Hyperspectral demosaicking and crosstalk correction using deep learning

Dijkstra, Klaas; van de Loosdrecht, J.; Schomaker, L. R. B.; Wiering, M. A.

Published in:
Machine Vision and Applications

DOI:
10.1007/s00138-018-0965-4

Publication date:
2018

Machine Vision and Applications (2019) 30:1–21
https://doi.org/10.1007/s00138-018-0965-4

ORIGINAL PAPER

Hyperspectral demosaicking and crosstalk correction using deep learning

K. Dijkstra[1,2] · J. van de Loosdrecht[1] · L. R. B. Schomaker[2] · M. A. Wiering[2]

Received: 22 December 2017 / Revised: 24 May 2018 / Accepted: 11 July 2018 / Published online: 26 July 2018

Abstract
Precision agriculture using unmanned aerial vehicles (UAVs) is gaining popularity. These UAVs provide a unique aerial perspective suitable for inspecting agricultural fields. With the use of hyperspectral cameras, complex inspection tasks are being automated. Payload constraints of UAVs require low weight and small hyperspectral cameras; however, such cameras with a multispectral color filter array suffer from crosstalk and a low spatial resolution. The research described in this paper aims to reduce crosstalk and to increase spatial resolution using convolutional neural networks. We propose a similarity maximization framework which is trained to perform end-to-end demosaicking and crosstalk-correction of a 4×4 raw mosaic image. The proposed method produces a hyperspectral image cube with 16 times the spatial resolution of the original cube while retaining a median structural similarity (SSIM) index of 0.85 (compared to an SSIM of 0.55 when using bilinear interpolation). Furthermore, this paper provides insight into the beneficial effects of crosstalk for hyperspectral demosaicking and gives best practices for several architectural and hyperparameter variations as well as a theoretical reasoning behind certain observations.

Keywords Demosaicking · Hyperspectral imaging · Deep learning · Precision agriculture · UAVs

Mathematics Subject Classification 68T45

1 Introduction

Inspection of agricultural fields using hyperspectral cameras and unmanned aerial vehicles (UAVs) is gaining popularity [1,2]. It is well known that increasing the spectral resolution can lead to more information about the properties of vegetation [3]. Crop recognition [4] has been performed using regular color channels (Red, Green and Blue). By expanding these measurements to near-infrared and the red-edge spectral ranges, Chlorophyll can be estimated to quantify overall vegetation health [5]. By further increasing the spectral resolution, diseases [3] and soil concentrations can be determined [6].

✉ K. Dijkstra
k.dijkstra@nhl.nl

[1] Centre of Expertise in Computer Vision and Data Science, NHL Stenden University of Applied Sciences, P.O. Box 1080, 8900 CB Leeuwarden, The Netherlands

[2] Bernoulli Institute for Mathematics, Computer Science and Artificial Intelligence, University of Groningen, P.O. Box 407, 9700 AK Groningen, The Netherlands

The properties of UAV-based camera systems are potentially ideal for the inspection of agricultural fields. UAVs are non-invasive. (They do not interact directly with the vegetation.) UAVs provide an aerial perspective at various heights and resolutions. UAVs can use GPS waypoints, visual features [7] or sensor fusion [8] to navigate automatically. However, one of the major downsides of UAVs is their limited payload capacity and the associated flight time. When aiming to utilize UAVs for precision agriculture efficient hyperspectral imaging devices are required.

Hyperspectral and multispectral imaging technologies can be divided into three major categories [9]. Multi-camera-one-shot describes a class of systems in which each spectral band is recorded using a separate sensor [10]. Examples are: Multiple cameras with different optical filters or multi-CCD cameras. The weight increase by the special optics and/or multiple cameras makes this class of systems not ideally suited for utilization on a UAV.

Single-camera-multi-shot systems aim to use a single camera to record different spectral bands at separate times. This includes filter-wheel setups, liquid crystal tunable filters

(LCTF) and line-scan hyperspectral cameras [11]. Because each image is delayed in time, it is difficult to get a correctly aligned spectral cube and to compensate for object movement (e.g., leaves moving in the wind).

An interesting class of cameras for UAVs are single-camera-one-shot. A standard RGB camera with a Bayer filter [12] is an example of this type of system. Recently these types of imaging systems have been further extended to 3×3, 4×4 and 5×5 mosaics [13] in both visible and near-infrared spectral ranges. This technology could potentially accommodate an arbitrary configuration of spectral bands. The main advantages of these sensors are their small size, low weight and the fact that each spectral band is recorded at the same time. This makes them suitable for a moving UAV. However these sensors suffer from a detrimental effect called crosstalk [14], which means that distinct spectral channels also receive some response of other spectral bands. These sensors also sacrifice spatial resolution to gain spectral resolution [15]. These effects become increasingly detrimental as the mosaic sizes increase and physical-pixel sizes decrease. An interpolation method for demosaicking beyond Bayer filter interpolation is not well defined.

A color filter array (CFA) or Bayer pattern [12] is a 2×2 mosaic of Red, Blue and Green twice, which is repeated over the entire imaging sensor. This resembles the visual appearance of a mosaic. With this CFA, each one of the 4 sensor elements are sensitive to either Red, Green or Blue. This means that not all three color spectra are known at all spatial locations. An unknown spectral band of a pixel is interpolated using Bayer interpolation [16]. This is essentially a regular zooming of each channel using bilinear pixel interpolation.

Because Bayer interpolation does not explicitly exploit information contained in the scenes (edges, shapes, objects), chromatic aberrations can be present in the interpolated images, mainly around strong image edges. These aberrations can be mitigated by incorporating edge information in the interpolation algorithm [9]. An interpolation method which learns directly from the image data by means of a shallow neural network, on several 2×2 mosaic images, is proposed in [17].

Demosaicking of 4×4 mosaic images is proposed in [18], using a greedy inpainting method. Additionally, a fast and trainable linear interpolation method is described in [19] for arbitrary sensor sizes. Recently, demosaicking algorithms integrate other tasks like noise reduction and use deep neural networks [20].

Image demosaicking is related to single image super resolution (SISR) [7,21]. Spectacular SISR has been achieved using convolutional neural networks (CNNs) [22]. This success is mainly due to upscaling layers which are also used in semantic image segmentation [23] and 3D Time of Flight (ToF) upscaling [24]. These algorithms benefit greatly from the information contained in the scenes of a large set of images. The main idea of SISR is to downsample images and to train a model that tries to reconstructs the original image. Our method uses a similar strategy. However mosaic images contain additional spectral correlations [25] which can be exploited. This makes demosaicking using a CNN even more prone to improvement.

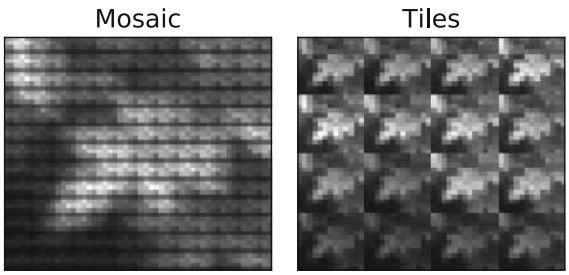

Fig. 1 Hyperspectral image with a 4×4 mosaic (left) and the actual spatial resolution of each channel (right)

Figure 1 (left) shows the raw image produced by a 4×4 mosaic sensor. The right image shows each of the 16 bands as separate tiles, which shows the actual spatial resolution of each channel. Because of the mosaic layout of the sensor, additional spatial information can possibly be uncovered by combining the information contained in each channel. The aim of this research is to increase the spatial resolution and decrease crosstalk of hyperspectral images taken with a mosaic image sensor.

By taking advantage of the flexibility and trainability of deep neural networks [26–28], a similarity maximization framework is designed to produce a full-resolution hyperspectral cube from a low-resolution hyperspectral mosaic using a CNN.

Our demosaicking results will be quantitatively evaluated with the structural similarity (SSIM) index [29] which is a full-reference metric often used to compare visual artifacts in images [30] and for evaluating SISR. Results are also qualitatively evaluated as visual images to give an intuition for the interpretation of this SSIM metric.

Our observations from several proposed solutions for demosaicking and crosstalk correction leads to three research questions:

1. How much does hyperspectral demosaicking benefit from spectral and spatial correlations?

Three major kinds of correlations exist in images taken with a mosaic sensor. (1) Each spectral filter is at a different location in the sensor mosaic so each spectral band gives additional spatial information. (2) Crosstalk between different spectral bands gives additional spectral information at spatial locations of other spectral bands. (3) Finally, correlations in scene contents are present at visually similar parts of the image.

