

Personalized Image Enhancement Using Neural Spline Color Transforms

Simone Bianco¹, Member, IEEE, Claudio Cusano², Member, IEEE, Flavio Piccoli¹,
and Raimondo Schettini¹, Member, IEEE

Abstract—In this work we present *SpliNet*, a novel CNN-based method that estimates a global color transform for the enhancement of raw images. The method is designed to improve the perceived quality of the images by reproducing the ability of an expert in the field of photo editing. The transformation applied to the input image is found by a convolutional neural network specifically trained for this purpose. More precisely, the network takes as input a raw image and produces as output one set of control points for each of the three color channels. Then, the control points are interpolated with natural cubic splines and the resulting functions are globally applied to the values of the input pixels to produce the output image. Experimental results compare favorably against recent methods in the state of the art on the MIT-Adobe FiveK dataset. Furthermore, we also propose an extension of the *SpliNet* in which a single neural network is used to model the style of multiple reference retouchers by embedding them into a user space. The style of new users can be reproduced without retraining the network, after a quick modeling stage in which they are positioned in the user space on the basis of their preferences on a very small set of retouched images.

Index Terms—Image enhancement, deep learning, data-driven methods, convolutional neural networks.

I. INTRODUCTION

TAKING good shots is often not sufficient to obtain beautiful photographs. Many cameras have embedded processing pipelines that, taken as input the raw image, produce results that are judged satisfactory by most amateur photographers. However, these pipelines often fail to meet the needs of professional photographers who are therefore called to manually steer a processing stage where most operations are applied interactively, with parameters that have to be manually tuned. Manual processing can be very time consuming, especially with the large volume of photographs produced nowadays since to obtain high-quality photographs requires

specific technical skills and developed artistic sense. These considerations motivate the need for an automatic retouching system that can rival with the ability of an expert human retoucher.

The study of image enhancement algorithms has a long tradition in the field of image processing and a large variety of methods have been proposed [1]. Recently, the problem has been tackled as an application of machine learning. In its conceptually simplest form, the problem is formulated as an *image-to-image translation* [2] between the raw input images and their versions post-processed by expert photographers. This approach led to remarkable results, but it is not computationally feasible to apply it to the high-resolution images produced by modern photo cameras. Image-to-image translation systems usually run on low-resolution thumbnails. Since these systems need to learn to reproduce the whole image content from scratch, they often produce photographs with some visible artifacts or some loss of the original details.

As noticed by Gharbi *et al.* the limitations of image-to-image translation can be avoided if, instead of estimating pixelwise the whole retouched image, only the transformation between raw and retouched images is computed [3]. In this way it is possible to perform the estimation on a low-resolution thumbnail and to apply the resulting transformation on the original high-resolution photograph. Moreover, fine image details can be preserved and image artifacts can be avoided by just restricting the transformation to belong to a suitable family of transformations.

In this work we follow the latter approach. We propose *SpliNet*, a method based on a convolutional neural network (CNN) which enhances raw images by estimating and applying global image transformations. The transformations considered here are channel-by-channel transfer functions that resemble those used by the “color curves” feature provided by most photo-editing programs. Color curves are a powerful tool that allows expert retouchers to obtain a variety of effects including brightness and contrast adjustment, color balancing, gamma correction, and many others [4]. The curves are typically defined as the interpolation of a set of reference points that the retoucher places on a plot. Once defined, the curves are globally applied to the pixels of the input image. The curves produced by our neural network are natural cubic splines. We chose splines because they can effectively approximate arbitrarily complex functions. Moreover, splines are easy to interpret and they would make it possible to design

Manuscript received October 7, 2019; revised February 28, 2020; accepted April 6, 2020. Date of current version May 8, 2020. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Ajmal S. Mian. (Corresponding author: Simone Bianco.)

Simone Bianco, Flavio Piccoli, and Raimondo Schettini are with the Department of Informatics, Systems, and Communications, University of Milano-Bicocca, 10126 Milano, Italy (e-mail: simone.bianco@unimib.it; flavio.piccoli@unimib.it; raimondo.schettini@unimib.it).

Claudio Cusano is with the Department of Electrical, Computer, and Biomedical Engineering, University of Pavia, 27100 Pavia, Italy (e-mail: claudio.cusano@unipv.it).

This article has supplementary downloadable material available at <http://ieeexplore.ieee.org>, provided by the authors.

Digital Object Identifier 10.1109/TIP.2020.2989584

1057-7149 © 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.

See <https://www.ieee.org/publications/rights/index.html> for more information.

a semi-automatic system, where the automatically estimated color curves could be manually refined by the photographer.

For an automated photo editing system, being able to learn to reproduce the retouching style of a single photographer often is not enough. In fact, there is not a single optimal way of retouching photographs. Each person may prefer a different level of saturation, brightness, contrast, etc. and these preferences may vary with the semantic content of the images. On the basis of these considerations, we extended the neural network to make it able to reproduce the different styles of the users with a single model. As a further extension, we devised two different strategies that allow, without retraining the network, to model new users on the basis of their retouching preferences on a very small set of images.

To verify the effectiveness of our approach we used the MIT-Adobe FiveK dataset [5]. This dataset consists of 5000 photographs, each one retouched by five photo-editing experts. We extended the dataset by including the edited versions of five additional amateur photographers. The results we obtained demonstrate that our method allows to accurately model both single and multiple users, even when their images are not used to train the neural network.

In short, the contributions of the paper include:

- an end-to-end trainable deep learning model including spline-based image processing;
- an image enhancement method that accurately reproduces the retouching style of an expert photographer;
- a method that allows to jointly model multiple users;
- two strategies for the quick modeling of new users not seen during training.

II. RELATED WORK

Image enhancement is a classic problem in image processing which consists in altering an input image to improve its perceived quality. The literature on image enhancement is huge and variegated. Broadly, it presents three main classes of methods:

- *heuristic- or model-based* methods, that automatically enhance the input images by making them adhere to some heuristics [6], [7], or use models of the lightness and color perception of human vision [8], [9];
- *interactive* methods, that exploit the interaction with the user to help him in modifying the image in order to improve it [10], [11];
- *learning-based* methods, that learn a model of the user's preference and style on a training set of reference images to automatically enhance unseen images [3], [12].

In this work we focus on learning-based enhancement which, over the other approaches, has the advantage of being applicable in a broad set of scenarios including those where the users lack any skill in photo editing or image processing.

Methods of this kind can be further divided in two categories, namely the *paired* and the *unpaired* ones. The former category uses a set of input/enhanced image pairs and learns how to enhance unseen images in a similar way. Methods in the latter category, instead, use two unrelated sets of low quality and enhanced images. Learning how to enhance an unseen image in this framework is clearly more difficult, since

it requires to abstract and model the concept of what is an *enhanced image* and how it can be obtained.

Since the semantic content can significantly influence the type of enhancement to be performed on a given image, some methods that we call *semantic aware* exploit specialized detectors to automatically extract additional information from the images, such as semantic segmentation [13], saliency maps [14], illumination maps [15].

Gharbi *et al.* presented a supervised method for image enhancement that splits the workload in a low-resolution stream and a high-resolution stream [3]. The low-resolution stream, which carries the highest payload, is composed of a CNN with two terminal branches that respectively estimate local and global features. The outputs of the two branches are later combined together. In turn, the high-resolution stream learns a guidance map used to upsample the features inferred in the low-resolution stream and finally uses the resulting values as parameters of a per-pixel transformation that is applied to the full resolution input image.

Yan *et al.* introduced an image descriptor that accounts for the local semantics of the input image [16]. This method applies a color transformation to each pixel of the input image. Specifically, the color transformation used in this case is a polynomial of degree two, therefore each input pixel must be firstly projected into a monomial basis containing the ten terms of the polynomial expansion. To improve accuracy from a perceptual point of view, the input image is firstly converted in the *CIE Lab* color space. Wang *et al.* addressed the problem of enhancing underexposed images [17]. Their method works by estimating an illumination map that is used to modify the contrast of input images.

Bianco *et al.* relax the constraint of inferring a per-pixel transform by capturing local and global variations with a set of patches [18]. Artifacts are prevented by constraining to a smooth transition between adjacent transformations. Here, the color transformation is a polynomial of degree three whose parameters are estimated by a CNN.

Ignatov *et al.* [19] designed a neural approach in which multiple losses are used to train a CNN to improve the quality of images acquired with mobile phones to the level of high-end DSLR cameras. The method proposed by Isola *et al.* uses adversarial training to perform an image-to-image translation [2]. In this case, the output of the neural network is the final enhanced image. The neural network acts as a regressor, and the price to pay to have such a powerful regressor is that the number of parameters is very high and, therefore, it is also high the number of training examples required to make it work properly. Omiya *et al.* leverage existing manual photo enhancement tools as a black-box model to predict the enhancement parameters of that model. Furthermore, to deal with the difficulty of obtaining training data, they proposed to generate supervised training data from high-quality professional images by randomly sampling realistic de-enhancement parameters [20]. One feature that this method has in common with our proposal is that the resulting parameters can be easily interpreted by their users.

