after applying the filter f'. According to Eq. (1), to obtain the maximum value we will have that $\sum_{a=1}^{k} \sum_{b=1}^{h} I_{(i-\lfloor \frac{k}{2} \rfloor + a - 1)(j - \lfloor \frac{h}{2} \rfloor + b - 1)} = k \times h$ and $f_{ab} = \delta$, for $a = 1, ..., k$ and $b = 1, ..., h$. Hence, the value of $M = k \times h \times \delta$

Fig. 1f we consider a thresh value of 0.9, only the last column and the three central pixels of the image will be set to 1 in the output image. Let $t \in [0, 1]$ be the threshold value and $y_{i,j} \in \{0, 1\}$ be the auxiliary variable, which represents the value of the pixel ij applying f' and the thresholding on it. We want to minimize our objective function (4) s.t. constraint (5), and all the following constraints are satisfied:

$$c_{i,j} - t \leq M y_{i,j} \qquad \forall i, j \in I \qquad (22)$$

$$t - c_{i,j} \leq M(1 - y_{i,j}) \qquad \forall i, j \in I \qquad (23)$$

$$y_{i,j} - I'_{i,j} \leq e_{i,j} \qquad \forall i, j \in I \qquad (24)$$

$$I'_{i,j} - y_{i,j} \leq e_{i,j} \qquad \forall i, j \in I \qquad (25)$$

$$x_{i,j} \in [-\delta, \delta] \qquad \forall i, j \in I \qquad (26)$$

$$e_{i,j}, c_{i,j}, t \in [0, 1] \qquad \forall i, j \in I \qquad (27)$$

$$y_{i,j} \in \{0, 1\} \qquad \forall i, j \in I \qquad (28)$$

Constraints 22 and 23 exploit if $c_{i,j}$ overcomes the threshold, so if the pixel should be activated or not. If $c_{i,j} > t$, $y_{i,j} = 1$ and the pixel is activated, otherwise $y_{i,j} = 0$. Constraints 24 and 25 are used to impose $e_{i,j}$ to be greater than or equal to the error between $I'_{i,j}$ and $y_{i,j}$ in absolute value.

3 Computational experiments

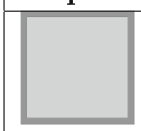
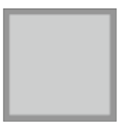
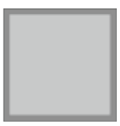
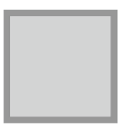
Several tests were performed to verify the effectiveness of our methods. All the experiments were performed on Windows 10 OS, with AMD Ryzen 7 3750H Processor 2.3 GHz CPU and 16 GB of RAM. The algorithms were implemented in JAVA 14, and the mathematical formulation was solved using CPLEX version 12.9. In general, for our computational experiment (if not explicitly described), we defined the size of the filter 3x3 and a range $[-10, 10]$ with $\delta = 10$. We considered a 3x3 filter size because this is one of the most common kernel sizes in the literature (Simonyan and Zisserman 2014). Therefore, some works compared the behavior of different sizes for different purposes; these sizes generally go from 1x1 to 7x7 (He et al. 2016, Szegedy et al. 2015). Even if several tests confirm that small sizes are better for image processing, in the field of semantic segmentation larger kernel, up to 15x15, can produce better results (Peng et al. 2017).

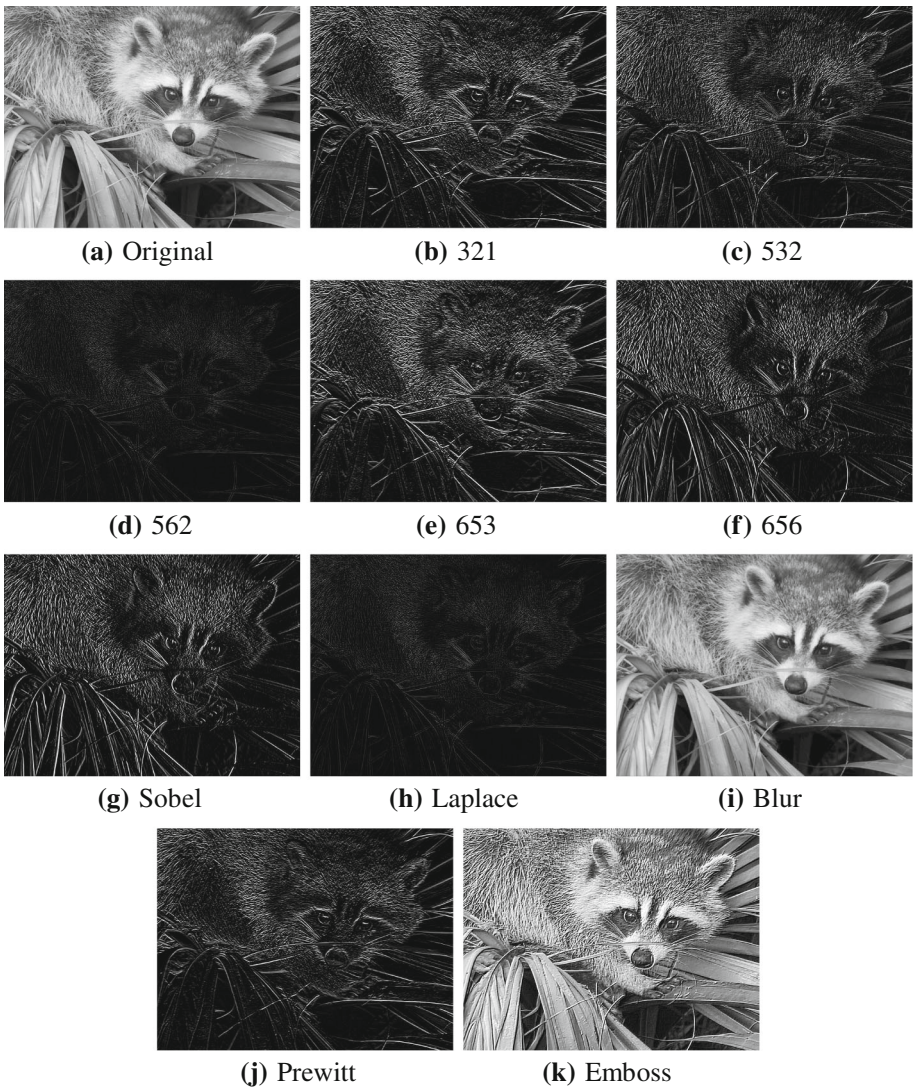
Furthermore, another critical parameter that should be set is the range of the filter. As said, we set that range to $[-10, 10]$ even if generally this range is not bounded; however, after several tests, described in Sect. 3.5, we can affirm that even bounding the filter range to $[-10, 10]$, the performance of our approaches is not affected.