This hypothesizes that hyperspectral demosaicking could actually benefit from spectral and spatial correlations in the image data. This will be investigated by demosaicking images with various degrees of crosstalk, and using various convolution filter footprints and training image sizes.

2. What are good practices for designing hyperspectral demosaicking neural networks?

Designing deep neural networks and tuning their hyperparameters can be quite cumbersome. This paper presents several variations of deep neural network designs and compares their performance both quantitatively and qualitatively.

3. How well can hyperspectral demosaicking sub-tasks be integrated for end-to-end training?

A particularly strong feature of deep neural networks is their ability to provide an end-to-end trainable solution to an increasing amount of challenges. The demosaicking task can be split into three sub-tasks: (1) Conversion from a raw-sensor mosaic to a 3-d hyperspectral cube, (2) upscaling and (3) crosstalk correction. These tasks are designed as separate steps toward the final solution. At the end of this paper, the effect of integrating these sub-tasks in an end-to-end neural network will be presented. This is the preferred solution, because training is incorporated in each stage of demosaicking.

1.1 Outline of this paper

In the next section, a brief introduction of the neural network principles used in this paper is given. Section 3 describes the imaging device and the dataset which has been used. Our similarity framework is explained in detail in Sect. 4. The design of our experiments is given in Sect. 5. Quantitative and qualitative results are discussed in Sect. 6. In the last two sections, the conclusions (Sect. 7) and future work (Sect. 8) are discussed.

2 Convolutional neural networks

A convolutional neural network (CNN) consists of several layers of neurons stacked together where at least one layer is a convolutional layer. The first layer receives the input vector and the output of a layer is passed as an input to the next layer. The final layer produces the output vector. Training a neural network requires a forward step which produces the output of the network and a backward step to update the weights of the network based on the desired output. The theory of CNNs is large, and for a comprehensive explanation we would like to refer the reader to [31].

To introduce the basic concepts of the neural networks used in this paper, the forward and backward steps of a single-layer neural network are explained. This network is very similar to the one we use for crosstalk correction. This section also briefly explains the convolutional layer, the inner-product layer and the deconvolution layers that are used in this research.

2.1 A basic single-layer neural network

At the basis of a neural network is a neuron which takes a linear combination between the input vector $\mathbf{x}$ and a weight vector $\mathbf{w}$. The scalar output is transformed using a nonlinear activation function g.

$$o = g\left(\mathbf{x}^\top \cdot \mathbf{w}\right) \tag{1}$$

In this paper, the activation function $g(\cdot)$ takes the form of either the sigmoid function ϕ or the Rectified Linear Unit (ReLU) ψ. The sigmoid function produces a soft clipping between zero and one and the ReLU function clips input values which are below zero but keeps values above zero.

$$\phi(x) = \frac{1}{1+e^{-x}} \tag{2}$$

$$\psi(x) = \max(0, x) \tag{3}$$

Multiple neurons are organized in layers. The input vectors to a layer are represented by an input matrix $\mathbf{I}$, the desired output vectors are given by matrix $\mathbf{Y}$ and the weight matrix $\mathbf{W}$ contains the weight vectors of each individual neuron in the layer.

$$\mathbf{X} = \left[\mathbf{x}_1^\top, \mathbf{x}_2^\top, \ldots, \mathbf{x}_n^\top\right] \tag{4}$$

$$\mathbf{Y} = \left[\mathbf{y}_1^\top, \mathbf{y}_2^\top, \ldots, \mathbf{y}_n^\top\right] \tag{5}$$

$$\mathbf{W} = \left[\mathbf{w}_1^\top, \mathbf{w}_2^\top, \ldots, \mathbf{w}_m^\top\right] \tag{6}$$

where n indicates the number of input vectors and m the number of neurons.

The output of the neurons are produced by multiplying the transposed input matrix $\mathbf{X}$ with the weight matrix $\mathbf{W}$ and applying an activation function $g(\cdot)$ to each element.

$$\mathbf{O} = g\left(\mathbf{X}^\top \mathbf{W}\right) \tag{7}$$

where $\mathbf{O}$ is the output matrix with $n \times m$ elements containing m neuron outputs for each of the n input vectors.

In the forward step, an input matrix is presented to the network and an output matrix is produced. A loss between the desired input and the output is computed to get the current error of the neural network. In this paper, the mean squared

error (MSE) loss e is used for all networks. The MSE loss calculates the average squared difference between the network output $\mathbf{O}$ and the desired output $\mathbf{Y}$.

$$e = \frac{1}{2n} \sum_{y=1}^{n} \sum_{x=1}^{m} \left(\mathbf{O}_{yx} - \mathbf{Y}_{yx}\right)^2 \tag{8}$$

In the backward step, a layer is trained by adjusting the weight matrix iteratively to converge toward the lowest loss using gradient descent. The weights are updated by calculating the derivative of the loss function with respect to the weight matrix $\mathbf{W}$. The momentum μ prevents oscillations during training by adding a small factor of the previous weight change. The new weight matrix $\mathbf{W}'$ is calculated by adding a factor α (learning rate) of the derivative to the current weights:

$$\mathbf{\Delta W}_t = -\frac{\partial e}{\partial \mathbf{W}_t} + \mu \times \mathbf{\Delta W}_{t-1} \tag{9}$$

$$\mathbf{W}_{t+1} = \mathbf{W}_t + \alpha \times \mathbf{\Delta W}_t \tag{10}$$

where $\mathbf{W}_t$ are weights at time-step t, and e, α and μ are the loss, learning rate and momentum.

The backward and forward steps are repeated in several epochs until the weights of the network stabilize.

2.2 Training the layers of the CNN

A typical CNN uses a slightly adapted training approach which is referred to as stochastic gradient descent (SGD). With SGD, the inputs are presented to the network in several batches which is more efficient when training using a GPU [32]. Training a single batch is referred to as an iteration. When all batches have been trained an epoch has elapsed. A network is trained using multiple epochs.

In this paper, three types of layers are used: the inner-product layer, the convolutional layer and the deconvolution layer.

2.2.1 Inner-product layer

In an inner-product layer (or sometimes called a fully connected layer), all inputs are connected to all outputs. This layer has been explained in the previous subsection and is defined by a matrix multiplication between the input matrix and the weights matrix followed by an activation function (Eq. 7).

2.2.2 Convolutional layer

A convolutional layer accepts a multi-dimensional input tensor. In our case, this is a hyperspectral cube with two spatial dimensions and one spectral dimension. It convolves the input using multiple trained convolution kernels. In contrast to the inner-product layer, the convolutional layer provides translation invariance. Instead of having a weight associated with each input element of the tensor, weights are shared between image patches. In this paper, the convolutional layer is denoted by the $\otimes$ symbol.

Fig. 2 Example images taken from the UAV. These 16-channel images have been converted to RGB for displaying

2.2.3 Deconvolution layer

The deconvolution layer (or strided convolution) performs an inverse convolution of the input tensor. With a deconvolution layer, the spatial dimensions of the output tensor are larger than the original input tensor. Deconvolution layers are used to upscale images to higher spatial [22] resolutions. The trick is to first pad the input tensor with zeros between individual spatial elements. Then a convolution is performed and weights of the convolution kernel are trained. This layer plays the most prominent role in demosaicking and is denoted in this paper with the $\oslash$ symbol.

3 Sensor geometry and datasets

A 10 bit, 4 × 4 mosaic sensor is used to make a dataset of 2500 aerial images. The mosaic sensor layout is shown in Table 1. A camera was mounted on a gimbal under a UAV. It navigated over an area of 15 m × 150 m at an altitude of 16 m altitude. A lens with a 35 mm focal length was used with the aperture set to 1.4. The scene contained mainly potato plants, grass and soil. A short-pass filter of 680 nm has been used to filter unwanted spectral ranges. In Fig. 2, a few example images are shown. The set is split into an East and a West set (separate ends of the field) containing 1000 and 1500 images. The East set is used for training, and the West set is used for validation. Although the set is quite large, only a random subset of images is used for the experiments.