The method proposed by Zhu *et al.* given two unpaired sets of images, trains two CNNs to transform images of the first

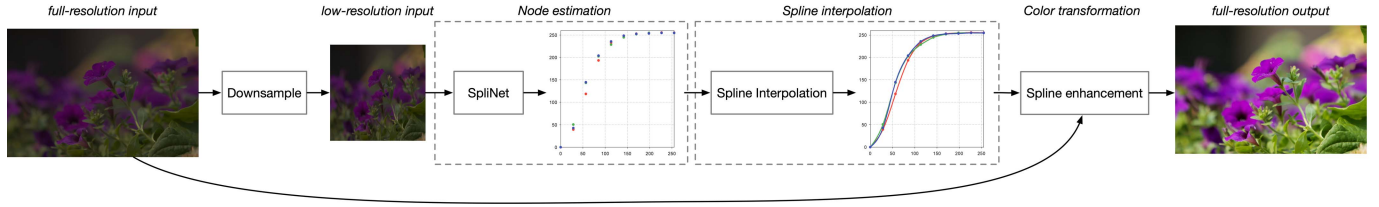


Fig. 1. Processing pipeline of the SpliNet image enhancement method. The input image is resized to 256×256 . Then the nodes of the spline are computed through a convolutional neural network. Finally, for each color channel a natural cubic spline interpolates the nodes found and the resulting color transformation is applied to the pixels of the input image.

type into images of the second, and vice-versa [21]. Cycle redundancy is used to constraint the two transformations and adversarial losses are used to make sure that the concatenation of the two transformations produces consistent results. Another method based on learning from unpaired images is that of Hu *et al.*, in which reinforcement learning is used to learn to apply sequences of simple image manipulations [22]. Reinforcement has also been used by Park *et al.* to train an agent to select the optimal sequence of image enhancement actions [23], and by Yang *et al.* who proposed an iterative method in which the exposure level of a mobile camera is automatically set on the basis of the appearance of the preview image [24].

Caicedo *et al.* make use of collaborative filtering to learn the enhancement style of a new user [25]. During training, users preferences are grouped into clusters. In first place, the new user is assigned to a cluster and the prediction of her/his enhancement is equivalent to the enhancement applied to the most similar training image in the cluster. In this way, distance among images becomes a crucial point and in Caicedo *et al.* is obtained through metric learning.

III. PROPOSED METHOD

In this work we propose a novel CNN-based method that estimates a global color transform for the enhancement of raw images. This method is designed to improve the perceived quality of the images by reproducing the ability of one or more experts in the field of photo editing. The transformation that is applied to the input image is found by a CNN that has been specifically trained for this purpose. More precisely, the network takes as input a raw image and produces as output three sets of *control points* (also called *nodes*), one set for each color channel. Then, the control points are interpolated and the resulting functions are globally applied to the values of the input pixels.

As interpolating functions we chose to use natural cubic splines, since they can approximate arbitrarily complex functions (provided that enough nodes are specified) while remaining as smooth as possible (among all interpolating functions natural splines are those with minimal average curvature). The smoothness of splines introduces an implicit form of regularization that prevents the abrupt changes and the artifacts that can be often observed for methods based on look-up tables or on image-to-image translation. An important property of this approach is that, being both smooth and global, the transformation preserves the content of the input images. Moreover, as we will show, spline interpolation can be computed very

effectively within a deep learning framework, allowing for a rapid training of the neural network.

The proposed network architecture is end-to-end trainable. This is possible because splines are continuous and differentiable functions. To speed up the computation and to make the method suitable for real-time processing, the estimation of the nodes of the spline is calculated on a downsampled version of the input image of size 256×256 , and the enhancement is applied to the original image at full resolution.

We called the proposed method SpliNet; its overall architecture is shown in Figure 1. We defined three different modes of use:

- single-user mode, in which SpliNet is trained to reproduce the style of a single target user;
- multi-user mode, in which the network is trained with images from multiple users; an additional input makes the network switch among the reproduction of the learned styles;
- new-user mode, in which the network trained from multiple users is adapted to reproduce the retouching style of a new user, not seen during the training.

The next sections describe these modes in detail.

A. Single-User Modeling

The reproduction of the style of a single user is obtained with a convolutional neural network that, given as input the raw image to process, yields as output the nodes of three different splines that are used as color curves. The processing pipeline includes three major steps: the estimation of the nodes, their interpolation with splines and, finally, the color transformation of each color channel with the corresponding spline.

The architecture of the CNN is inspired by the work of Bianco *et al.*, since it provided good performances on a similar problem [18]. More popular and larger architectures would probably be unsuitable for this task, due to the limited amount of training data. The downsampled version of the input image having size 256×256 , is firstly processed by a set of convolutional layers followed by ReLUs. With the exception of the first block, all the activations are normalized through a Batch Normalization layer with a momentum of 0.01. After four of these blocks, we apply an average pooling with size 7×7 to reduce the feature map to a single feature vector. The last part of the architecture includes two fully connected layers separated by a ReLU that projects the activations into the splines node space. As a final step, we add the identity

TABLE I
STRUCTURE OF THE CONVOLUTIONAL NEURAL NETWORK USED TO
ESTIMATE THE COORDINATES OF THE NODES OF THE SPLINES.
 N DENOTES THE NUMBER OF NODES OF THE SPLINE AND
 $H \times W$ THE SIZE OF THE INPUT IMAGE

Stage	Operation	Output size
Pre-processing	Input	$H \times W \times 3$
	Bilinear resizing	$256 \times 256 \times 3$
Conv. Network	Conv. + ReLU	$63 \times 63 \times 8$
	Conv. + ReLU + B.N.	$31 \times 31 \times 16$
	Conv. + ReLU + B.N.	$15 \times 15 \times 32$
	Conv. + ReLU + B.N.	$7 \times 7 \times 64$
	Avg. Pooling	$1 \times 1 \times 64$
	Linear + ReLU	64
	Linear	$3N$
	Reshape	$N \times 3$
Post-processing	Identity Sum	$N \times 3$

function to the output of the last fully connected layer. This last operation is inspired by the success of residual networks [26] and has the effect, when the parameters of the network are close to zero, to produce splines close to the identity function. This allows for a quicker training of the network and for an easier initialization of its parameters.

The output is a vector containing the N nodes of the 3 splines, therefore it's composed by $N \times 3$ elements. The architecture is summarized in Table I.

Each spline is a composition of cubic polynomials whose coefficients can be found by solving a system of linear equations (for more details see Bartels *et al.* [27]). For each color channel we have:

$$\begin{aligned} a_i &= \frac{m_{i+1} - m_i}{6h}, \\ b_i &= \frac{m_i}{2}, \\ c_i &= \frac{y_{i+1} - y_i}{h} - \left(\frac{m_{i+1} + 2m_i}{6} \right) h, \\ d_i &= y_i, \end{aligned} \quad (1)$$

where a_i, b_i, c_i, d_i are the coefficients of the polynomials, h is the distance between adjacent nodes, y_1, \dots, y_n is the output of the neural network, and the components of the vector \mathbf{m} are obtained by solving the following system:

$$\mathbf{A}\mathbf{m} = \frac{6}{h^2}\mathbf{V}\mathbf{y}, \quad (2)$$

defined in terms of the vector $\mathbf{y} = (y_2, y_3, \dots, y_{N-1})^T$ and of the tridiagonal matrices

$$\begin{aligned} \mathbf{A} &= \begin{bmatrix} 4 & 1 & 0 & \cdots & 0 \\ 1 & 4 & 1 & \cdots & 0 \\ 0 & 1 & 4 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 4 \end{bmatrix}, \\ \mathbf{V} &= \begin{bmatrix} -2 & 1 & 0 & \cdots & 0 \\ 1 & -2 & 1 & \cdots & 0 \\ 0 & 1 & -2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & -2 \end{bmatrix}, \end{aligned} \quad (3)$$

where the equations (1) and (2) and the coefficients in (3) are obtained by requiring that the polynomials pass through the coordinates (x_i, y_i) and that the resulting function is continuous and with continuous first and second derivatives.

Once the nodes have been estimated on a downsampled version of the input image and the three splines have been interpolated, the color transformation is applied to the RGB components of each pixel of the high-resolution image (we also experimented with other color spaces, without any significant improvement).

By design the method preserves the content of the image and cannot corrupt its local structure. Therefore a per point loss can be used to train the neural network. In this work we used the average ΔE_{76} difference between the output image and the same image retouched by an expert photographer.

Note that all operations are differentiable, and that they can be easily parallelized. Therefore the algorithm is suitable for an efficient implementation capable of running on the modern GPUs commonly used for deep learning applications.