In the remaining of this paper, we named as f_i a generic filter; for more information regarding the filters, refer to Appendix A.

In all the performed tests, the computation time reported is shown in seconds; it is essential to clarify that in the following tests, the time limit for the execution of the proposed approaches was set to 1800 seconds. According to this, when the time limit is reached, the models return the best filter found.

Table 1 Computational results of LP model on well-known filter cases

input	filter	output	LP filter	produced	MAE
	f_{sobel}		f_1		0.6%
	$f_{laplace}$		f_2		0.2%
	f_{blur}		f_3		2.0%
	$f_{prewitt}$		f_4		0.2%
	f_{emboss}		f_5		1.8%



(a) Original **(b)** 321 **(c)** 532

(d) 562 **(e)** 653 **(f)** 656

(g) Sobel **(h)** Laplace **(i)** Blur

(j) Prewitt **(k)** Emboss

Fig. 2 Racoon Test images. **a** is the input image and **b**, **c**, **d**, **e**, **f**, **g**, **h**, **i**, **j** and **k** are the output images obtained applying f_{321}, f_{532}, f_{562}, f_{653}, f_{656}, f_{sobel}, $f_{laplace}$, f_{blur}, $f_{prewitt}$ and f_{emboss}, respectively

Several experiments have been designed to study the behavior of the developed models. Test A analyzes the ability of the models to identify filters applied to reference images. Test B studies the robustness of the proposed models, particularly their tolerance to the noise present on the input data. Test C verifies that the models can identify filters capable of emphasizing specific characteristics of the images. Test D aims to verify the impact of the image sample size on the quality of the solution produced. Finally, test E checks the behavior of the model if we try to identify filters of different sizes than those initially used for creating the sample images.

3.1 Test A: identification of well-known filters

In this specific test, we performed our experiment on a geometrical benchmark. We used grayscale images with dimension equal to 32×32 pixels. We performed several tests applying well-known filters, i.e., f_{sobel}, $f_{laplace}$, f_{blur}, $f_{prewitt}$, f_{emboss}. Table 1 shows the filters produced by LP model. The ILP model is able to identify the filters optimally with computational times not exceeding 0.30 seconds. For this reason, ILP results are omitted from the table. In the first three columns, the input image, the filter applied, and the resulting image are reported. Then, the fourth and fifth columns display the filter obtained by the model and the output image produced by applying this filter on the input image. Finally, the last column reports the percentage MAE according to the formula (3).

The LP model provides filters able to reproduce the output image with low MAE, which does not exceed the 2%.

To test further the proposed approaches, we conducted several experiments on larger images: Fig. 2a and Fig. 3a. Both the images are grayscale. Figure 2a is 1024x768 and Fig. 3a is 512x512 as size. On these samples, we have applied several filters: in addition to the well-known filters, f_{321}, f_{532},

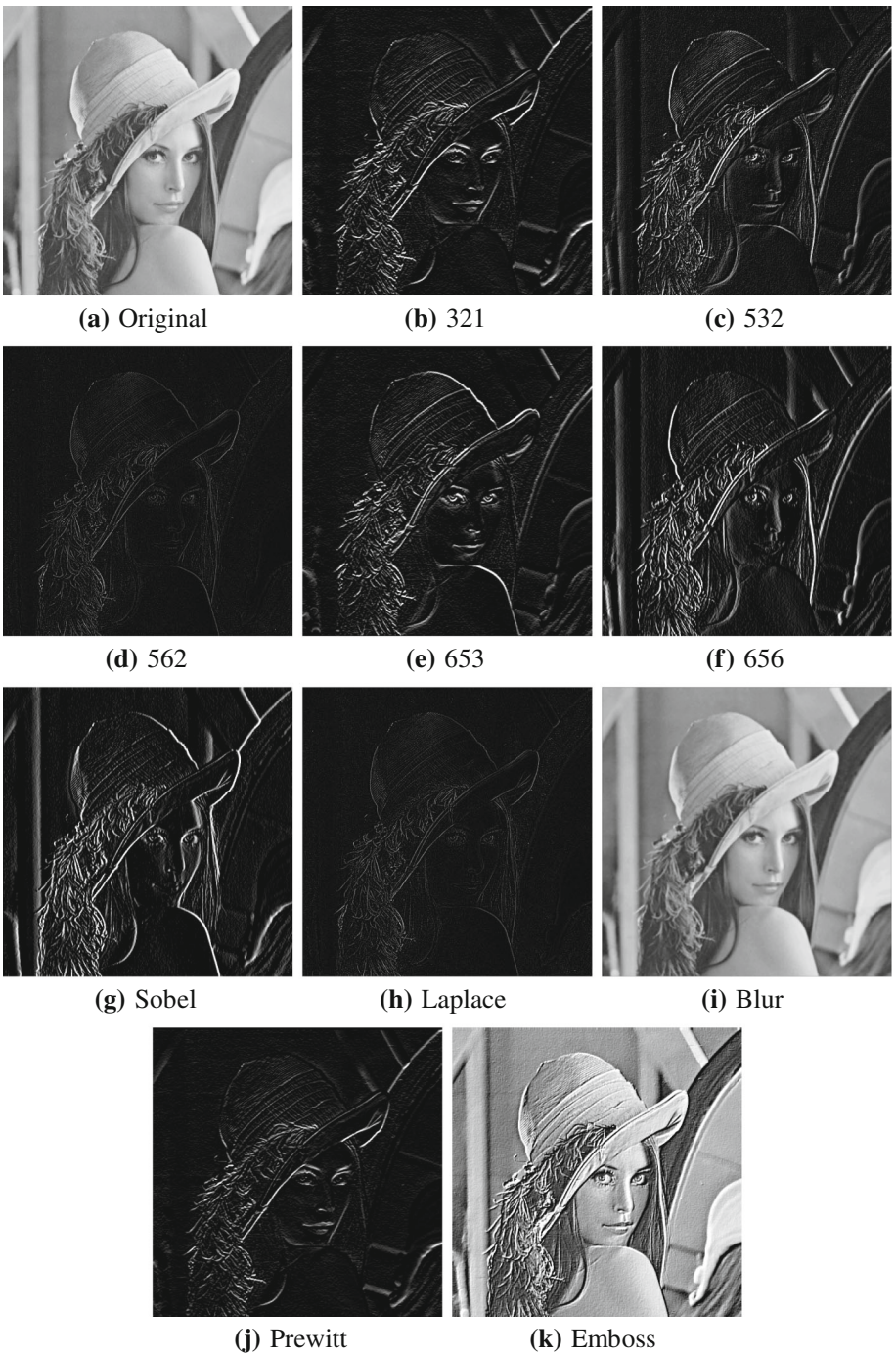


Fig. 3 Lena Test images. **a** is the input image and **b**, **c**, **d**, **e**, **f**, **g**, **h**, **i**, **j** and **k** are the output images obtained applying f_{321}, f_{532}, f_{562}, f_{653}, f_{656}, f_{sobel}, $f_{laplace}$, f_{blur}, $f_{prewitt}$ and f_{emboss}, respectively

f_{562}, f_{653}, f_{656} are applied on them. Figures 2 and 3 show the output images.