3.1 Calibration data

The camera vendor provides calibration data containing the response of the MCFA sensor for 16 spectral ranges. A calibrated light source illuminates a white reference [33]. The color is adjusted with increments of 1 nm from 400 to

Table 1 Layout for the 4 × 4 mosaic sensor

489 nm	496 nm	477 nm	469 nm
600 nm	609 nm	586 nm	575 nm
640 nm	493 nm	633 nm	624 nm
539 nm	550 nm	524 nm	511 nm

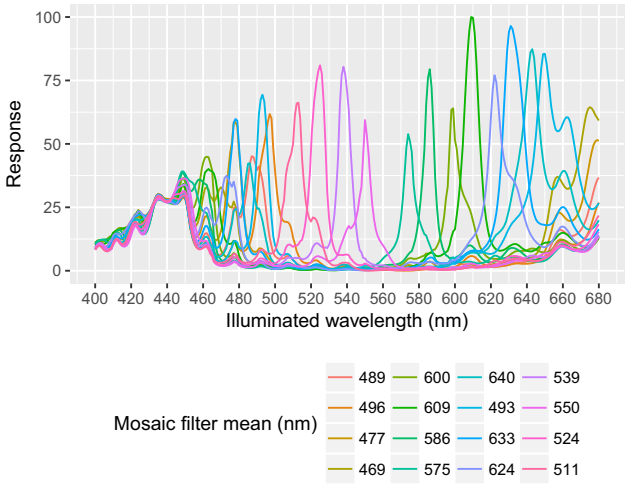

Fig. 3 Measured calibration data of the 4 × 4 mosaic sensor. Showing responses for all 16 mosaic filter wavelengths

1000 nm. The response of each spectral pixel for all 16 spectral bands is measured. Values for the same spectral band are averaged for multiple spectral pixels in the image. This produces 16 measurements per 1 nm increment of the illumination source. In Fig. 3, crosstalk between the various filters can be observed. Because the 4 × 4 mosaic on the camera sensor contains a band-pass filter, only the responses of the spectral range from 400 to 680 nm are shown.

4 Similarity maximization

In this section, our similarity maximization framework for demosaicking hyperspectral images is proposed. This framework is shown in Fig. 4. Each arrow in the diagram represents an operator of the framework. Two squares represent the convolutional neural network (CNN) in both the training phase and the validation phase.

The left part of Fig. 4 illustrates the procedure of training the CNN. A region of the input image is downsampled to $\frac{1}{16}$th of the original size without filtering. This is denoted by the dashed square in the input image. A CNN is trained to upscale this region back to the original size. A loss is calculated by comparing the upscaled and the original region. This loss is then back propagated to update the weights of the CNN. With each iteration, the CNN gets better at reconstructing the image.

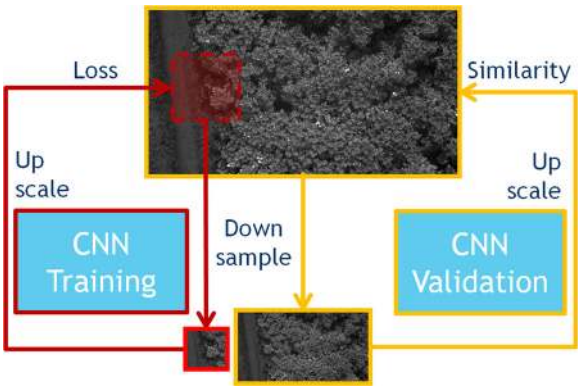

Fig. 4 Similarity maximization framework. The left part shows training of the neural network. The right part shows validation of the neural network

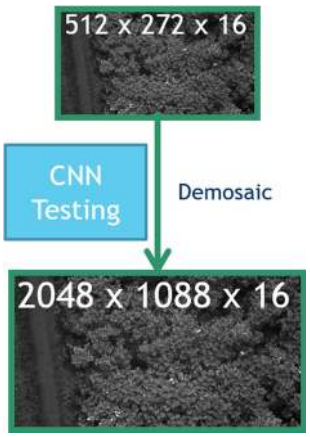

Fig. 5 Testing the neural network for final demosaicking

The right part of Fig. 4 illustrates the procedure for validating the quality of the reconstruction. The original image is downsampled and then upscaled using the trained CNN. The structural similarity (SSIM) index is used to quantitatively evaluate the reconstruction.

Final demosaicking of a hyperspectral image is achieved during the testing phase illustrated in Fig. 5. The trained CNN produces a full-resolution demosaicked hyperspectral cube of 2048 × 1088 × 16 from a hyperspectral cube of 512 × 272 × 16 pixels. In the testing phase no ground-truth is available for the images.

The demosaic and the upscale operators both use identical (de)convolution kernels. However in our definition demosaicking produces the final image and upscaling just reconstructs the downsampled patches or images for training and evaluation. Furthermore a mosaic image is defined as the 2-d image produced by the imaging sensor and the hyperspectral cube is defined as the 3-d hyperspectral structure.

In the next subsections, all operators of our similarity framework are explained in detail.

4.1 Normalization

A typical neural network needs normalized values between zero and one. The normalization operator $\mathrm{NR}(\cdot)$ normalizes the values of the hyperspectral cubes by multiplying each element of the tensor by a scalar. Output values of the neural network can be scaled back before display by the inverse operator $\mathrm{NR}^{-1}(\cdot)$:

$$\mathrm{NR}(\mathbf{I}) = \mathbf{I} \times \frac{1}{2^{bpp} - 1} \tag{11}$$

$$\mathrm{NR}^{-1}(\mathbf{I}) = \mathbf{I} \times \left(2^{bpp} - 1\right) \tag{12}$$

where $\mathbf{I}$ is the input cube and bpp is the amount of bits per pixel of the imaging sensor (in our case 10 bpp).

The normalization operators are implicit throughout the paper. When explicitly referring to unnormalized tensors, an accent ($'$) notation is used.

4.2 Mosaic to cube

Converting pixel values from the mosaic image to a spectral cube is not entirely trivial because spatial and spectral information is intertwined in a mosaic image. This conversion can be handcrafted, but can also be implemented as a convolutional neural network layer with a specific stride.

The mosaic-to-cube operator generates a 3-d structure $\mathbf{I}$ with two spatial axes and a separate spectral axis from the original 2-d mosaic $\mathbf{M}$:

$$\mathbf{I} = \mathrm{MC}(\mathbf{M}) \tag{13}$$

No information is removed or added during this operation. Only a $2048 \times 2048 \times 1$ mosaic image is converted to a $512 \times 512 \times 16$ hyperspectral cube.

The handcrafted mosaic-to-cube operator is defined as:

$$\mathbf{C}_{x,y,z} = \mathbf{M}_{x \times s + z \bmod s,\, y \times s + z \operatorname{div} s} \tag{14}$$

$$\mathrm{MC}_{\mathrm{hc}}(\mathbf{M}) = \mathbf{C} \tag{15}$$

where $\mathbf{M}$ is the input mosaic image with a subscript indicating the 2-d pixel coordinate, the 3-d coordinate in the hyperspectral cube $\mathbf{C}$ is denoted by x, y and z. The size of the mosaic is denoted by s which is 4, for a 4×4 mosaic. The operators div and mod are an integer division and modulo.

An alternative implementation of the mosaic-to-cube operator uses a convolutional layer:

$$\mathcal{G} = \left\{ \mathbf{G}_{\mathbf{1}}^{\mathbf{9}\times\mathbf{9}}, \mathbf{G}_{\mathbf{2}}^{\mathbf{9}\times\mathbf{9}}, \ldots, \mathbf{G}_{\mathbf{16}}^{\mathbf{9}\times\mathbf{9}} \right\} \tag{16}$$

$$\mathrm{MC}_{nn}(\mathbf{M}) = \mathbf{M} \otimes_4 \mathcal{G} \tag{17}$$

where the $\mathbf{G}$ matrices denote 9×9 convolutional filters, $\mathcal{G}$ denotes the filter bank of 16 filters (The amount of spectral planes), and $\otimes_4$ is the convolution operator with a stride equal to the mosaic size, 4.

This convolutional method for the mosaic-to-cube conversion will be identical to the handcrafted method if one element of each filter contains the value one (it selects the correct mosaic pixel from the image mosaic). There is some freedom in choosing the size of these convolution filters. The theoretical minimum size is 4×4. With a filter size of 9×9, mosaic pixels from all around the current mosaic pixel can be used by the network. In practice an oddly sized convolutional layer is used so the padding for all sides of the input image is the same. The weight initialization is uniform random.