B. Multi-User Modeling

For a completely automatic system, being able to accurately reproduce the retouching abilities of a professional photographer is certainly a remarkable result. However, in practice it doesn't matter how accurate the reproduction and how good the photographer are, such a system would leave many users unsatisfied. In fact retouching is not only a technical issue, but it is also a form of art, where subjective aesthetic judgments can be of cardinal importance. A very colorful style could be perceived as stunning by some users and as exaggerated by others; similarly, a conservative style could be judged by different viewers as realistic and natural looking, or as too bland and dull. An ideal system for automatic image enhancement should be able to adapt its style to the taste of the user.

In order to make our system adaptive, we modified the model described in the previous section to support the reproduction of the styles of multiple retouchers. In addition to the image to process, the neural network now takes as input a signature that uniquely represents a user. The signature is transformed in a profile of that user, consisting of a tuple of coordinates in a suitable user space. The profile is then injected in the network that will use it to adjust its output accordingly. More in detail, a linear projection U maps the signature vector \mathbf{s} into the user profile $\mathbf{p} = U\mathbf{s}$ which is then concatenated to the feature vector produced by the convolutional layers of the network, after the spatial average and before the fully-connected layers. The coefficients of the projection matrix U are learned together with the other parameters of the network.

The training process is performed by feeding to the network pairs of images and signatures and by requiring (via the loss function) that the output images match those retouched by the users identified by the signatures. By optimizing U , the training process places users in suitable positions in the user space. For instance, users with a similar retouching style are expected to be located near each other (i.e. they will have similar profiles). A low-dimensional user space forces the

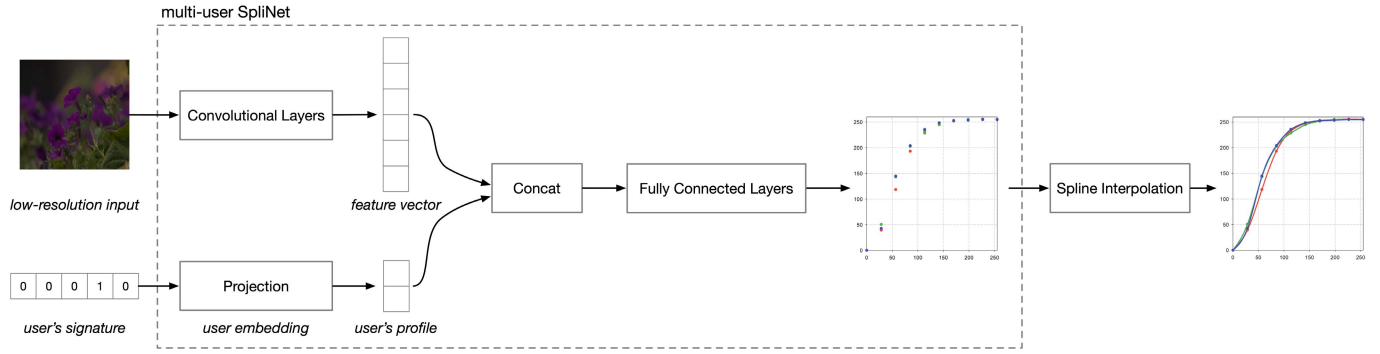


Fig. 2. Multi-user version of SpliNet. The input image is processed by taking into account the identity of the user, provided as a signature, and his preferences, encoded in a numerical profile.

network to model the style of the users by identifying their common preferences and by highlighting their main stylistic traits. A scheme of the multi-user version of the system is reported in Figure 2. The loss used by the multi-user version is the same used by the single-user case.

C. Modeling a New User

Given a new user it would be desirable to be able to adapt the system to his style without asking him to provide a whole new training set of retouched photographs. Instead, it would be very convenient to reproduce his retouching style just by choosing a suitable profile for him in the user space. To this end, we devised two strategies: the *voting strategy* and the *fitting strategy*. The first is targeted to novices, and only requires that they select their preferred images in a sequence of proposals. The second is for more experienced users, since it asks them to retouch a small set of images.

In the voting strategy multiple retouched versions of the same image are shown to the new user who is asked to choose the one he or she likes most. This is repeated multiple times with different images keeping track of the answers. Let K be the number of users on which the multi-user model has been trained and let \mathbf{p}_i be their profiles in the user space $i = 1, \dots, K$. The profile $\mathbf{p}^{(V)}$ for the new user is computed from the number N_i of times in which he chose an image retouched by user i :

$$\mathbf{p}^{(V)} = \frac{\sum_{i=1}^K \mathbf{p}_i N_i}{\sum_{i=1}^K N_i}. \quad (4)$$

In practice the new profile is a convex combination of the known ones, where the coefficients are the fractions of votes expressed by the new user.

In the fitting strategy the new user is given a set of m images I_1, \dots, I_m and is asked to retouch them according to his preferences, producing the images $\hat{I}_1, \dots, \hat{I}_m$. Then his profile $\mathbf{p}^{(F)}$ is chosen by minimizing the average dissimilarity between the output of the network and the retouched images:

$$\mathbf{p}^{(F)} = \arg \min_{\mathbf{p} \in \mathbb{R}^2} \frac{1}{m} \sum_{i=1}^m d(\hat{I}_i, \text{SpliNet}(I_i, \mathbf{p})), \quad (5)$$

where $d(\cdot, \cdot)$ is a dissimilarity measure and $\text{SpliNet}(I_i, \mathbf{p})$ is the output of the method when fed with the input image I_i and

when injected with the user profile \mathbf{p} . To find a solution to this optimization problem we initialize \mathbf{p} with random values and we perform several iterations of the gradient descent algorithm by keeping all the parameters of the network fixed and by updating each time only the profile \mathbf{p} .

IV. EXPERIMENTAL SETUP

Measuring the performance of image enhancement methods is a challenging task. Beside the subjective human judgment, objective assessment procedures are limited by the amount of reference data available. The most common approach consists in measuring the difference between the images produced by the algorithms and the images retouched by human operators (hopefully chosen among experts in the field). The evaluation procedures reported in the literature differ in the sets of images used for training and test, in the performance metrics etc. Since there is not a commonly recognized standard for the evaluation of enhancement methods, we devised our own setup that is described in detail by the following sections.

A. Dataset

The dataset used in this work is the MIT-Adobe FiveK dataset [5] (in short FiveK), that contains 5000 samples. Each sample is composed by a raw image in the *DNG* format and by the same image enhanced by five experts in *16-bit TIFF* format. The dataset is also distributed as a single Adobe Lightroom® catalog, which makes the download easier and gives the possibility to explore the history of the enhancement steps performed by the experts. The dataset has been used in many works but unfortunately each work has exported the images from the catalog in a different, and often non-documented, way. In order to limit the diffusion of multiple, different, copies of the dataset, to convert the *DNG* files in a python-readable format and to allow the comparison of our method with the state of the art, we converted the images using the procedure described by Hu *et al.*, which is publicly documented and available online [22]: starting from the catalog, in the collection list we select the collection *Inputs/Input with Daylight WhiteBalance minus 1.5*; the raw images are exported as *16-bit TIFF* in the *ProPhotoRGB* color space and then converted into the *sRGB* color space as *8-bit*



Fig. 3. A sample image from the FiveK dataset. In addition to the raw image the dataset includes five different versions retouched by five expert photographers (Expert A–E). For 100 images we added five versions retouched by amateur users (User F–J).



Fig. 4. A sample of raw images from the RAISE dataset.

PNG. This last conversion is required to make it possible to correctly interpret the images with common image processing libraries. The experts' images are exported as 8-bit *PNG* in the *sRGB* color space. From the same work by Hu *et al.* we also take the subdivision into a training set of 4000 images and a test set of 1000 images.

To make it possible to explore the modeling of new users, we integrated the FiveK dataset by asking five amateur photographers to retouch 100 images (randomly picked from the 1000 test images) using the “color curves” tool of a popular photo editing software. A sample image from the FiveK dataset and its edited versions are reported in Figure 3, where it can be noticed the different style of the retouchers.

To verify if the proposed method generalizes well to other datasets, whose images played no role in the training of the neural network, we also conducted some experiments on 100 images randomly samples from the RAISE dataset [28]. This dataset has been primarily designed for the evaluation of digital forgery detection algorithms. It consists of 8156 high-resolution raw images taken by four photographers with three different cameras. Before using these images in the experiments we processed them with the same pre-processing pipeline defined by the authors of the FiveK dataset. A sample of the result is reported in Figure 4.

B. Metrics

Following Gharbi *et al.*, the first performance criterion considered is based on the difference in color appearance as perceived by a human observer [3]. To this end, the ground truth and output images are converted in the perceptually uniform color space *CIE Lab*. Then, to measure the difference between two images we used the average of the widely used

ΔE_{76} distance and its more accurate version ΔE_{00} [29]. Close attention should be paid to the conversion step of images into *CIE Lab* color space [30], since using the wrong transformations could easily bias the error calculation.

The third error metric used is the ΔL , that only considers the difference in lightness, thus discarding the color information. This is used to better understand how the lightness and color components contribute to the final ΔE error.


The last metric considered is the structural similarity index (SSIM), that is a perception-based model that considers image degradation as perceived change in structural information [31]. The SSIM is calculated on 11×11 sliding windows to produce a local distortion map, which is then averaged to obtain the final global SSIM index.