We considered a portion of the image to execute our test, i.e., the top-left corner with size 20x20 pixels, starting from (0,0) to (19,19). The results are reported in Table 2, organized as follows: The first two columns report the sample used and the ID of the filter applied. Then, the last two columns show the results obtained by LP and ILP model. Each result is composed of the filter computed by the model, the percentage MAE relative to the original output image and the image produced applying the obtained filter, according to the formula 3, and the computational times in seconds.

Since the images under test I are grayscale, the value of each pixel i is defined in a range [0, 1]. The application of a filter f could increase or decrease the value of i over the limits. This introduces an error due to the truncation of the value to 1 or 0.

Table 2 Computational results for test images

Sample	ID	LP Filter	MAE	Time	ILP Filter	MAE	Time
Racoon	321	fpl_{321}	18.3%	0.02	f_{321}	0.0%	0.16
	532	$f'pl_{532}$	10.3%	0.02	f_{532}	0.0%	0.18
	562	$f'pl_{562}$	2.8%	0.02	f_{562}	0.0%	0.12
	653	$f'pl_{653}$	18.1%	0.02	f_{653}	0.0%	0.20
	656	$f'pl_{656}$	8.4%	0.02	f_{656}	0.0%	0.22
	Sobel	$f'pl_{sobel}$	9.0%	0.02	f_{sobel}	0.0%	0.21
	Laplace	$f'pl_{laplace}$	2.8%	0.02	$f_{laplace}$	0.0%	0.12
	Blur	$f'pl_{blur}$	0.1%	0.02	$f'pl_{blur}$	0.1%	0.33
	Prewitt	$f'pl_{prewitt}$	13.9%	0.02	$f_{prewitt}$	0.0%	0.17
	Emboss	$f'pl_{emboss}$	5.7%	0.02	f_{emboss}	0.0%	0.20
Lena	321	$f''pl_{321}$	1.1%	0.02	f_{321}	0.0%	0.22
	532	$f''pl_{532}$	3.5%	0.02	f_{532}	0.0%	0.11
	562	$f''pl_{562}$	1.4%	0.02	f_{562}	0.0%	0.25
	653	$f''pl_{653}$	3.6%	0.02	f_{653}	0.0%	0.25
	656	$f''pl_{656}$	3.8%	0.02	f_{656}	0.0%	0.20
	Sobel	$f''pl_{sobel}$	3.0%	0.02	f_{sobel}	0.0%	0.20
	Laplace	$f''pl_{laplace}$	1.4%	0.02	$f_{laplace}$	0.0%	0.22
	Blur	$f''pl_{blur}$	0.2%	0.02	$f''pl_{blur}$	0.2%	0.33
	Prewitt	$f''pl_{prewitt}$	1.0%	0.02	$f_{prewitt}$	0.0%	0.19
	Emboss	f_{emboss}	0.0%	0.02	f_{emboss}	0.0%	0.09

Despite this, as reported in Table 2, ILP model is able to correctly identify all the filters applied on the test image, except for f_{blur} case, for which the error is always lower than 0.2.

3.2 Test B: filter identification on disrupted images

A further test was performed considering a noisy image: We disrupted the output images 2b, 2c, 2d, 2e and 2f with different percentage of noise, i.e., 1%, 3%, 5% and 10%. As we can notice in Fig. 4, the disruption applied on the image does not affect excessively the ILP result, which is able to provide filters with $MAE \leq 4\%$.

Considering our computational experiments, the results shown in Table 1, 2 and Fig. 4, allow us to affirm that our approach is able to identify a big set of well-known filters even when the image is altered with noise, with low MAE.

3.3 Test C: identification of filters for features enhancement

In this experiment, we investigated the applicability of our models to the retrieval of filters that can emphasize features highlighted manually. To verify this question, we tested our model considering samples with different shapes, e.g., hexagon, flower, rectangle and triangle. For each sample, we have manually highlighted several features: For hexagon, triangle and rectangle we emphasized the vertices; for hexagon and triangle, we strongly marked the edges. Finally, for the flower, we considered the petals.

Table 3 shows the computed filter for the proposed cases. The table is organized as follows: In the first two columns, the input image and the features to highlight are shown. The third column displays the image produced with the filter computed by the ILP model. Finally, the last two columns report the computational time in seconds and the MAE in percentage.

The results provided by the LP model are not reported because, in most of the cases, the solution provided is a *all-black* filter, the one that makes the image entirely black. This is due to the fact that we want to minimize the error by providing a good filter. In our instances, the percentage of black pixels is higher than the white ones. Then, the *all-black* filter has a low absolute MAE value, but the same MAE relative to the white-pixel percentage in the image is massive. Our approach is able to identify all the filters related to the edge detection reasonably. In particular, cases in which we want to identify a filter that emphasizes discontinuous regions, i.e., the vertices, our approaches approximate them with a continuous one.

Since the percentage of black pixels affects the overall results of our approach, we performed another test by feature extraction. We considered 4 cases: the angle of the hexagon, the petals of the flower, the angle of the square and the angle of the triangle. Table 4 shows the results of LP and ILP model.

By feature extraction, LP and ILP are able to identify better filters; generally, the results reported in this section show