Training this $\mathrm{MC}_{nn}(\cdot)$ operator in an end-to-end fashion with the rest of the neural network will be investigated in Sect. 5. In these results, it is shown that the learned filters select specific mosaic-pixel regions from the image mosaic as expected.

4.3 Downsampling

The downsampling operator generates a low-resolution mosaic image from an original mosaic image. This operator is designed to give information on what the demosaicked version of a lower-resolution image would look like. The downsampling operator $\mathrm{DS}(\cdot)$ is defined by:

$$\mathbf{N}_{x,y} = \mathbf{M}_{x \times s + x \bmod s,\, y \times s + y \bmod s} \tag{18}$$

$$\mathrm{DS}(\mathbf{M}) = \mathbf{N} \tag{19}$$

where $\mathbf{M}$ and $\mathbf{N}$ are the original and the downsampled mosaic images with a subscript indicating the mosaic-pixel coordinate in the mosaic image, x, y are coordinates within the downsampled mosaic image and s is the size of the mosaic pattern.

Finally the downsampled spectral cube is produced by

$$\mathbf{D} = \mathrm{MC}(\mathrm{DS}(\mathbf{M})) \tag{20}$$

where $\mathrm{DS}(\cdot)$ and $\mathrm{MC}(\cdot)$ are the downsample and mosaic-to-cube conversion operators.

An important feature of this downsampling method is that it respects the spatial/spectral correlations of the mosaic pattern by selecting different spectral bands ($\bmod s$) at different coordinates (x and y). The main reason for this is to ensure that the learned upscaling is not too different from demosaicking. The downsampled image has an area which is s^2 times smaller than the original image (16 times for a 4×4 mosaic). By choosing a downsampling factor which aligns with the mosaic only whole mosaic pixels are sub-sampled. This means that no additional filtering is required or even desired.

4.4 Upscaling

Upscaling is at the heart of our similarity maximization framework. Because of the close relation between the frequently used Bayer interpolation and bilinear upscaling, we compare several designs of our convolutional neural network architecture to a standard bilinear interpolation method. The upscaling operator will be investigated quantitatively with a full-reference metric (SSIM). These experiments can be found in Sect. 5.

The upscaling operator $\mathrm{US}(\cdot)$ scales a hyperspectral cube, or 3-d tensor, to another hyperspectral cube with a higher spatial resolution.

In general a set of convolutional filters in a filter bank is given by

$$\mathcal{F}_m^{t\times t} = \left\{\mathbf{F_1^{t\times t}}, \mathbf{F_2^{t\times t}}, \ldots, \mathbf{F_m^{t\times t}}\right\} \tag{21}$$

where $\mathbf{F^{t\times t}}$ is 3-d tensor with the first two dimension sizes set to $t \times t$ and m is the number of filters in the filter bank.

Note that all convolution filters in $\mathcal{F}$ are three dimensional because they act on hyperspectral cubes. Only the first two dimensions $t \times t$ are specified as hyperparameters. The third dimension of the convolution filter is the same as the size of its input tensor. For example, if the input tensor is the hyperspectral cube the third dimension is equal to the amount of spectra (16 in our case).

The specialization of the upscaling operator for bilinear interpolation is defined by

$$US_{bl}(\mathbf{D}) = \phi\left(\mathbf{D} \oslash_4 \mathcal{F}_{16}^{8\times 8}\right) \tag{22}$$

where $\mathbf{D}$ is the downsampled input tensor, ϕ is the Sigmoid activation function and the deconvolution operator is denoted by $\oslash_4$, where the subscript 4 determines the stride of the convolution which in turn is responsible for the upscaling factor in both x and y directions.

The convolution filters are initialized by a bilinear filler type described in [34]. When using a single deconvolutional layer, mainly linear upscaling can be achieved (the function ϕ is responsible for some nonlinearity).

To introduce more nonlinearity, at least two deconvolution layers are used. The first layer determines the capacity of the neural network and the second layer should have the same amount of filters as the number of required spectral bands in the output:

$$US_m^{t\times t}(\mathbf{D}) = \phi\left(\phi\left(\mathbf{D} \oslash_2 \mathcal{F}_m^{t\times t}\right) \oslash_2 \mathcal{F}_{16}^{t\times t}\right) \tag{23}$$

where $US_m^{t\times t}(\cdot)$ is the upscaling operator with m filters of size $t \times t$. Each deconvolution operator performs an upscaling of $2 \times 2 = 4$ times, which results in a total spatial upscaling factor of $4 \times 4 = 16$.

To avoid extrapolating beyond the original resolution, the product of the strides of both deconvolution layers should not exceed the size of the mosaic. This presents an interesting theoretical implication: Optimal sizes for mosaic sensors should ideally not be prime numbers. These cannot be upscaled with multiple consecutive deconvolution layers to introduce nonlinearity. For example a 5×5 mosaic, currently also available on the market in the Near-InfraRed (NIR) range, can only be demosaicked using a single deconvolution layer.

The upscaling operators which only use a single deconvolution layer are referred to in the text as linear upscaling and the upscaling operators using more than one deconvolution layer are referred to as nonlinear upscaling.

4.5 Demosaicking

There is a subtle difference between the upscaling and the demosaicking operator. Following the definition of the upscaling operator, CNNs are trained to reconstruct original images from downsampled images. The demosaicking operator is actually the final goal of hyperspectral demosaicking. This is what produces a high-resolution hyperspectral cube from a low-resolution cube. The main difference between the upscaling operator and the demosaicking operator is the size of the input and output tensors.

The upscaling operator in our similarity maximization framework is trained on small regions of the original image. Because these regions are downsampled to $\frac{1}{16}$th of the original size, the neural network is trained to enlarge a downsampled region from $\frac{1}{16}$th to its original resolution.

The demosaicking operator uses this trained neural network to enlarge an original image by a factor 16. This results in an interesting trade-off regarding the footprint size of the deconvolution filters. The footprint should be sufficiently large to interpolate between spatial structures. At the same time, the footprint should be kept sufficiently small so that the neural network learns to generalize between increasing the spatial resolution from $\frac{1}{16}$th to the original resolution and to increase the original resolution by a factor of 16.

In our case, the footprint of the deconvolution filters is kept sufficiently small so the network cannot exploit large spatial structures in the images. The idea is that this helps generalize the upscaling operator to be suitable as a demosaicking operator. Another difference between the upscaling operator and the demosaicking operator is that the demosaicking operator can and will only be evaluated visually because the full-resolution demosaicked image is not known a-priori, and thus, a full-reference comparison cannot be performed.

4.6 Loss function

A loss is calculated between the upscaled cube **U** and the original cube **I**. A popular method for calculating loss is the Euclidean loss which is both fast and accurate.

The operator LS(·) calculates the MSE loss between two tensors of equal dimensions and size:

$$\mathrm{LS}(\mathbf{U}, \mathbf{I}) = \sqrt{\sum_{i=1}^{c} (\mathrm{U}_i - \mathrm{I}_i)^2} \tag{24}$$

where c is number of elements of the 3-d tensor and the subscript i indicates an element of the tensors.

The loss estimates the degree of similarity between two hyperspectral cubes and is back-propagated through the neural network in a fashion similar to the algorithm described in Sect. 2.1.

4.7 Structural similarity

The Euclidean loss gives a fast and unnormalized metric of similarity which is used for training. For quantitatively comparing the measure of equality between two spectral cubes, the structural similarity (SSIM) index is used [30]. This metric calculates three different aspects of similarity where a value of zero means that there is no similarity and a value of one means that the spectral cubes are identical. The SSIM index is a symmetric similarity metric meaning that switching the input tensors has no effect on the output value. A brief summary of the algorithm will be given:

First a weight matrix **G** is constructed using an 11×11 Gaussian function with a 1.5 standard deviation with a sum of one. A sliding window is used to calculate luminance, contrast and structure at every pixel (each channel of the hyperspectral image is calculated separately):

$$\mu_x = \mathbf{g}^\top \mathbf{x} \tag{25}$$

$$\mu_y = \mathbf{g}^\top \mathbf{y} \tag{26}$$

$$\sigma_x = \sqrt{\sum_{i=1}^{n} (\mathbf{g}_i \times (\mathbf{x}_i - \mu_x))^2} \tag{27}$$

$$\sigma_y = \sqrt{\sum_{i=1}^{n} \left(\mathbf{g}_i \times \left(\mathbf{y}_i - \mu_y\right)\right)^2} \tag{28}$$

$$\sigma_{xy} = \sqrt{\sum_{i=1}^{n} \mathbf{g}_i \left(\mathbf{x}_i - \mu_x\right) \left(\mathbf{y}_i - \mu_y\right)} \tag{29}$$

where **g** is the 1-d contiguous vector of the weight matrix **G**, the 1-d contiguous vector of pixels from a single 2-d patch is denoted by **x** and **y**, n is the number of elements in these vectors ($11 \times 11 = 121$).