The three metrics are used to assess the accuracy in reproducing enhanced images from three different points of view. ΔE and ΔL respectively measure the chromatic and lightness differences taking into account human perception, giving an idea of the color and lightness fidelity. SSIM measures differences in the local structures of images, thus highlighting the presence of structural artifacts.

C. Implementation and Training

SpliNet has been implemented in the Python programming language using the Pytorch deep learning framework [32]. All the operations, including image preprocessing, downsampling, color space conversions, node estimation and spline interpolation have been integrated into an end-to-end deep learning module. Except where differently specified, we set the neural network to estimate ten nodes per spline.

The training procedure consisted of 40 000 iterations of the Adam learning algorithm [33] set to minimize the average ΔE_{76} between the output and the ground truth images. The learning rate was set to 10^{-4} , the weight decay was 0.1 and each mini-batch consisted of 20 images. The whole training procedure requires about 220 minutes of computation on a NVIDIA Titan Xp GPU.

The source code, the trained models and the data are publicly available in the github repository  https://github.com/dros1986/neural_spline_enhancement.

V. EXPERIMENTAL RESULTS

We conducted several experiments to evaluate the performance of the proposed method. We considered three scenarios: (i) modeling of a single user, (ii) modeling of a group of users, (iii) modeling of new users not seen during training. The results we obtained in these three scenarios are reported in the next sections.

A. Single-User Scenario

In the first experiment we assessed how well the proposed method is able to learn to reproduce the retouching style of a single photographer. We trained the single-user model on the 4000 training images retouched by expert C who, due to the consistency of the style, is the one more frequently considered in the state of the art. Performance is evaluated on the 1000 test images retouched by the same expert.

TABLE II

AVERAGE ACCURACY IN REPRODUCING THE 1000 TEST IMAGES RETOUCHE BY EXPERT C (FOR ΔE_{76} , ΔE_{00} AND ΔL THE LOWER THE BETTER, FOR SSIM THE HIGHER THE BETTER)

Method	ΔE_{76}	ΔE_{00}	ΔL	SSIM
Optimal gamma correction	17.05	11.45	9.14	0.793
Exposure [22]	16.98	12.38	9.54	0.872
CycleGAN [21]	16.23	11.96	9.20	0.835
Average splines	14.57	10.40	8.15	0.899
Distort and recover [23]	13.18	9.66	7.44	0.902
Unfiltering [18]	13.17	9.56	6.76	0.922
HDRNet [3]	12.14	9.00	6.27	0.930
Pix2pix [2]	11.13	8.22	5.55	0.915
SpliNet	10.70	7.75	5.52	0.942

In Table II we report the results obtained by our method measured in terms of ΔE_{76} , ΔE_{00} , ΔL , and SSIM. The table also reports the results obtained by recent methods in the state of the art retrained on our version of the training set by using the source code provided by their authors. We also included the errors obtained by a gamma correction where the gamma is chosen for each image with a grid search that minimizes the ΔE_{76} . As a further reference we reported the results obtained by using the average splines (one for each color channel) estimated on the training set by SpliNet trained on expert C images.

The first thing that can be noticed from the results reported in Table II is that the worst results are obtained by the optimal gamma correction; this means that learning just a gamma correction, even if optimal, is not enough to learn to reproduce the retouching style of a photographer; on the other side, this also means that all the considered methods learn a transformation that is more complex than a simple gamma correction. With the average splines we obtained better, but still unsatisfactory, results. This means that the image correction must be adaptive with respect to the content of the images.

From the comparison with the state of the art it can be seen that the proposed method obtains the best results in terms of all the error metrics considered. The second best method, i.e. Pix2Pix, is very close in terms of ΔL lightness error but makes larger ΔE_{76} and ΔE_{00} colorimetric errors. Moreover, thanks to the smoothness and the globality of the spline-based transformation used, the proposed method has the largest SSIM index. This means that our method is the one that introduces the lesser amount of structural artifacts, which is the main drawback of GAN-based methods such as CycleGAN and Pix2Pix. The second best method in terms of SSIM index is HDRNet, that is the third one in terms of colorimetric error.

In Figure 5 we report some sample images from the FiveK and RAISE datasets processed by our method and by the other methods in the comparison. It is possible to notice how Exposure and CycleGAN tend to produce visual results that are very different from the ground truth, while our method, HDRNet and Pix2pix are much closer. This shows how difficult it is to learn from unpaired examples.

In order to show what the typical shape of the splines learned by our method is, some further examples of images processed by our method together with the corresponding learned splines are reported in Figure 6. It is possible to notice

how the transformation is different for each image, and how for some images the splines are almost the same for each color channel, while for others are different.

1) *Subjective Evaluation*: Objective measures do not always capture the effectiveness of an enhancement method. Sometimes an image that is clearly different from the reference one may still look good to human observers. In order to assess the perceived quality of the image enhancement by the proposed method we organized a subjective test. We recruited 11 volunteers and we showed them a subset of 100 images taken from the test set of the FiveK dataset. For each image we presented, in random order, the image enhanced by our method and those obtained with other five methods in the state of the art. The volunteers selected the image they considered as more visually appealing. At the end of the experiment we collected a total of 707 preferences (not all the volunteers completed the test).

For the RAISE dataset subjective preferences represent the only option available for performance evaluation, since there are no ground truth images to compare against the output of the algorithms. The volunteers that participated to the evaluation of the RAISE images were 13, and they collectively expressed 574 preferences. The results of the experiments are summarized in Figure 7. SpliNet was the method that obtained the largest amount of preferences for both datasets (about 32.3% for FiveK and about 39.9% for RAISE). In both cases the difference between SpliNet and the second best is statistically significant, as it passes a one-tailed binomial test with a p -value of 0.005. Over the two datasets HDRNet, Pix2pix and Unfiltering obtained a good amount of preferences. The second best algorithm was HDRNet for the FiveK dataset (27.5%) and Unfiltering for RAISE (21.1%). Pix2pix was generally considered good in choosing the color distribution, but its output was often corrupted by artifacts, that have been evaluated very negatively even when they were small enough to not have a big impact on the performance measures. The two methods following the unpaired training paradigm (Exposure and CycleGAN) and the one using reinforcement learning (Distort and Recover) obtained a significantly smaller amount of preferences.

2) *Color Distribution*: In Figure 8 we report the average histograms in *CIELab* color space for the FiveK images processed with our method and the ground truth images retouched by expert C. Three different histograms are reported, one for each color channel, all computed with bin size equal to five. To better show the differences between the two histograms they are reported using a logarithmic scale. From the plots it is possible to notice how our method does a pretty good job in predicting the lightness channel L , while it tends to produce images with a and b distributions more dense towards zero, thus resulting more conservative in terms of color saturation.

3) *Dependence on the Image Content*: The best enhancement for an image may depends on its semantic content. To identify the type of images that favor our method and those where the worst errors occur, we computed the performance metrics on homogeneous subsets of the test set. To do so we used the categorization provided with the FiveK