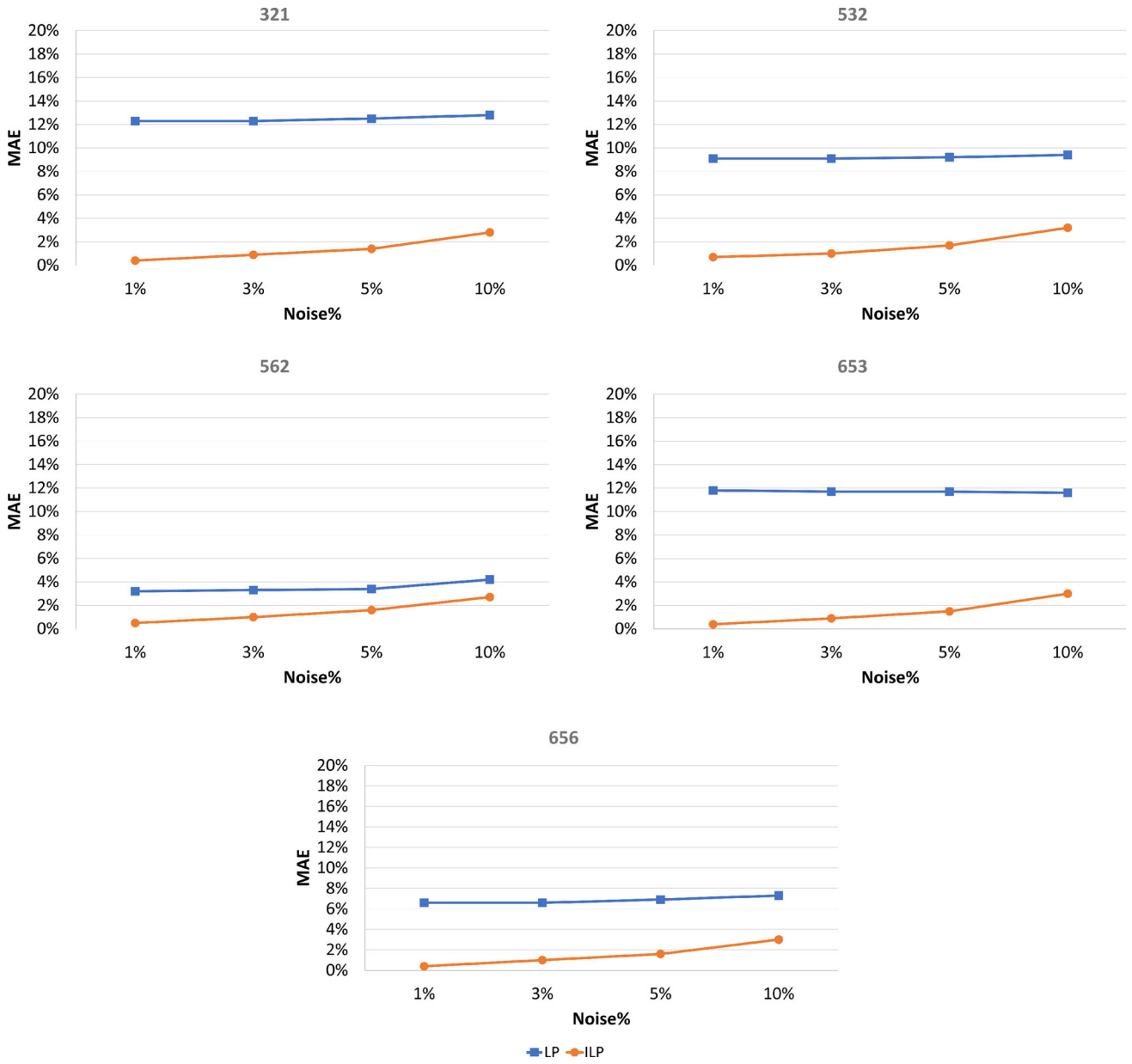


Fig. 4 MAE analysis computed on test images (2) with different percentages of noise. For each graph, on the x-axis is reported the noise percentage, instead on the y-axis the MAE percentage; the blue line refers to the linear programming model described in Sect. 2.1, while the orange line refers to the integer linear programming model shown in Sect. 2.2

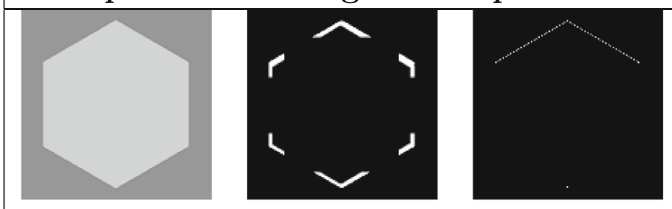
that our approach, with just a single image in which the feature is manually highlighted, is capable of identifying a filter $k \times h$ useful to emphasize the desired characteristics, with respect to the techniques as neural networks (Burger et al. 2012) which needs ad hoc training set to recognize features. Furthermore, an important consideration about these special tests is that if our approach cannot produce a reliable filter capable of emphasizing the desired feature, we can affirm that it is not possible to enhance the feature through filters.

3.4 Test D: robustness tests

When working on large images, a fragment of the image (a sample) is selected to decrease the processing time. In this experiment, we verified the correlation between the selected sample and the result obtained, in order to test the stability of the proposed method.

When considering a sample of an image, it is crucial to take into account the size and the significance of the sample.

Table 3 Computational results of ILP model on special scenario filter cases

input	target	produced	time	MAE
			1800	24%
			1800	2%
			1800	3%
			67.86	12%
			1800	15%
			42.00	1%

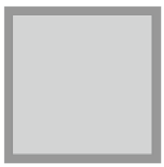
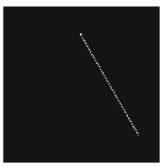
To see if the size of a sample affects the final result, we tested our approaches on four different sample size: 20×20, 50×50, 100×100, 200×200 (Image 2a). Each sample was elaborated starting from (0,0). The results are reported in Fig. 5. The ILP results are omitted because the model could identify the filters optimally not exceeding 0.30 seconds of computational time.

It is possible to notice that the LP model is size dependent. Moreover, in the majority of the cases, the more extensive is the sample, the lower is the error committed.

To verify if a different sample can affect the final result, we tested our approaches on four different samples of the same image with different variances. The samples considered are shown in Fig. 6 and exploited in Table 5 and Table 6. In Table 5 is reported the top-left (starting point) and bottom-right (ending point) corner of the image portion considered, while in Table 6 is reported the percentage variance of each portion.

The results are shown in Fig. 7. The ILP results are omitted because the model could identify the filters optimally not

Table 4 Computational results of LP and ILP model on features filter cases

input	feature	LP	ILP	time LP	time ILP	MAE LP	MAE ILP
				0.00	0.06	4%	3%
				0.00	4.51	8.7%	8.7%
				0.00	1.06	14%	14%
				0.00	1.11	4%	3%

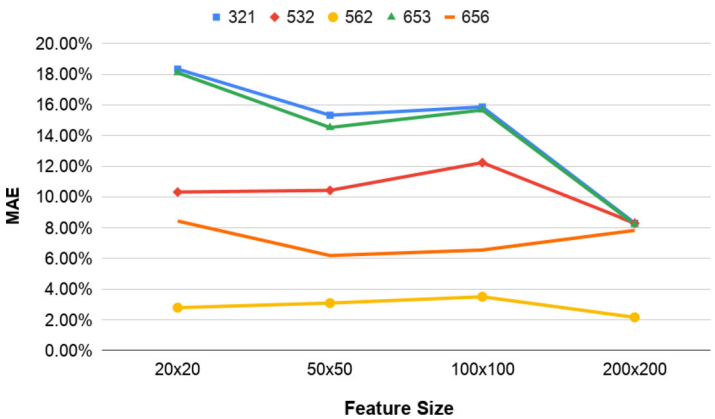


Fig. 5 MAE analysis computed on test images (2) with different sizes of the same feature. The feature considered for this set of tests is the top-left corner of the test images with the size of 20x20, 50x50, 100x100 and 200x200. In the graph, on the x-axis is reported the feature size, instead on the y-axis, the MAE percentage; each line represents the result of the linear programming model 2.1 computed on different images. The integer linear programming model is not reported because it is able to identify the filter optimally with a MAE equal to 0 in all the cases

exceeding 0.30 seconds of computational time. It is possible to notice that the LP model is sample dependent, but there is no strong relation between the variance of the sample considered and the output obtained. Nevertheless, if the sample variance is beyond 10%, the LP model elaborates filters that produce images with relatively high MAE with respect to the original one.