The per-channel SSIM index is calculated at each spatial coordinate of the hyperspectral plane by

$$\mathrm{SSIM}(\mathbf{x}, \mathbf{y}) = \frac{\left(2\mu_x\mu_y + C1\right)\left(2\sigma_{xy} + C2\right)}{\left(\mu_x^2 + \mu_y^2 + C1\right)\left(\sigma_x^2 + \sigma_y^2 + C2\right)} \tag{30}$$

where $C1 = (0.01 \times 2^{bpp})^2$ and $C2 = (0.03 \times 2^{bpp})^2$ are constants taken from the original paper [29].

The final mean SSIM index between two hyperspectral cubes is calculated by first taking the mean over all patches and then taking the mean over all channels by:

$$\mathrm{MSSIM}(\mathbf{X}_c, \mathbf{Y}_c) = \frac{1}{n} \sum_{i=1}^{n} \mathrm{SSIM}(\mathbf{X}_{ci}, \mathbf{Y}_{ci}) \tag{31}$$

$$\mathrm{SI}(\mathbf{X}, \mathbf{Y}) = \frac{1}{nc} \sum_{c=1}^{nc} \mathrm{MSSIM}(\mathbf{X}_c, \mathbf{Y}_c) \tag{32}$$

where **X** and **Y** are two tensors where c indicates the image plane of spectral channel c. Subscript i indicates a 1-d contiguous vector of an 11×11 image patch of the tensors and n and nc are the number of image patches and number of channels, respectively. The final similarity operator $SI(\cdot)$ calculates the average similarity over all channels and is used to estimate similarities between upscaled and original spectral cubes. In the text, the output of the SI(·) operator is mostly referred to as the 'SSIM index'.

4.8 Crosstalk correction

A mosaic imaging sensor suffers from crosstalk. This means that each filter in the mosaic is not only sensitive to the designed spectral range, but information from other bands bleeds through. This is mostly regarded as an unwanted effect and can be observed by a desaturation of the image colors [14].

A linear method for correcting crosstalk is proposed in [33]. The crosstalk between spectral responses for a spectral pixel is corrected by talking a linear combination of all spectral responses for that specific pixel:

$$\mathrm{CT}(\mathbf{X}) = \psi\left(\mathbf{X}^\top \mathbf{W}\right) \tag{33}$$

where **X** is a matrix containing column vectors of spectral responses, **W** is the crosstalk-correction matrix and CT(·) is the crosstalk-correction operator. The ReLU function ψ clips values below zero because negative spectral responses cannot exist. This attributes to some nonlinearity and also means that crosstalk correction is an irreversible operation which reduces information.

The remainder of this subsection explains how crosstalk correction is implemented using an inner-product layer so it can be integrated into a deep neural network to perform a combination of tasks, e.g., performing a combination of demosaicking and crosstalk correction.

Matrix $\mathbf{W}$ is the weight matrix of the inner-product layer and is a square matrix.[1]

The ideal spectral responses are constructed as a collection of Gaussian responses with a fixed standard deviation and a given mean (the mean of each filter in the mosaic is given by the vendor):

$$\mathbf{Y} = \left[\mathbf{y_{400}}^\top, \mathbf{y_{401}}^\top, \ldots, \mathbf{y_{680}}^\top\right] \tag{34}$$

where the target matrix $\mathbf{Y}$ contains the ideal spectral responses from 400 to 680 nm of the 16 spectral bands in the mosaic.

The weight matrix is trained by stochastic gradient descent (SGD) using the Euclidean loss between the crosstalk-corrected output $\mathrm{CT}(\mathbf{X})$ and the ideal output $\mathbf{Y}$. The crosstalk-calibration set is shown in Fig. 7. The figure shows individual samples on the x-axis and the spectral responses of these samples on the y-axis. Showing from top to bottom: The measured, ideal and corrected response. Mean values of the ideal responses are shown at the bottom of Fig. 7.

The crosstalk-correction matrix $\mathbf{W}$ is normalized after training by multiplying each element by the amount of elements in the main diagonal divided by the trace. This forces the matrix to roughly preserve absolute pixel values:

$$\mathbf{W}_{ij} = \mathbf{W}_{ij} \times \frac{16}{\mathrm{TRACE}(\mathbf{W})}; \forall i, \forall j \tag{35}$$

where i and j are ith row and jth column of matrix $\mathbf{W}$.

If crosstalk correction was perfect the correlations between spectral bands will be eliminated. This means that it is probably more difficult for the upscaling operator to reconstruct the image using spectral correlations. While crosstalk is mostly regarded as detrimental, it could be beneficial to demosaicking. This is further explored in Sect. 5 where also end-to-end training of a crosstalk-corrected image is investigated separately.

5 Experiments

Our main goal is to demosaick images and to minimize crosstalk between spectral bands. All experiments in this section attribute to achieving this goal. Experiments are also specifically designed to gain deeper insight by trying to answer the three research questions presented in Sect. 1.

This section is divided into three main parts: Starting with the effect of crosstalk correction, followed by the good practices of several neural network designs. Finally, we discuss a fully end-to-end trainable convolutional neural network for demosaicking which can process data directly from the raw sensor and produce the final hyperspectral cube.

5.1 The effects of crosstalk correction

The goal of this experiment is to investigate the effect of crosstalk correction on the reconstruction result.

Raw image data from the mosaic sensor is first normalized and converted to a spectral cube by our handcrafted conversion method:

$$\mathbf{C} = \mathrm{MC}_{hc}(\mathrm{NR}(\mathbf{M})) \tag{36}$$

where $\mathbf{M}$ is the 2-d mosaic image and $\mathbf{C}$ is the 3-d hyperspectral cube.

By changing the order in which operators are executed, we investigate how crosstalk effects the final reconstruction result.

$$noCT_{si} = \mathrm{SI}(\mathrm{US}(\mathrm{DS}(\mathbf{C})), \mathbf{C})) \tag{37}$$

$$preCT_{si} = \mathrm{SI}(\mathrm{US}(\mathrm{DS}(\mathrm{CT}(\mathbf{C}))), \mathrm{CT}(\mathbf{C}))) \tag{38}$$

$$postCT_{si} = \mathrm{SI}(\mathrm{CT}(\mathrm{US}(\mathrm{DS}(\mathbf{C}))), \mathrm{CT}(\mathbf{C}))) \tag{39}$$

where $\mathrm{DS}(\cdot)$, $\mathrm{US}(\cdot)$, $\mathrm{CT}(\cdot)$ and $\mathrm{SI}(\cdot)$ are the downsample, upscale, crosstalk-correction and similarity operators, respectively. Outputs of the upscaled versions of the downsampled cubes are compared to either the original spectral cube $\mathbf{C}$ or the crosstalk-corrected spectral cube $\mathrm{CT}(\mathbf{C})$. The metrics $noCT$, $preCT$ and $postCT$ contain are output values of the SSIM index.

Equation 37 performs an upscaling after downsampling to investigate how well upscaling performs without correcting crosstalk. This is used as a baseline in this paper.

Equation 38 first performs a crosstalk-correction before applying downsampling and upscaling. This simulates demosaicking of a mosaic image taken with an MCFA sensor with minimum crosstalk. This will show if crosstalk will actually help demosaicking.

Equation 39 corrects crosstalk after applying downsampling and upscaling. This will show how well crosstalk correction will perform when applied as a separate operator.

In all the cases mentioned here, the crosstalk-correction operator is trained using the method discussed in Sect. 4.8 and is used as a stand-alone operator. Later in this paper, it is explained how the crosstalk-correction operator is integrated

[1] Because the number of spectral bands in the input and output are identical, this is a square matrix; however, the number of output neurons could be less or more than the number of input spectral bands (e.g., map directly to RGB or map to multiple spectral harmonics).

into a neural network which is trained in an end-to-end fashion.

5.2 Demosaicking

The goal of the experiments in this subsection is to determine best practices and to get an intuition for setting several hyperparameters. A number of demosaicking neural network designs will be evaluated. These can be categorized into variations in: model, footprint, image size and image count. Within each configuration of the similarity-framework design, relevant parameters will be varied.