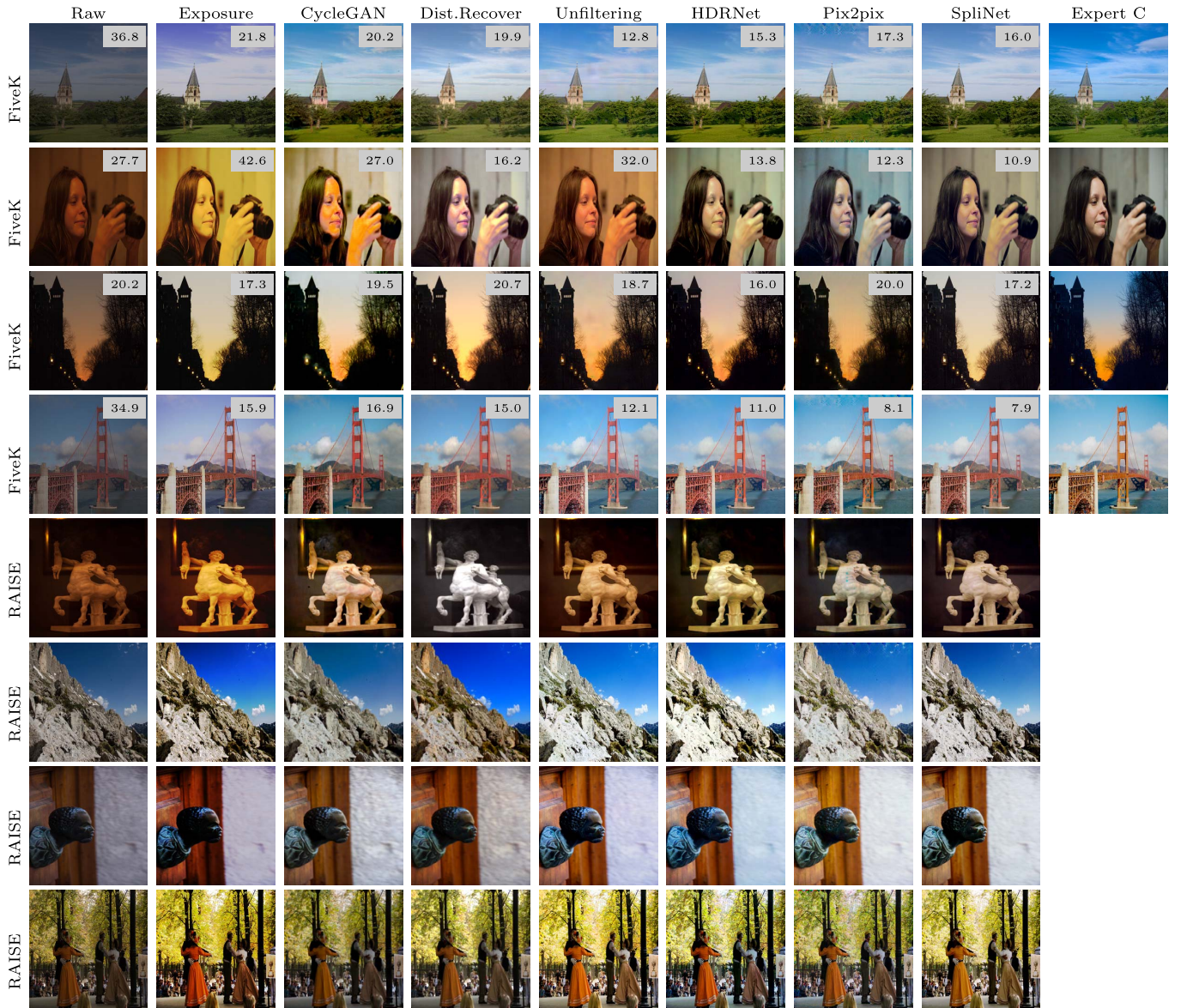


Fig. 5. Examples of test images enhanced by SpliNet and by other methods in the state of the art. The first column contains the input raw images, the last shows the target images retouched by expert C (only for FiveK images) and the remaining columns report the results of the enhancement methods considered. For the FiveK images we report in the upper right corner the ΔE_{76} with respect to the ground truth image retouched by expert C.

dataset which divides images by subject, illumination, time of the day and location. The results in terms of average ΔE_{76} are reported in Figure 9. Low errors are obtained on outdoor images taken by day under a natural illumination. This kind of images usually do not present severe distortions such as strong color casts, exaggerated contrasts, over- or under-exposition, etc. Colors tend to be natural, and their distribution does not require a sophisticated adjustment. Moreover, these images are quite common so that it is relatively easy for the model to learn to reproduce a pleasant dynamic range using easily identifiable elements, such as the sky, as a reference.

The most difficult images to enhance are those with an unusual content (e.g. abstract subject) or imaging conditions (e.g. taken in low light). Beside being inherently more difficult to process, these kind of images have also few occurrences in the training set, making them even harder to enhance for a learning-based method.

4) *Sensitivity Analysis:* We performed a sensitivity analysis of our method with respect to the number of spline nodes to be estimated. We plot in Figure 10 the ΔE_{76} obtained by our method varying the number of spline nodes in the range [5, 20] by steps of five. From the plot it can be seen that using a low number of spline nodes results in the worst performance, and then the performance starts to flatten when using ten nodes. Therefore in the experiments we used ten spline nodes, since it represents the best trade-off between accuracy and model complexity. Increasing that number would make the model more flexible but also harder to train, due to the additional number of parameters.

5) *Processing Time:* One of the advantages of our method is that it allows to quickly process high-resolution images. The more complex operations are carried out on a low-resolution thumbnail and the final application of the splines scales linearly with the number of pixels in the input image.

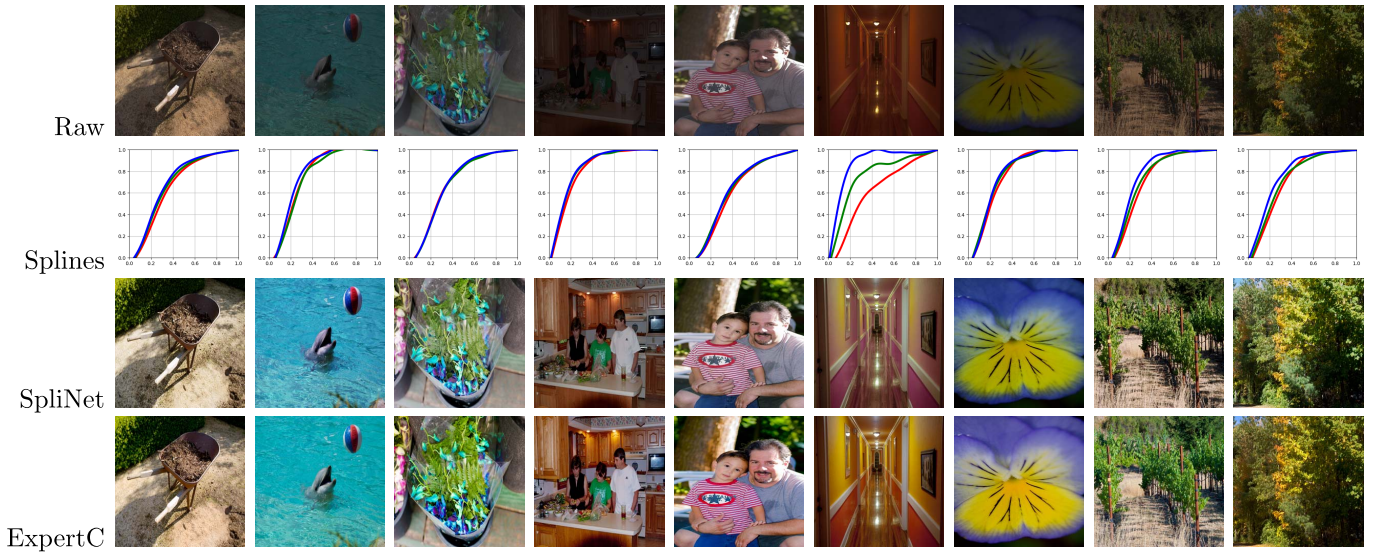


Fig. 6. Examples of images from the FiveK dataset processed by SpliNet. From top to bottom the rows contain: input raw images, inferred splines color curves, SpliNet outputs and the ground-truth images retouched by expert C.

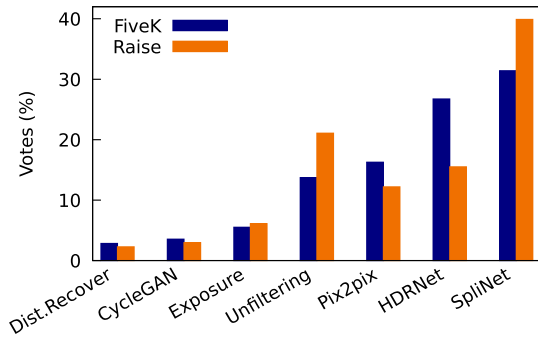


Fig. 7. Distribution of the preferences of the participants of the subjective test over the seven enhancement methods considered.

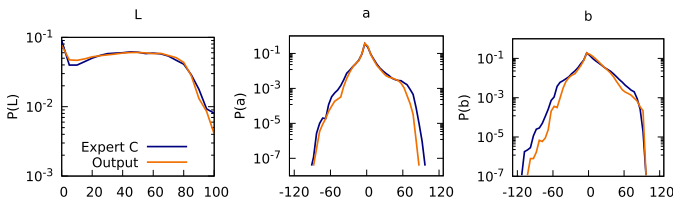


Fig. 8. Comparison of the distributions of the Lab color channels for the whole FiveK test set retouched by expert C and processed by the proposed method.

Figure 11 reports the actual processing time as a function of the size of the input image obtained by executing our method on a computer equipped with a NVIDIA Titan Xp GPU. The plot shows how for small images the processing time grows slowly. In this case, in fact, the most expensive operation is represented by the forward step of the convolutional neural network, which is executed on thumbnails of 256×256 pixels. For larger images, the total time is dominated by the application of splines. Even large images can be processed in reasonable times. For instance, it takes about 185 milliseconds to enhance a 2048×2048 image.

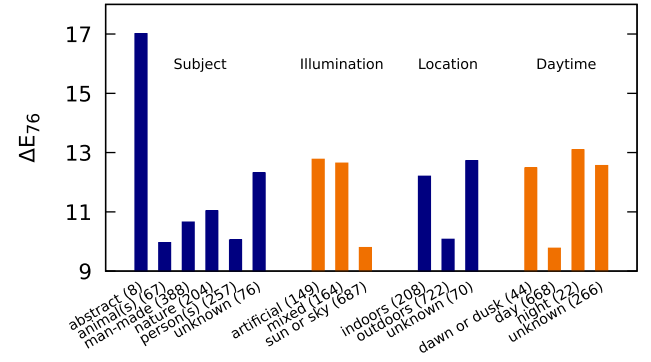


Fig. 9. Detail of the accuracy (average ΔE_{76}) in reproducing the FiveK test images retouched by expert C, measured on different subsets of the test set. Images are divided by subject, type of illumination, location and time of the day. The numbers of images to which a label applies are reported in parenthesis.