3.5 Test E: filters with different range and size

In this paper, we defined a priori the size and range of the filter values that we want to identify.

In this section, we wanted to study the behavior of our models in case a different size or range of values is set for the filter to be searched with respect to the original filter that produced the input image.

Fig. 6 The features considered for the test described in Sect. 3.4. These feature are reported in Table 5

In particular, our models take as input the size of the filter $k \times h$, and δ which defines the range of values for a generic element of the filter f'_{ab}. According to this, given the original filter f and the computed filter f', we wanted to study what happens when f' has different size or range with respect to f. For what concerns the different size, the experiment was performed on the test image 2a on which were applied five different filters: f_{701}, f_{811}, f_{911}, f_{532}, f_{562}. We produced the filters with LP and ILP models considering different ranges: $[-2, 2]$, $[-4, 4]$, $[-6, 6]$, $[-8, 8]$ and $[-10, 10]$ and elaborated the output images. The results are reported in Table 7, organized as follows: In the first column is described the range considered by the model and the filter under test, with range interval showed in square brackets; the last two columns report the percentage MAE and the computational time for LP and ILP model, respectively.

As it is reported, the ILP model is range dependent: Let $Max(f)$ be the element of f with the maximum absolute value, the more δ is closer to $Max(f)$, the lower is the MAE among the images. Instead, the LP model is not range dependent on mid-high δ values. Furthermore, the computational times of ILP model follow a descending trend for $\delta \leq Max(f)$. Then, the times are constant for $\delta \geq Max(f)$.

For the identification of the filter of different size, we applied on the test image 2a three different 5x5 filters: $f_{sharpen}$, $f_{verticaledges}$, $f_{emboss5x5}$. We elaborated filters with LP and ILP models considering different size as input: 3×3, 5×5 and 7×7. We produced the output images and computed the MAE according to Eq. 3. The computation times of the LP model are not reported because it can provide its solutions in less than 1 second. The results are reported in Table 8.

When the size of the filter to compute is smaller than the size of the original one, ILP reaches the time limit. Nevertheless, the computed filters are able to produce images with $MAE \leq 12\%$. Instead, when the size of the filter to compute is greater than or equal to the original one, ILP is able to identify the applied filter correctly. It is essential to notice that when the size of the filter to compute is greater than the original one, ILP identifies the filter, but it produces a scaled version of it. An example is provided as follows:

$$f_{sharpen} = \begin{bmatrix} -1 & -1 & -1 & -1 & -1 \\ -1 & 2 & 2 & 2 & -1 \\ -1 & 2 & 1 & 2 & -1 \\ -1 & 2 & 2 & 2 & -1 \\ -1 & -1 & -1 & -1 & -1 \end{bmatrix}$$

Table 5 Features considered for the test. The table reports the top-left (start) and the bottom-right (end) corners of the studied features. These are reported graphically in Fig. 6

Name	Start	End
TopLeft	(0,0)	(19,19)
Nose	(600,450)	(619,469)
Eye	(550,300)	(569,319)
Ear	(450,50)	(469,69)

Table 6 Variances of the samples considered

Variance	TopLeft	Nose	Eye	Ear
321	11.500%	1.400%	2.400%	4.600%
532	6.500%	0.600%	3.300%	0.500%
562	1.000%	0.006%	0.230%	0.130%
653	13.300%	0.640%	1.150%	1.340%
656	4.650%	0.480%	10.170%	0.390%

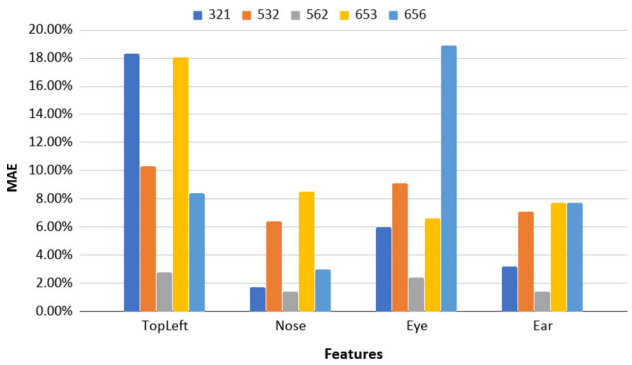


Fig. 7 MAE analysis computed on test images (2) with different features of the same size. The characteristics of the features considered for these tests are shown in Table (5). For each graph, on the x-axis are reported the features (e.g., Top-left, Nose, Eye, Ear), instead on the y-axis the MAE percentage. At the same time, the chart refers to the linear programming model described in Sect. 2.1, and the results of the integer linear programming model shown in Sect. 2.2 are not reported because it is able to identify the filter optimally with a MAE equal to 0 in all the cases

Table 7 Results of test of computing filters with different ranges of value: $[-2, 2]$, $[-4, 4]$, $[-6, 6]$, $[-8, 8]$, $[-10, 10]$. The MAEs are computed between the original output image and the one produced by applying the obtained filter. We can notice that the LP model is not range dependent, while the ILP model is strongly dependent. Let $Max(f)$ be the element of f with the maximum absolute value, the more δ is near to $Max(f)$, the less is the error committed by the model