All notations will follow the general upscaling operator defined in Eqs. 22 and 23.

5.2.1 Models

The upscaling operator $\mathrm{US}(\cdot)$ in Eqs. 37, 38 or 39 is implemented as one of six models. The goal here is to determine the optimal capacity of the neural network for demosaicking.

$$\mathrm{US}_{bl}(\mathbf{D}) = \phi\left(\mathbf{D} \oslash_4 \mathcal{F}_{16}^{8x8}\right) \tag{40}$$

$$\mathrm{US}_{bl3d}(\mathbf{D}) = \mathrm{US}_{bl}(\mathbf{D})' \tag{41}$$

$$\mathrm{US}_{4}(\mathbf{D}) = \phi\left(\phi\left(\mathbf{D} \oslash_2 \mathcal{F}_{4}^{4\times4}\right) \oslash_2 \mathcal{F}_{16}^{4\times4}\right) \tag{42}$$

$$\mathrm{US}_{16}(\mathbf{D}) = \phi\left(\phi\left(\mathbf{D} \oslash_2 \mathcal{F}_{16}^{4\times4}\right) \oslash_2 \mathcal{F}_{16}^{4\times4}\right) \tag{43}$$

$$\mathrm{US}_{32}(\mathbf{D}) = \phi\left(\phi\left(\mathbf{D} \oslash_2 \mathcal{F}_{32}^{4\times4}\right) \oslash_2 \mathcal{F}_{16}^{4\times4}\right) \tag{44}$$

$$\mathrm{US}_{256}(\mathbf{D}) = \phi\left(\phi\left(\mathbf{D} \oslash_2 \mathcal{F}_{256}^{4\times4}\right) \oslash_2 \mathcal{F}_{16}^{4\times4}\right) \tag{45}$$

where $\mathbf{D}$ is the downsampled cube before or after crosstalk correction, $\oslash$ is a deconvolution operator with a specific stride, and $\mathcal{F}$ is a convolution filter bank with a specific number of filters of a given size.

The operator $\mathrm{US}_{bl}(\cdot)$ performs upscaling using bilinear interpolation and is used as a reference. $\mathrm{US}_{bl3d}(\cdot)$ is essentially the same as $\mathrm{US}_{bl}(\cdot)$, but the weights of this model are trained. This can be viewed as the best-achievable result using linear upscaling.

The remaining $\mathrm{US}(\cdot)$ operators are nonlinear upscaling models where the number of neurons in the first deconvolution layer are set to either 4, 16, 32 or 256 neurons.

5.2.2 Footprint

The footprint of a (de)convolution layer is related to the size of the filter and determines the spatial context of the input to the filter. As explained in Sect. 4, a larger footprint is expected to better interpolate spatial structures while being less general.

In this experiment, the upscaling operator $\mathrm{US}(\cdot)$ in Eqs. 37, 38 or 39 is implemented as one of three models. The idea is to investigate the effect of the footprint of the convolution filter.

$$\mathrm{US}^{2\times2}(\mathbf{D}) = \phi\left(\phi\left(\mathbf{D} \oslash_2 \mathcal{F}_{32}^{2\times2}\right) \oslash_2 \mathcal{F}_{16}^{2\times2}\right) \tag{46}$$

$$\mathrm{US}^{4\times4}(\mathbf{D}) = \phi\left(\phi\left(\mathbf{D} \oslash_2 \mathcal{F}_{32}^{4\times4}\right) \oslash_2 \mathcal{F}_{16}^{4\times4}\right) \tag{47}$$

$$\mathrm{US}^{8\times8}(\mathbf{D}) = \phi\left(\phi\left(\mathbf{D} \oslash_2 \mathcal{F}_{32}^{8\times8}\right) \oslash_2 \mathcal{F}_{16}^{8\times8}\right) \tag{48}$$

where $\mathbf{D}$ is the downsampled cube before or after crosstalk correction, $\oslash$ is a deconvolution operator with a specific stride and $\mathcal{F}$ is a convolution filter bank with a specific number of filters of a given size.

The operator $\mathrm{US}^{2\times2}(\cdot)$ uses a 2×2 footprint. Because the stride of the convolution is 2, no spatial context is used during upscaling. Therefore this model can only exploit correlations in spectral information. This model is used to investigate the effect of context information (or spatial correlation) of the upscaling operator.

The operator $\mathrm{US}^{4\times4}(\cdot)$ uses a 4×4 footprint. Because of the strided convolution, this actually is a 2×2 spatial context when looking at spectral-pixel neighborhoods with respect to the original downsampled cube $\mathbf{D}$.

The operator $\mathrm{US}^{8\times8}(\cdot)$ uses a 8×8 footprint. Because of the strided convolution, this actually is a 4×4 spatial context in the first deconvolution layer and a 2×2 spatial context in the second deconvolution layer with respect to the original downsampled cube $\mathbf{D}$.

In Fig. 6, the sizes of the convolution filters are shown. Black pixels indicate original spectral pixels from the downsampled image, dark gray spectral pixels are interpolated by the first deconvolution layer and light gray spectral pixels are interpolated by the second deconvolution layer. Note that the number of original, black, spectral pixels that are used by different footprint sizes varies with the size of the convolution filters and also varies depending on the layer of the upscaling operator.

5.2.3 Image size and image count

The proposed similarity maximization framework uses images which are a region of an original image for training. Generally image size and image count will both contribute to the number of training samples. This can be understood by looking at the nature of a convolution. A convolution is generally an independent operation taking only a small spatial context as input. Because no fully connected or inner-product layers are used for demosaicking, the outputs from spatially separated convolutions are never merged. This means that each image patch (equal to the convolution filter footprint) can be viewed as a separate sample. This effect of image count and

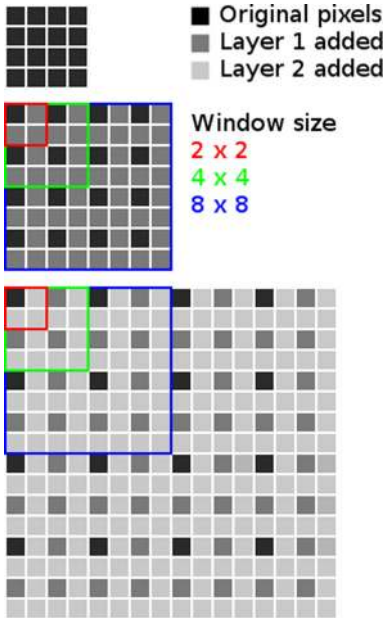

Fig. 6 Upscaling using two deconvolution layers. Original image (top), result of first deconvolution layer with interpolated spectral pixels in dark gray (middle) and final upscaling result with interpolated spectral pixels in light gray (bottom). The red, green and blue delineations indicate convolution filter sizes of 2×2, 4×4 and 8×8 spectral pixels (color figure online)

sample size will be quantified to determine the optimal region size and the number of regions to extract from the original images.

Region sizes will be varied from 1 to 30 spectral pixels with increments of 5 spectral pixels. When the region size is too small, it will probably suffer from border effects. A region size of 1 is included to force the network to not use any spatial context.

Image counts will be varied from 1 through 5 and 100, 500 and 1000 images. The idea is that a certain maximum amount of images is enough for training the upscaling models. Theoretically, increasing the amount of images makes more sense than increasing the size of a region because different images will contain more spatially uncorrelated spectral pixels.

5.3 End-to-end trainable neural network

Prior knowledge about the input mosaic image and the hyperspectral cube in the output can be exploited to train an end-to-end demosaicking deep neural network. In this experiment, all earlier operators are combined.

First the normalized mosaic image $\mathbf{I}$, the downsampled mosaic $\mathbf{M}$, the hyperspectral cube $\mathbf{C}$, and the downsampled cube $\mathbf{D}$ are generated:

$$\mathbf{I} = \mathrm{NR}\left(\mathbf{I}'\right) \tag{49}$$

$$\mathbf{M} = \mathrm{DS}(\mathbf{I}) \tag{50}$$

$$\mathbf{C} = \mathrm{MC}_{hc}(\mathbf{I}) \tag{51}$$

$$\mathbf{D} = \mathrm{MC}_{hc}(\mathbf{M}) \tag{52}$$

where $\mathbf{I}'$ is the raw (unnormalized) input mosaic image, $\mathrm{NR}(\cdot)$ and $\mathrm{DS}(\cdot)$ are the normalization and downsample operators, and $\mathrm{MC}_{hc}(\cdot)$ is the handcrafted conversion operator (to convert from a mosaic image to a hyperspectral cube).