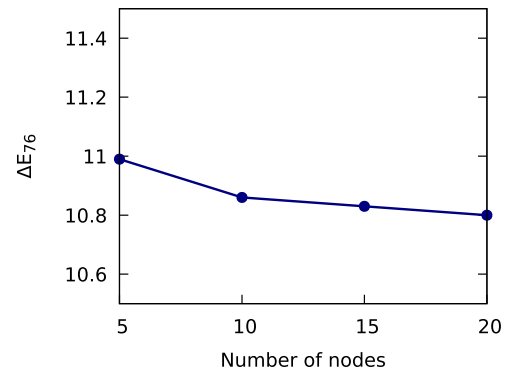


Fig. 10. Errors in reproducing the images retouched by expert C varying the number of nodes defining the splines. The error is averaged over all the test images in the FiveK dataset.

B. Multi-User Scenario

We trained the multi-user variant of SpliNET on the training set of the FiveK dataset by using the 4000 training images

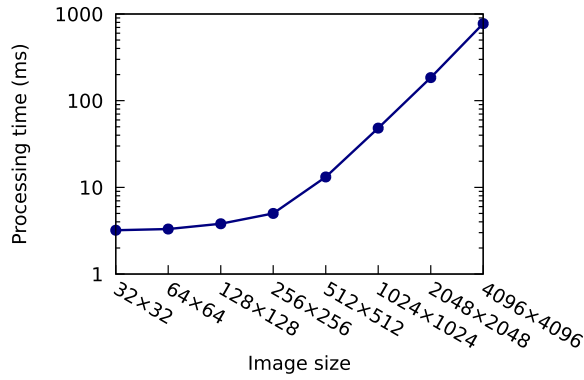


Fig. 11. Processing time (in milliseconds) taken to enhance images of various sizes. Each data point is the average over 100 runs.

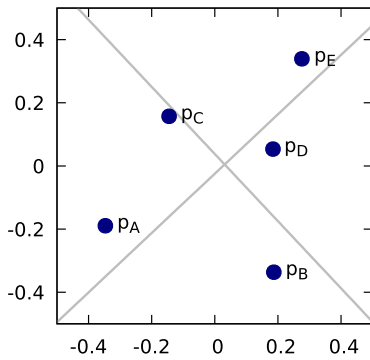


Fig. 12. Graphical representation of the user space. The points represent the profiles of the five experts, while the dashed lines represent the two principal components (the first goes approximately in the p_A-p_E direction, while the second is in the p_B-p_C direction).

retouched by all the five available experts. We used five-dimensional one-hot vectors as signatures (the signature of expert A was $s_A = \langle 1, 0, 0, 0, 0 \rangle$, and so on), and we set the user space to be two-dimensional (to increase its dimensionality we would probably need more experts). The number of training iterations was five times that used to train the single expert version.

As a result of the learning process we obtained, together with the trained model, a definition of the user space. Figure 12 shows the positions in that space of the profiles of the five experts. To better understand the user space we performed a principal component analysis of the five profiles and we observed how moving along the two principal components changes the output of the network. Figure 13 reports the outcome of this experiment on a small sample of test images. It is evident that by moving in the direction of the first principal component it is possible to adjust the brightness of the image. For the second component, instead, we observe images with warmer tones on one side and a prevalence of cold colors on the other. Therefore we can say that the second component captures the chromatic element of the retouching style. By combining these directions (i.e. by moving in the user space) it is possible to obtain a variety of retouching styles including, but not limited to, those of the five experts used during the training.

TABLE III
PERFORMANCE OF THE SINGLE- AND MULTI-USER MODELS MEASURED ON THE TEST IMAGES OF THE FIVEK DATASET

Model	User	ΔE_{76}	ΔE_{00}	ΔL	SSIM
Single-user	Expert A	11.32	8.51	7.40	0.911
	Expert B	9.01	6.81	4.87	0.965
	Expert C	10.70	7.75	5.52	0.942
	Expert D	10.93	8.29	7.03	0.929
	Expert E	10.78	8.01	5.73	0.951
Multi-user	Expert A	10.91	8.18	7.16	0.915
	Expert B	8.69	6.59	4.88	0.965
	Expert C	10.50	7.58	5.61	0.942
	Expert D	10.63	8.07	6.97	0.939
	Expert E	10.33	7.67	5.52	0.951

We evaluated the learned model on the 1000 test images repeating the evaluation five times, each time selecting the profile of a different expert. Table III reports the results obtained. By comparing the performance of the single- and the multi-user models we notice how this latter does not incur in any loss in performance. On the contrary, the multi-user version slightly outperforms the single-user version. This is not too surprising, if we consider that, like in multitask learning [34], the multi-user model is encouraged to learn more robust intermediate features in order to be able to reproduce different styles.

From the computational point of view the difference between the multi- and the single-user versions is negligible: the multi-user model includes just 138 additional parameters: 10 in the projection matrix U and 128 extra coefficients in the first fully-connected layer.

C. New User Scenario

To assess the two modeling strategies (voting and fitting) we performed several simulations in which one of the experts in the FiveK dataset plays the role of the new user, and the other four act as the known users. To do so, we retrained the multi-user model five times, each one by excluding from the training set the images retouched by one of the experts. Then, in each simulation we selected a given number m of the 4000 training images and applied the voting or the fitting strategy to obtain the user profile of the expert whose images were excluded from training. In the case of the voting strategy for each image the vote was assigned to the expert whose retouched version was least dissimilar from that retouched by the target expert. In the case of the fitting strategy we simply used in Equation (5) the images retouched by the target expert. We tried with different number m of images and for each value of $m \in \{1, 3, 10, 30, 100, 300, 1000\}$ we carried out ten simulations for both the strategies. As a dissimilarity measure we used the average ΔE_{76} . For the fitting strategy we performed 100 iterations of gradient descent with a learning rate of 0.1.

For each simulation, after the computation of the profile, we evaluated the multiuser model by running it on the 1000 test images, and by comparing the output with the ground truth images retouched by the target experts. The outcome

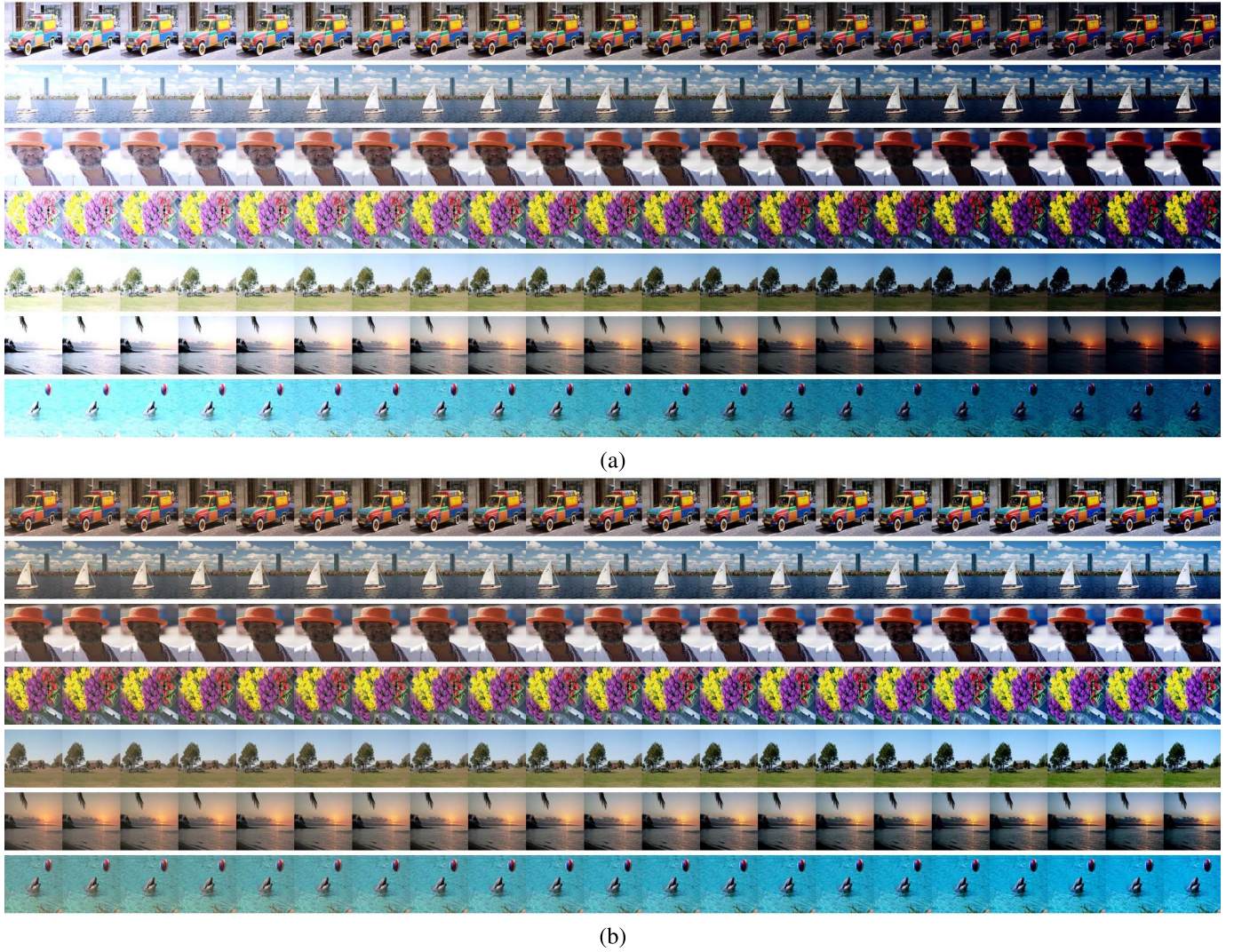


Fig. 13. Examples of image enhancement performed by the multi-user model by moving the user profile along the first (a) and the second (b) principal components in the user space. For each principal component the pictures span a range of ± 10 standard deviations from the center of the user space.