Range	f Range	LP Model		ILP Model	
		MAE	Time	MAE	Time
$[-2, 2]$	562 $[-4, 1]$	2.80%	0.00	0.71%	9.48
	532 $[-2, 6]$	10.32%	0.00	2.50%	14.06
	701 $[-7, 1]$	3.75%	0.00	3.54%	46.17
	811 $[-1, 8]$	9.50%	0.00	2.90%	48.38
	911 $[-1, 9]$	6.40%	0.00	5.00%	36.39
$[-4, 4]$	562 $[-4, 1]$	2.80%	0.00	0.00%	0.00
	532 $[-2, 6]$	10.32%	0.00	0.92%	10.88
	701 $[-7, 1]$	2.14%	0.00	1.87%	24.19
	811 $[-1, 8]$	9.15%	0.00	1.75%	24.38
	911 $[-1, 9]$	5.50%	0.00	2.65%	15.77
$[-6, 6]$	562 $[-4, 1]$	2.80%	0.00	0.00%	0.00
	532 $[-2, 6]$	10.32%	0.00	0.00%	0.00
	701 $[-7, 1]$	0.84%	0.00	0.78%	2.78
	811 $[-1, 8]$	9.15%	0.00	1.00%	3.69
	911 $[-1, 9]$	4.92%	0.00	1.65%	12.72
$[-8, 8]$	562 $[-4, 1]$	2.80%	0.00	0.00%	0.00
	532 $[-2, 6]$	10.32%	0.00	0.00%	0.00
	701 $[-7, 1]$	0.00%	0.00	0.00%	0.00
	811 $[-1, 8]$	9.15%	0.00	0.00%	0.00
	911 $[-1, 9]$	4.92%	0.00	0.70%	1.56
$[-10, 10]$	562 $[-4, 1]$	2.80%	0.00	0.00%	0.00
	532 $[-2, 6]$	10.32%	0.00	0.00%	0.00
	701 $[-7, 1]$	0.00%	0.00	0.00%	0.00
	811 $[-1, 8]$	9.15%	0.00	0.00%	0.00
	911 $[-1, 9]$	4.92%	0.00	0.00%	0.00

$$f_{sharpen7x7} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & -1 & -1 & -1 & -1 & 0 \\ 0 & -1 & 2 & 2 & 2 & -1 & 0 \\ 0 & -1 & 2 & 1 & 2 & -1 & 0 \\ 0 & -1 & 2 & 2 & 2 & -1 & 0 \\ 0 & -1 & -1 & -1 & -1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Table 8 MAE obtained comparing the figures produced applying the original filters to the test image and the figures produced applying the filters generated using the LP (2.1) and ILP (2.2) models to the test image. The original filters are 5×5, while the produced filters are 3×3, 5×5 and 7×7. When the size of the filter to compute is smaller than the size of the original one, ILP reaches the time limit. Nevertheless, the computed filters are able to produce images with $MAE \leq 12\%$. Instead, when the size of the filter to compute is greater than or equal to the original one, ILP is able to identify correctly the applied filter

Size		LP	ILP	$TimeILP$
3×3	$f_{sharpen}$	16.50%	12.21%	1800.00
	$f_{verticaledges}$	5.40%	1.70%	1800.00
	$f_{emboss5x5}$	26.40%	7.75%	1800.00
5×5	$f_{sharpen}$	12.38%	0.00%	1.42
	$f_{verticaledges}$	3.50%	0.00%	0.33
	$f_{emboss5x5}$	18.00%	0.00%	0.31
7×7	$f_{sharpen}$	12.80%	0.00%	2.24
	$f_{verticaledges}$	5.00%	0.00%	0.36
	$f_{emboss5x5}$	22.50%	0.00%	0.38

4 Conclusion

In this work, we faced the filter retrieval problem, in which we want to identify the filter applied on an input image given the output image. We proposed three mathematical formulations: Two of them are related to the deconvolution process, one to the thresholding instead. We formulated a linear programming model (LP) and an integer linear programming (ILP) model for the deconvolution filters. Several tests were performed to prove the validity and the performance of the models. It is possible to notice from the results that our formulations, in particular the ILP model, are able to identify several filters applied on an input image, even if the output image is disrupted by noise, with just a mere 20x20 sample of the total image. The output images produced by the computed filters have low MAE considering the original one as oracle. Also, our models are able to enhance features with just a single image in which the feature is manually highlighted. Finally, suppose our approach is not able to produce a reliable filter capable to emphasize the desired feature. In that case, we can affirm that is not possible to enhance the feature through filters. A significant feature of the filters is their low computational complexity; this makes them usable in many contexts where the computing power is minimal. Thanks to the work done in recent years on numerous papers related to the identification of drone routes (Carrabs et al.

2017a, Carrabs et al. 2017b, 2020), an exciting direction for this research could be to improve the navigation algorithms by integrating obstacle recognition systems currently widely used for drones, with suitable filters. This approach could lead to usable energy savings to improve the routes produced. In particular, drones use the images captured by video cameras for multiple purposes, particularly for positioning and recognizing obstacles and targets. The processing connected to the analysis of these images requires an energy consumption that otherwise would have been exploited to extend the flight time. The use of filters for some of these tasks could lead to impressive energy savings. Given that for many complex applications (e.g., object detection), a single convolution kernel, whatever its size, is not enough; several approaches in literature tried to find the best combination of different filters to reproduce the desired effect (Coyle and Lin 1988, Coyle et al. 1989, Gabbouj and Coyle 1991). For this reason, a possible future improvement for this work, considering its effectiveness in the filter retrieval in simple cases, is its usage, in combination with other techniques such as genetic algorithm (Cerrone et al. 2016), neural networks (Abiodun et al. 2018), heuristics (Cerrone et al. 2017) and meta-heuristic Carrabs et al. 2014 approaches, to reproduce more complex scenarios such as the creation of filter sequences to be applied to the image to enhance specific characteristics. In recent years, approaches based on neural networks have benefited enormously from the use of filters; these networks using convolutional filters are able to process images very effectively (Khan et al. 2020). An exciting direction for this research could be to develop filters suitable for joint use with neural networks.

Funding Open access funding provided by Università degli Studi di Genova within the CRUI-CARE Agreement.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Human and animal rights This article does not contain any studies with human participants or animals performed by any of the authors.

A Filters

The majority of the filters produced in our experimentation are 3x3. For simplicity, we report them as a one row vector.

For example

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

is reported as [1, 0, -1, 2, 0, -2, 1, 0, -1].

f_{sobel}	1.000, 0.000, −1.000, 2.000, 0.000, −2.000, 1.000, 0.000, −1.000
$f_{laplace}$	0.000, 1.000, 0.000, 1.000, −4.000, 1.000, 0.000, 1.000, 0.000
f_{blur}	0.111, 0.111, 0.111, 0.111, 0.111, 0.111, 0.111, 0.111, 0.111
$f_{prewitt}$	1.000, 1.000, 1.000, 0.000, 0.000, 0.000, −1.000, −1.000, −1.000
f_{emboss}	−2.000, −1.000, 0.000, −1.000, 1.000, 1.000, 0.000, 1.000, 2.000
f_{321}	1.000, 2.000, 1.000, 0.000, 0.000, 0.000, −1.000, −2.000, −1.000
f_{532}	−2.000, −2.000, 0.000, −2.000, 6.000, 0.000, 0.000, 0.000, 0.000
f_{562}	0.000, 1.000, 0.000, 1.000, −4.000, 1.000, 0.000, 1.000, 0.000
f_{653}	−1.000, −2.000, −1.000, 0.000, 0.000, 0.000, 1.000, 2.000, 1.000
f_{656}	−1.000, 0.000, 1.000, −2.000, 0.000, 2.000, −1.000, 0.000, 1.000
f_{701}	1.000, 1.000, 1.000, 1.000, −7.000, 1.000, 1.000, 1.000, 1.000
f_{811}	−1.000, −1.000, −1.000, −1.000, 8.000, −1.000, −1.000, −1.000, −1.000
f_{911}	−1.000, −1.000, −1.000, −1.000, 9.000, −1.000, −1.000, −1.000, −1.000