Four deep neural networks are defined by varying two operators. The mosaic-to-cube conversion operator will be varied to either use the handcrafted version $\mathrm{MC}_{hc}(\cdot)$ or the trainable version $\mathrm{MC}_{nn}(\cdot)$. Also the crosstalk-correction operator will either be trained end-to-end or will be applied after upscaling:

$$\mathrm{US} = \mathrm{US}_{32}^{4\times4} \tag{53}$$

$$mc_{hc}/ct_{post} = \mathrm{SI}(\mathrm{CT}(\mathrm{US}(\mathrm{MC}_{hc}(\mathbf{M})))), \mathrm{CT}(\mathbf{C})) \tag{54}$$

$$mc_{nn}/ct_{post} = \mathrm{SI}(\mathrm{CT}(\mathrm{US}(\mathrm{MC}_{nn}(\mathbf{M})))), \mathrm{CT}(\mathbf{C})) \tag{55}$$

$$mc_{hc}/ct_{nn} = \mathrm{SI}(\mathrm{US}(\mathrm{MC}_{hc}(\mathbf{M})), \mathrm{CT}(\mathbf{C})) \tag{56}$$

$$mc_{nn}/ct_{nn} = \mathrm{SI}(\mathrm{US}(\mathrm{MC}_{nn}(\mathbf{M})), \mathrm{CT}(\mathbf{C})) \tag{57}$$

where mc_{hc}/ct_{post} contains the SSIM index when all operator are executed separately. The similarity measurement mc_{nn}/ct_{post} contains the SSIM index when using a trained conversion operator $\mathrm{MC}_{nn}(\cdot)$ and a separate crosstalk-correction operator $\mathrm{CT}(\cdot)$. The similarity measurement mc_{hc}/ct_{nn} is the SSIM index of a model with an integrated crosstalk correction and a handcrafted conversion operator $\mathrm{MC}_{hc}(\cdot)$. Finally mc_{nn}/ct_{nn} contains the SSIM index when all steps are integrated into a single deep neural network.

Furthermore the upscaling operator uses 32 convolution filters in the first layer. Filters have a size of 4×4 pixels. Also note that the similarity operator $\mathrm{SI}(\cdot)$ always calculates the SSIM index using the crosstalk-corrected hyperspectral cube $\mathrm{CT}(\mathbf{C})$.

To show how our final end-to-end trainable convolutional neural network can use mosaic images as an input directly, it can be rewritten in terms of convolutions by expanding all operators:

$$\mathrm{E2E}(\mathbf{M}) = \mathrm{US}_{32}^{4\times4}\left(\mathrm{MC}_{nn}(\mathbf{M})\right) \tag{58}$$

$$= \phi\left(\phi\left(\phi\left(\mathbf{M} \otimes_4 \mathcal{G}_{16}^{9\times9}\right) \oslash_2 \mathcal{F}_{32}^{4\times4}\right) \oslash_2 \mathcal{F}_{16}^{4\times4}\right) \tag{59}$$

where ϕ is the logistic activation function, $\otimes$ is the convolution operator, $\oslash$ is the deconvolution operator, and $\mathcal{G}$ and $\mathcal{F}$ are the filter banks.

6 Results

Special care has been taken to tune the hyperparameters so that they are equal and lead to good performance for all models. All models are trained using 500K epochs, a learning rate of 0.0005 and a momentum of 0.01. We observed that a relatively low learning rate prevents the models from saturating or overflowing the activation functions while the high amount of iterations ensures convergence on this application. This may indicate that the demosaicking problem is convex. No regularization methods like weight decay [35] or dropout [36] are used because over-fitting was not observed in our demosaicking models.

The remainder of this section is divided into four parts. In the first part, the results of the crosstalk-correction operator CT(·) are discussed. In that part also a spectral analysis of the crosstalk-corrected hyperspectral cube is presented. The second part discusses the quantitative results of the upscaling operator US(·) by comparing SSIM index values between original and upscaled images. The third part presents the qualitative analysis with visual details of the output images produced by the upscaling and demosaicking operators. The final part of this section presents a spectral analysis of the results of both the upscaling and the demosaicking operator.

6.1 Crosstalk-correction operator

The results of the crosstalk-correction operator CT(·) are shown in Fig. 7 where the measured graph contains the raw measured spectral responses. The responses in the ideal graph represent the generated ideal Gaussian spectral responses. The corrected graph shows responses after training and applying the crosstalk-correction operator. These results show that the crosstalk in the lower and higher wavelengths has been drastically reduced because the spectral response graphs are less intertwined for the corrected graph.

Generally the peaks of Fig. 7 (corrected) are all of similar height and the energy is conserved between the measured graph and the corrected graph by the normalization of the crosstalk-correction operator formulated in Eq. 35.

Another two interesting observations can be made from the graphs in Fig. 7. Firstly, crosstalk is corrected at the cost of the 496 nm wavelength which is almost completely attenuated (indicated by the red arrow in the ideal graph). Secondly, the optical filter for wavelength 493 also has a major peak at 650 nm which is corrected by the crosstalk correction (indicated by the blue arrows in the measured graph).

The spectral profiles for the average spectral pixel values of some images are shown in Fig. 8. The top row shows the profiles of the original, downsampled, upsampled and demosaicked images before crosstalk correction. The bottom row shows the spectral profiles of the images after crosstalk correction. The average spectral response for all types of images

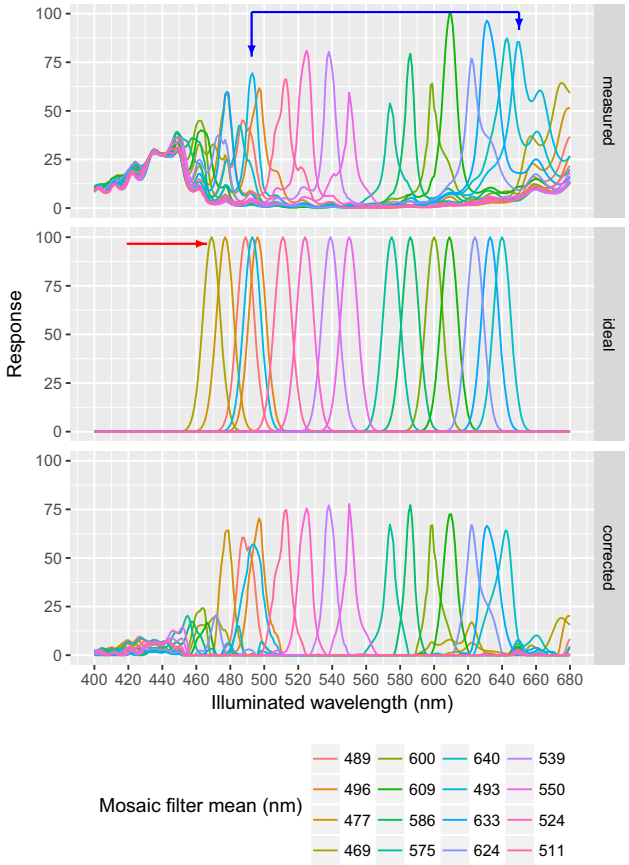

Fig. 7 The measured calibration data of the 4 × 4 mosaic sensor (top), the ideal Gaussian response (middle) and the crosstalk-corrected result (bottom). The illumination wavelength is on the x-axis and the spectral response is on the y-axis. Showing responses for all 16 mosaic filter wavelengths and the vendor provided mean of the spectral responses

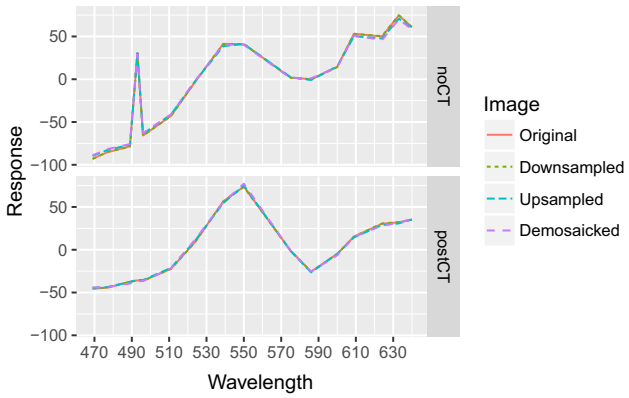

Fig. 8 Average spectral-pixel values of the original, downsampled, upsampled and crosstalk-corrected images before (noCT) and after (postCT) crosstalk correction. The strong peak at wavelength 493 is caused by crosstalk with wavelength 650. The graph has been centered for display by subtracting the mean