of the simulations is summarized in Figure 14. With the voting strategy we obtained promising performance, never far from those obtained by the model trained on all the five experts (i.e. those in Table III). Its behavior is very stable as the worst case is relatively close to the average one. The voting strategy presents one major weakness: the estimated user profile is restricted to be within the convex hull of the profiles of the known users. This means that when the optimal profile lies outside the convex hull we cannot obtain a good approximation. For instance, if we refer to Figure 12 this happens for expert A, since it cannot be expressed as a combination of the other four. In other words, the voting strategy cannot extrapolate the style of the new user outside the range of styles seen during training. This weakness is reflected in the plots: only for experts B, C and D the voting strategy allows to match the performance of the fully trained model.

The fitting strategy, instead, poses no restrictions to the placement of the new profile in the user space. In fact, in all the cases it allowed to obtain performance that was equal or even slightly better than that obtained by the trained model,

provided that at least ten images are used to fit the user profile. However, the lack of restrictions makes this strategy unreliable when a small number of images are used. In the worst case the fitting strategy can perform very poorly if less than ten images are used.

In terms of performance, the outcome of the simulations provided a clear recommendation: the voting strategy should be preferred when three or less images are used, otherwise the choice should fall on the fitting strategy. Beside that, the voting strategy has a couple of advantages over the fitting strategy: it is computationally less intensive, and it does not require the new user to actually retouch the images since he is just asked to choose among those already retouched by other users.

1) *Amateur Users*: We repeated the experiment described above by modeling five additional users that provided us 100 retouched FiveK images taken from the test set. For each of them we obtained their profiles with the voting and the fitting strategy by using the multi-user model trained on all the five FiveK experts. For each simulation a random subset of m of the 100 retouched images was used to model the users, and

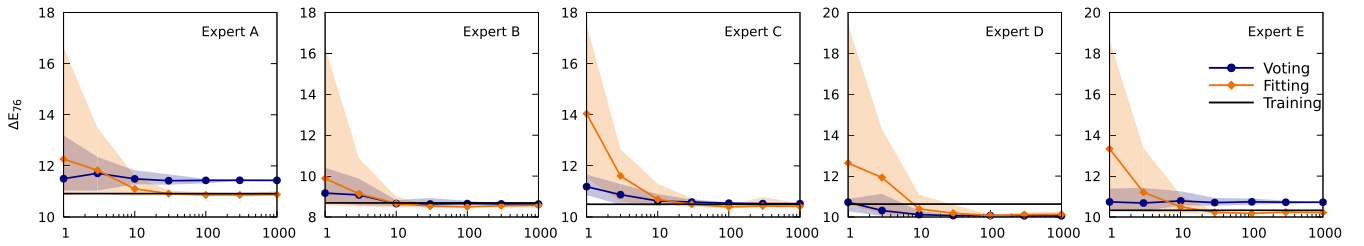


Fig. 14. Performance obtained on the test set of the FiveK dataset by modeling each expert with the voting and the fitting strategies. Each plot reports the mean ΔE_{76} over ten simulations, varying the number of images used to model the user. The boundaries of the shaded areas represents the best and the worse simulations, while the black line reports the performance obtained for the target expert with the multi-user model trained on all the experts.

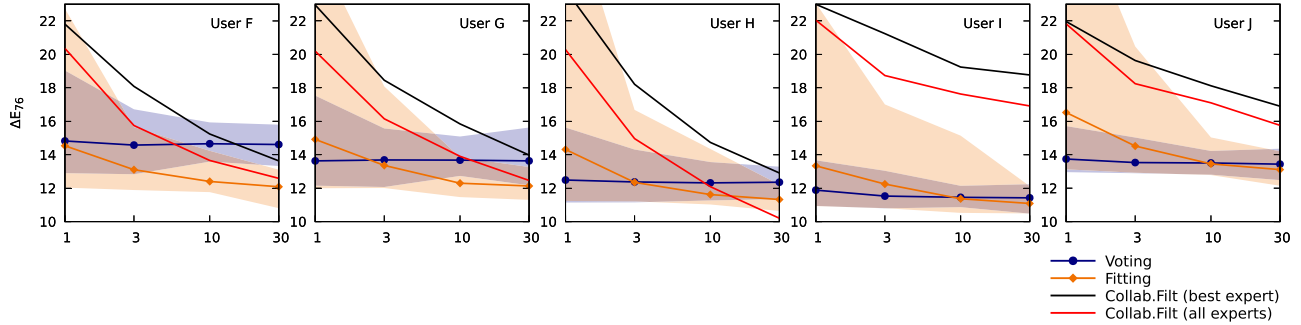


Fig. 15. Performance obtained on 100 FiveK test images by modeling five new users with the voting and the fitting strategies. Each plot reports the mean ΔE_{76} over ten simulations, varying the number of images used to model the user. The boundaries of the shaded areas represents the best and the worse simulations, while the black and red lines report two upper bounds to the performance that can be obtained by simulating collaborative filtering methods.

the remaining $(100 - m)$ images were used for performance evaluation (we tried with $m \in \{1, 3, 10, 30\}$). Figure 15 shows the results we obtained. In all the five cases we observed good performance for both strategies, with voting more consistent with a small number of images, and fitting more accurate when more images are available.

An alternative approach to the modeling of new users would be the use of collaborative filtering methods that leverage the collective knowledge of a community of users. Due to the limited pool of experts, we cannot really experiment this approach on the FiveK dataset. We devised anyway an experiment to show the potential and limitations of collaborative filtering for image enhancement. In collaborative filtering a new user would be compared to other users to select the most suitable correction curves for a given image. This would be done by looking to similar contents and styles among those already experienced by the community. We simulated this approach by using the m reference images retouched by a new user to select the most compatible expert as the one whose retouching curves are, on average, the closest to those of target user. Then, for each of the remaining $(100 - m)$ test images we:

- take the RGB curves used by the new user to correct the test image;
- select the closest RGB curves among the m set of curves used by the selected expert to retouch the reference images;
- apply the RGB curves found above to correct the test image.

We also repeated the experiment without the selection of the most compatible expert and by simply selecting the RGB curves among those of all the five experts for the m reference

images. The performance of the two simulations (with the “best” expert, and with all the experts) can be considered as a lower bound on the error that can be obtained with pure collaborative filtering. In fact in both the simulations we made use of the knowledge of the correction curves of the target user that in a real scenario would not be available. The results we obtained are reported in Figure 15. Our fitting strategy outperformed collaborative filtering in all the cases but one (User H when using 30 reference images). When 10 or less reference images are available, both the voting and the fitting strategies obtained better performance than collaborative filtering. This experiment shows how being adaptive to the content of the images (that is not used in pure collaborative filtering) is of primary importance for image enhancement. Furthermore, pure collaborative filtering is restricted to the corrections already experienced by the community, while our multi-user model is able to generalize to new image contents by effectively combining the information from the input image and the user’s profile.

VI. CONCLUSIONS

In this paper we presented SpliNet, a method for the enhancement of raw images combining convolutional neural networks and spline-based color curves. The method allows to retouch images preserving their content and without introducing any artifacts. The method can be used to reproduce a variety of retouching styles in the single-user and in the multi-user scenarios.

The results obtained demonstrate that SpliNet allows to reproduce with great fidelity the retouching styles of the individual experts in the FiveK dataset. Moreover, our method

performs favorably with respect to other methods that have been recently proposed in the literature. SpliNet can be used to model multiple experts at the same time without incurring any loss in accuracy or computational resources. Finally, we have shown how it is possible to reproduce the style of new users not seen during training by estimating their user profile on a very small set of reference images.