The remaining filters are 5x5 and reported as follows:

$$f_{sharpen} = \begin{bmatrix} -1 & -1 & -1 & -1 & -1 \\ -1 & 2 & 2 & 2 & -1 \\ -1 & 2 & 1 & 2 & -1 \\ -1 & 2 & 2 & 2 & -1 \\ -1 & -1 & -1 & -1 & -1 \end{bmatrix}$$

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

f_1	0.347, −0.014, −0.333, 2.576, 0.011, −2.587, 0.267, 0.001, −0.268
f_2	0.014, 0.893, 0.000, −0.024, −2.720, 0.836, 0.092, 0.744, 0.164
f_3	0.153, 0.041, 0.136, 0.066, 0.099, 0.110, 0.111, 0.137, 0.106
f_4	0.543, 1.469, 0.210, −0.127, −1.974, 0.563, −0.416, 0.504, −0.773
f_5	−1.416, −0.883, 0.006, −0.883, 1.437, 0.927, 0.006, 0.927, 0.880
$f'pl_{321}$	1.203, 0.727, 0.593, −0.063, −1.342, −2.213, 0.850, 0.854, −0.024
$f'pl_{532}$	−1.298, 0.087, 0.084, −0.390, 0.926, 0.208, 0.052, 0.580, 0.098
$f'pl_{562}$	0.148, 0.436, 0.166, 0.231, −1.938, 0.290, 0.218, 0.421, 0.131
$f'pl_{653}$	−0.352, −1.372, −0.122, −0.058, −0.217, −0.047, 0.845, 0.949, 0.689
$f'pl_{656}$	0.026, 1.085, 1.508, −2.071, −1.014, −0.737, 0.244, 0.126, 1.413
$f'pl_{sobel}$	0.648, 0.628, 0.878, −0.058, −0.217, −0.047, −0.155, −1.051, −0.311
$f'pl_{laplace}$	0.148, 0.436, 0.166, 0.231, −1.938, 0.290, 0.218, 0.421, 0.131
$f'pl_{blur}$	0.107, 0.110, 0.106, 0.118, 0.114, 0.117, 0.107, 0.108, 0.109
$f'pl_{prewitt}$	1.068, 0.515, 0.613, −0.365, −0.778, −2.085, 0.860, 0.434, 0.176
$f'pl_{emboss}$	−1.576, −0.011, −0.047, −0.280, 0.627, 0.283, 0.085, −0.085, 2.067
$f''pl_{321}$	0.813, −0.016, −0.732, 1.632, −0.001, −1.572, 0.691, 0.022, −0.828

$f''pl_{532}$	−1.005, −0.796, −0.111, −0.601, 2.562, −0.186, −0.055 0.210, 0.011
$f''pl_{562}$	-0.095, −0.519, 0.018, −0.369, 1.864, −0.532, 0.113, −0.525, 0.064
$f''pl_{653}$	−0.187, −0.016, 0.268, −0.368, −0.001, 0.428, −0.309, 0.022, 0.172
$f''pl_{656}$	−0.588, −0.905, −0.424, 0.107, 0.014, 0.022, 0.450, 0.812, 0.530
$f''pl_{sobel}$	0.412, 1.095, 0.576, 0.107, 0.014, 0.022, −0.550, −1.188, −0.470
$f''pl_{laplace}$	−0.095, −0.519, 0.018, −0.369, 1.864, −0.532, 0.113, −0.525, 0.064
$f''pl_{blur}$	0.096, 0.119, 0.119, 0.119, 0.104, 0.110, 0.115, 0.100, 0.115
$f''pl_{prewitt}$	0.815, 0.020, −0.718, 0.729, 0.002, −0.766, 0.687, 0.035, −0.797

$$f_{emboss5x5} = \begin{bmatrix} -1 & -1 & -1 & -1 & 0 \\ -1 & -1 & -1 & 0 & 1 \\ -1 & -1 & 0 & 1 & 1 \\ -1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

References

Abiodun Oludare Isaac, Jantan Aman, Omolara Abiodun Esther, Dada Kemi Victoria, Mohamed Nachaat AbdElatif, Arshad Humaira (2018) State-of-the-art in artificial neural network applications: A survey. Heliyon 4(11):e00938

Alpaydin Ethem (2009) Introduction to machine learning. MIT press, Cambridge

Arora Siddharth, Acharya Jayadev, Verma Amit, Panigrahi Prasanta K (2008) Multilevel thresholding for image segmentation through a fast statistical recursive algorithm. Pattern Recognit Lett 29(2):119–125

Behera Bichitrananda, Kumaravelan G et al (2019) Performance evaluation of deep learning algorithms in biomedical document classification. In 2019 11th International Conference on Advanced Computing (ICoAC), pages 220–224. IEEE

Bovik Alan C (2010) Handbook of image and video processing. Academic press, Cambridge

Burger Harold C, Schuler Christian J, Harmeling Stefan (2012) Image denoising: Can plain neural networks compete with bm3d? In 2012 IEEE conference on computer vision and pattern recognition, pages 2392–2399. IEEE

Carrabs Francesco, Cerrone Carmine, Cerulli Raffaele (2014) A tabu search approach for the circle packing problem. In 2014 17th International conference on network-based information systems, pages 165–171. IEEE

Carrabs Francesco, Cerrone Carmine, Cerulli Raffaele, D'Ambrosio Ciriaco (2017a) Improved upper and lower bounds for the close enough traveling salesman problem. In International Conference on Green, Pervasive, and Cloud Computing, pages 165–177. Springer

Carrabs Francesco, Cerrone Carmine, Cerulli Raffaele, Gaudioso Manlio (2017b) A novel discretization scheme for the close enough traveling salesman problem. Comput Operat Res 78:163–171

Carrabs Francesco, Cerrone Carmine, Cerulli Raffaele, Golden Bruce (2020) An adaptive heuristic approach to compute upper and lower bounds for the close-enough traveling salesman problem. INF J Comput 32(4):1030–1048

Castro Wilson, Yoshida Hideaki, Gil Lucía Seguí, López Luis Mayor, Oblitas Jimy, De-la Torre Miguel, Avila-George Himer (2019) Microstructural analysis in foods of vegetal origin: an approach with convolutional neural networks. In 2019 8th International Conference On Software Process Improvement (CIMPS), pages 1–5. IEEE

Cerrone Carmine, Cerulli Raffaele, Gaudioso Manlio (2016) Omega one multi ethnic genetic approach. Optim Lett 10(2):309–324