By design SpliNet is constrained to perform a global enhancement. This is often an advantage, but sometimes it can represent an unnecessary restriction. We are currently investigating promising ways of introducing the support for spatially varying transformations.

Our experiments also showed the major limitations of SpliNet. The method is sometimes too conservative (i.e. the saturation is slightly below the optimal level) in particular when facing uncommon subjects (e.g. abstract) or imaging conditions (e.g. scarce illumination). This is unfortunate since the help of an automatic enhancement system would be especially valuable when dealing with unusual images. In the future we plan to overcome these limitations by working in several directions. First of all, we believe that better performance could be achieved by adopting a larger dataset, including a more diverse range of subjects and imaging conditions. A larger dataset would also allow to make better use of the high-level knowledge that modern CNNs are able to reliably extract from the images. To this end, we plan to augment the proposed method by injecting additional semantic information obtained by suitably trained neural networks. Concerning the slightly dull look of some images, caused by the lack of saturation, we plan to improve it by introducing alternative loss functions, such as those already proposed in automatic colorization methods to address the same issue [35].

ACKNOWLEDGEMENT

The authors gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

REFERENCES

- [1] D. C. C. Wang, A. H. Vagnucci, and C. C. Li, "Digital image enhancement: A survey," *Comput. Vis., Graph., Image Process.*, vol. 24, no. 3, pp. 363–381, Dec. 1983.
- [2] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-Image translation with conditional adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5967–5976.
- [3] M. Gharbi, J. Chen, J. T. Barron, S. W. Hasinoff, and F. Durand, "Deep bilateral learning for real-time image enhancement," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 1–12, Jul. 2017.
- [4] D. Djurdjic. (Dec. 2018). *Four Reasons Why Curves Beats Other Tools in Photoshop*. [Online]. Available: <https://www.diyphotography.net/four-reasons-why-curves-beats-other-tools-in-photoshop/>
- [5] V. Bychkovsky, S. Paris, E. Chan, and F. Durand, "Learning photographic global tonal adjustment with a database of input/output image pairs," in *Proc. CVPR*, Jun. 2011, pp. 97–104.
- [6] S. M. Pizer *et al.*, "Adaptive histogram equalization and its variations," *Comput. Vis., Graph., Image Process.*, vol. 39, no. 3, pp. 355–368, 1987.
- [7] N. Moroney, "Local color correction using non-linear masking," in *Proc. Color Imag. Conf.*, no. 1. Springfield, VA, USA: Society for Imaging Science and Technology, 2000, pp. 108–111.
- [8] E. H. Land and J. J. McCann, "Lightness and retinex theory," *J. Opt. Soc. Amer.*, vol. 61, no. 1, p. 1, Jan. 1971.
- [9] D. J. Jobson, "Retinex processing for automatic image enhancement," *J. Electron. Imag.*, vol. 13, no. 1, p. 100, Jan. 2004.
- [10] D. Lischinski, Z. Farbman, M. Uyttendaele, and R. Szeliski, "Interactive local adjustment of tonal values," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 646–653, Jul. 2006.
- [11] X. An and F. Pellacini, "AppProp: All-pairs appearance-space edit propagation," *ACM Trans. Graph.*, vol. 27, no. 3, pp. 1–9, Aug. 2008.
- [12] S. B. Kang, A. Kapoor, and D. Lischinski, "Personalization of image enhancement," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 1799–1806.
- [13] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3431–3440.
- [14] J. Yang and M.-H. Yang, "Top-down visual saliency via joint CRF and dictionary learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 3, pp. 576–588, Mar. 2017.
- [15] X. Guo, Y. Li, and H. Ling, "LIME: low-light image enhancement via illumination map estimation," *IEEE Trans. Image Process.*, vol. 26, no. 2, pp. 982–993, Feb. 2017.
- [16] Z. Yan, H. Zhang, B. Wang, S. Paris, and Y. Yu, "Automatic photo adjustment using deep neural networks," *ACM Trans. Graph.*, vol. 35, no. 2, pp. 1–15, May 2016.
- [17] R. Wang, Q. Zhang, C.-W. Fu, X. Shen, W.-S. Zheng, and J. Jia, "Underexposed photo enhancement using deep illumination estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 6849–6857.
- [18] S. Bianco, C. Cusano, F. Piccoli, and R. Schettini, "Artistic photo filter removal using convolutional neural networks," *J. Electron. Imag.*, vol. 27, no. 1, Dec. 2017, Art. no. 011004.
- [19] A. Ignatov, N. Kobyshev, R. Timofte, and K. Vanhoey, "DSLR-quality photos on mobile devices with deep convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 3277–3285.
- [20] M. Omiya, E. Simo-Serra, S. Iizuka, and H. Ishikawa, "Learning photo enhancement by black-box model optimization data generation," in *Proc. SIGGRAPH Asia Tech. Briefs SA*, 2018, p. 7.
- [21] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired Image-to-Image translation using cycle-consistent adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2223–2232.
- [22] Y. Hu, H. He, C. Xu, B. Wang, and S. Lin, "Exposure: A white-box photo post-processing framework," *ACM Trans. Graph.*, vol. 37, no. 2, pp. 1–17, Jul. 2018.
- [23] J. Park, J.-Y. Lee, D. Yoo, and I. S. Kweon, "Distort-and-recover: Color enhancement using deep reinforcement learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5928–5936.
- [24] H. Yang, B. Wang, N. Vedapant, M. Guo, and S. B. Kang, "Personalized exposure control using adaptive metering and reinforcement learning," *IEEE Trans. Vis. Comput. Graphics*, vol. 25, no. 10, pp. 2953–2968, Oct. 2019.
- [25] J. C. Caicedo, A. Kapoor, and S. B. Kang, "Collaborative personalization of image enhancement," in *Proc. CVPR*, Jun. 2011, pp. 249–256.
- [26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
- [27] R. H. Bartels, J. C. Beatty, and B. A. Barsky, *An Introduction to Splines for Use in Computer Graphics and Geometric Modelling*. Los Altos, CA, USA: Morgan Kaufmann, 1998.
- [28] D.-T. Dang-Nguyen, C. Pasquini, V. Conotter, and G. Boato, "RAISE: A raw images dataset for digital image forensics," in *Proc. 6th ACM Multimedia Syst. Conf. MMSys*, 2015, pp. 219–224.
- [29] *Colorimetry—Part 4: Cie 1976 L* A* B* Colour Space*, document ISO,(11664-4: 2008 (CIE S 014-4/E: 2007)), 2008.
- [30] M. D. Fairchild, *Color Appearance Models*. Hoboken, NJ, USA: Wiley, 2013.
- [31] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.
- [32] A. Paszke *et al.*, "Automatic differentiation in pytorch," in *Proc. Autodiff Workshop NIPS*, Oct. 2017.
- [33] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent.*, 2015.
- [34] R. Caruana, "Multitask learning," *Mach. Learn.*, vol. 28, no. 1, pp. 41–75, 1997.
- [35] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Oct. 2016, pp. 649–666.



Simone Bianco (Member, IEEE) received the B.Sc. and M.Sc. degrees in mathematics and the Ph.D. degree in computer science from the University of Milano-Bicocca, Italy, in 2003, 2006, and 2010, respectively. He is currently an Associate Professor in computer science with the Department of Informatics, Systems, and Communication, University of Milano-Bicocca. His current research interests include computer vision, machine learning, optimization algorithms, and color imaging.



Flavio Piccoli received the Ph.D. degree in computer science from the University of Milano-Bicocca, Italy, in 2019. He has also done an abroad period at the Computer Vision Center (CVC), Learning and Machine Perception (LAMP) Group, Universitat Autònoma de Barcelona (UAB). He is currently a Postdoctoral Research Fellow of the Imaging and Vision Laboratory (IVL), Department of Informatics, Systems, and Communication (DISCo), University of Milano-Bicocca. He is focusing on several research activities. The main ones are image enhancement, image generation, and anomaly detection for automatic industrial quality inspection.



Claudio Cusano (Member, IEEE) received the master's and Ph.D. degrees in computer science from the University of Milano-Bicocca, in 2002 and 2006, respectively. Since 2002, he has been a fellow of the Multimedia Information Technologies Institute, National Research Council, Italy, where he started his research activity as a researcher through a grant. In 2006, he became a Postdoctoral Researcher with the Imaging and Vision Laboratory, University of Milano-Bicocca. Since 2012, he has been serving as an Assistant Professor in computer engineering with

the University of Pavia, where he became an Associate Professor in 2015. His topics of interest are focused on automatic image analysis and recognition, and include scene classification, texture analysis, color processing, face recognition, and 3D imaging.



Raimondo Schettini (Member, IEEE) is a Full Professor with the University of Milano-Bicocca, Italy. He is the Head of the Imaging and Vision Laboratory. He has been a team leader of many research projects and published more than 300 refereed papers. He held several patents about image and video analysis, retrieval, and classification. He is a fellow of the International Association of Pattern Recognition. He has been the general/program chair of several international workshops and conferences, and associate/guest editors of many international journals.