Cerrone Carmine, Cerulli Raffaele, Golden Bruce (2017) Carousel greedy: a generalized greedy algorithm with applications in optimization. Comput Oper Res 85:97–112

Chen Yangkang, Fomel Sergey (2015) Random noise attenuation using local signal-and-noise orthogonalization. Geophysics 80(6):WD1–WD9

Chottera A, Jullien G (1982a) Design of two-dimensional recursive digital filters using linear programming. IEEE Trans Circuit Syst 29(12):817–826

Chottera A, Jullien G (1982b) A linear programming approach to recursive digital filter design with linear phase. IEEE Trans Circuit Syst 29(3):139–149

Coyle Edward J, Lin J-H (1988) Stack filters and the mean absolute error criterion. IEEE Trans Acoust, Speech, Signal Process 36(8):1244–1254

Coyle Edward J, Lin J-H, Gabbouj Moncef (1989) Optimal stack filtering and the estimation and structural approaches to image processing. IEEE Trans Acoust, Speech, Signal Process 37(12):2037–2066

Dellamonica Domingos, Silva Paulo JS, Humes Carlos, Hirata Nina ST, Barrera Junior (2007) An exact algorithm for optimal mae stack filter design. IEEE Trans Image Process 16(2):453–462

Dhawan Atam Prakash, Rangayyan Rangaraj M, Gordon Richard (1985) Image restoration by wiener deconvolution in limited-view computed tomography. Appl optic 24(23):4013–4020

Afreen Farzana U, Abirami S, Srivani M (2019) A framework for captioning the human interactions. In 2019 11th International Conference on Advanced Computing (ICoAC), pages 13–17. IEEE

Gabbouj Moncef, Coyle Edward J (1991) On the lp which finds a mmae stack filter. IEEE Trans Signal Process 39(11):2419–2424

Harikrishna Pillutla, Amuthan A (2020) Sdn-based ddos attack mitigation scheme using convolution recursively enhanced self organizing maps. Sadhana, 45(1)

Harvey Neal R, Marshall Stephen (1996) The use of genetic algorithms in morphological filter design. Signal Process: Image Commun 8(1):55–71

He Kaiming, Zhang Xiangyu, Ren Shaoqing, Sun Jian (2016) Deep residual learning for image recognition. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2016

Jaiswal Tushar, Rajesh Siddavatam (2009) Image noise cancellation by lifting filter using second generation wavelets (lfsgw). In 2009 International Conference on Advances in Recent Technologies in Communication and Computing, pages 667–671. IEEE

Khan Asifullah, Sohail Anabia, Zahoora Umme, Qureshi Aqsa Saeed (2020) A survey of the recent architectures of deep convolutional neural networks. Artif Intel Rev 53(8):5455–5516

Asem Khmag, Abd Rahman Ramli, Al-Haddad SAR, Hashim SJ, Mohd Noh Zarina, Najih Abdulmawla AM (2015) Design of natural image denoising filter based on second-generation wavelet transformation and principle component analysis. J Med Imag Health Inf 5(6):1261–1266

Maria G, Fahmy M (1974) An l p design technique for two-dimensional digital recursive filters. IEEE Trans Acoust, Speech, Signal Process 22(1):15–21

Michailovich Oleg, Tannenbaum Allen (2007) Blind deconvolution of medical ultrasound images: A parametric inverse filtering approach. IEEE Trans Image Process 16(12):3005–3019

Nussbaumer Henri J (2012) Fast Fourier transform and convolution algorithms, vol 2. Springer Science & Business Media, Berlin

Park Youngtae (2001) Shape-resolving local thresholding for object detection. Pattern Recognit Lett 22(8):883–890

Pedrino Emerson Carlos, Roda Valentin Obac, Kato Edilson Reis Rodrigues, Saito José Hiroki, Tronco Mário Luiz, Tsunaki Roberto H, Morandin Orides Jr, Nicoletti Maria C (2013) A genetic programming based system for the automatic construction of image filters. Integr Comput-Aided Eng 20(3):275–287

Peng Chao, Zhang Xiangyu, Yu Gang, Luo Guiming, Sun Jian (2017) Large kernel matters – improve semantic segmentation by global convolutional network. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2017

Sadek Rowayda A (2012) Svd based image processing applications: state of the art, contributions and research challenges. arXiv preprint arXiv:1211.7102

Sathya PD, Kayalvizhi R (2011) Modified bacterial foraging algorithm based multilevel thresholding for image segmentation. Eng Appl Artif Intel 24(4):595–615

Shrivastava Mayank (2019) Fire severity measures and engineering demand parameters for probabilistic structural fire engineering

Simonyan Karen, Zisserman Andrew (2014) Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556,2014

Szegedy Christian, Liu Wei, Jia Yangqing, Sermanet Pierre, Reed Scott, Anguelov Dragomir, Erhan Dumitru, Vanhoucke Vincent, Rabinovich Andrew (2015) Going deeper with convolutions. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2015

Torres Pedro, Colton Simon, Rüger Stefan (2008) Experiments in example-based image filter retrieval. In Proceedings of the cross-media workshop

Venkatappareddy P, Lall Brejesh, Jayanth C, Dinesh K, Deepthi M (2017) Novel methods for implementation of efficient median filter. In 2017 14th IEEE India Council International Conference (INDICON), pages 1–5. IEEE

Wang Qinxia, Xie Shanghong, Wang Yuanjia, Zeng Donglin (2020) Survival-convolution models for predicting covid-19 cases and assessing effects of mitigation strategies. medRxiv

Wang Wei, Li Jianwei, Huang Feifei, Feng Hailiang (2008) Design and implementation of log-gabor filter in fingerprint image enhancement. Pattern Recognit Lett 29(3):301–308

Weisstein Eric W (2014) Convolution theorem. From MathWorld-A Wolfram Web Resource, 2006a. URL http://mathworld.wolfram.com/ConvolutionTheorem.html

Wu Yang, Zhang Yin, Zhang Yongchao, Huang Yulin, Yang Jianyu (2017) Tsvd with least squares optimization for scanning radar angular super-resolution. In 2017 IEEE Radar Conference (RadarConf), pages 1450–1454. IEEE

Chengqi Xu, Aissaoui Idriss, Jacquey Serge (1994) Algebraic analysis of the van cittert iterative method of deconvolution with a general relaxation factor. JOSA A 11(11):2804–2808

Zhao Fangwei, Desilva Christopher JS (1998) Use of the laplacian of gaussian operator in prostate ultrasound image processing. In Proceedings of the 20th Annual International Conference of the IEEE Engineering in Medicine and Biology Society. Vol. 20 Biomedical Engineering Towards the Year 2000 and Beyond (Cat. No. 98CH36286), volume 2, pages 812–815. IEEE