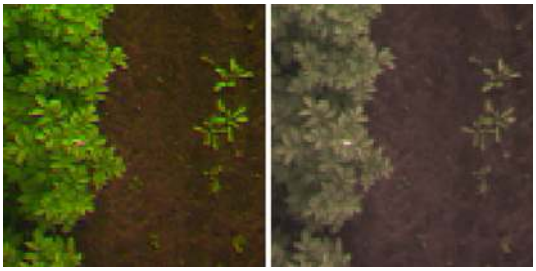

Fig. 9 Left shows a crosstalk-corrected image and right shows an original image. Both images are mapped from 16 spectra to RGB

are almost identical. This means that the relative intensity is preserved between conversions. Furthermore the strong peak in Fig. 8 (noCT) at 493 nm which was most likely caused by crosstalk is corrected in Fig. 8 (postCT). This shows that the crosstalk-correction operator, which has been trained on the calibration dataset, is performing well on the real images. After crosstalk correction, the highest peak is observed at 550 nm. This is known to be the peak reflection wavelength of Chlorophyll [37] and responsible for the green color of vegetation. In Fig. 9 this manifests as a more vivid green color of the image.

The RGB color images in this paper are generated from the 16-channel hyperspectral cube. Our goal with this is to visually interpret differences between demosaicking models and no attempt has been made to generate realistic or even plausible RGB images. Therefore a simple scheme for mapping hyperspectral colors to RGB colors is used. The mean values of the responses of the 469, 477, 489, 493 and 496 nm spectral bands are mapped to blue. The mean values of the responses of the 511, 524, 539 and 550 nm spectral bands are mapped to green. And the mean responses of the spectral bands for wavelengths 575, 586, 624, 633 and 640 nm are mapped to red.

6.2 Quantitative analysis

Quantitative results are produced by comparing the original hyperspectral cubes with the upscaled cubes by analyzing the structural similarity (SSIM) index, starting with the models, then discussing the footprint, then the image size and the image count. The results are presented in Table 2. The noCT column of Table 2 shows the performance of upscaling without crosstalk correction and serves as a reference. The preCT column shows the performance with crosstalk corrected prior to upscaling, and the postCT column shows the performance with crosstalk corrected after upscaling.

Finally, the results for upscaling with our end-to-end convolutional neural networks are discussed.

Table 2 Median SSIM for upscaling using various models and inputs

	noCT	preCT	postCT
Model			
BL	0.48	0.63	0.55
BL3D	0.88	0.70	0.84
16	0.89	0.80	0.85
32	0.89	0.81	0.85
256	0.89	0.80	0.85
Footprint			
2×2	0.87	0.75	0.82
4×4	0.89	0.81	0.85
8×8	**0.90**	0.81	**0.86**
Images			
1	0.81	0.72	0.75
2	0.85	0.77	0.80
5	0.87	0.78	0.82
100	0.89	0.81	0.85
1000	0.89	0.81	0.85
Size			
1	0.70	0.58	0.63
10	0.89	0.78	0.85
20	0.89	0.81	0.85
30	0.89	0.81	0.85

The bold result in the noCT column indicates the best median SSIM when not correcting crosstalk. The bold result in the postCT indicates the best median SSIM when crosstalk correction is applied. The column noCT serves as a reference where no crosstalk correction is applied. In preCT and postCT crosstalk correction is applied before or after upscaling respectively

6.2.1 Models

Standard upscaling with bilinear interpolation (BL) is compared to the linear upscaling (BL3D) and nonlinear upscaling models (model 16, 32 and 256) that have been defined in Sect. 5.2.1. The results that are discussed here are indicated by 'Model' in Table 2.

The BL3D model is the same as the BL (Bilinear Interpolation) model, with the exception that weights are trained. Interestingly this BL3D model is almost as accurate as nonlinear upscaling when crosstalk correction is applied after upscaling (postCT) but falls short when crosstalk is corrected before upscaling (preCT). This suggests that more complex models are needed to upscale images with less crosstalk.

The median similarity increases from 0.55 to 0.85 when comparing the bilinear model to the nonlinear models (see column postCT). It is also shown that increasing the number of convolution filters in the initial upscaling layer does not need to exceed 16 filters, the SSIM index stays at 0.85.

The overall best result is achieved when not applying crosstalk correction at all (noCT column, SSIM 0.89). This

is probably explained by the fact that crosstalk correction is an operator which reduces information. Regardless of the trained model, applying crosstalk correction after upscaling outperforms crosstalk correction before upscaling. This supports the hypothesis that demosaicking benefits from crosstalk.

6.2.2 Footprint

The results for using various different footprints for the convolution filters are shown in Table 2 and are indicated by 'Footprint'. These footprint sizes are measured in terms of the spectral cube not the mosaic image, e.g., the conversion operator MC(·) has already been applied.

The largest improvement is achieved when going from a 2×2 footprint to a 4×4 footprint. Although the results increase asymptotically, the results for the 8×8 filter still improves (SSIM 0.86) because also the information of the original spectral pixels is exploited in the final upscaling layer (explained earlier in Fig. 6).

The highest SSIM index observed in this paper is 0.90 and is achieved when performing upscaling without applying crosstalk correction. This shows excellent baseline performance for our nonlinear upscaling models.

6.2.3 Image size and image count

Two methods for increasing the training set size are either to increase the number of training images or to increase the size of the training images (explained in Sect. 5).

The results in Table 2 indicated by 'Images' show the SSIM index for increasing the number of training images. Interestingly, when only one training image is used, already quite good results are achieved (the SSIM index is higher than 0.7). This is probably because one training image already contains a lot of information about the spectral/spatial correlations. By further increasing the amount of training images the results keep improving. However increasing beyond 100 training images does not seem to further improve the results.

The results in Table 2 indicated by 'Size' show the SSIM index for using different training image sizes. An image size of one (basically a vector of 16 spectral intensity values) performs poorly because the upscaling operator is only able to exploit spectral information to reconstruct spatial information. Increasing the size of the training images leads to an increased performance because more spatial information can be exploited to spatially interpolate pixels. Increasing the size of the training image beyond 20 pixels seems to not further improve the result. Interestingly, when upscaling images with minimized crosstalk (the preCT column), image size seems to matter more. This can be explained by the fact that for these images the upscaling operator cannot exploit spectral correlations and needs to rely more on spatial information for a valid reconstruction.

Table 3 The median SSIM for upscaling with crosstalk correction and mosaic-to-cube conversion trained into an end-to-end network or applied separately. Results shown for training 1000 images of size 20

	CT_{post}	CT_{nn}
MC_{hc}	0.85	0.84
MC_{nn}	0.86	0.85

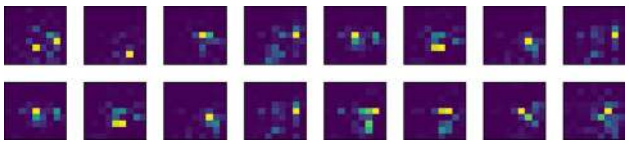

Fig. 10 The 81 weights of each of the 16 convolution filters for the learned mosaic-to-cube operator (MC_{nn}). The bright-yellow pixels indicate large weights and dark-blue values indicate small weights (color figure online)

6.2.4 End-to-end

This final section of the quantitative analysis shows the results when comparing different degrees of end-to-end deep neural networks. The crosstalk-correction operator is either applied after upscaling indicated by $\mathrm{CT}_{post}(\cdot)$ or the crosstalk-correction operator is trained directly into the network indicated by $\mathrm{CT}_{nn}(\cdot)$. Also the mosaic-to-cube conversion is either applied separately in a handcrafted manner with the $\mathrm{MC}_{hc}(\cdot)$ operator or is trained as an extra convolution layer into the neural network with $\mathrm{MC}_{nn}(\cdot)$. The combination of $\mathrm{MC}_{nn}(\cdot)$ and $\mathrm{CT}_{nn}(\cdot)$ operators represent the end-to-end trainable deep neural network for demosaicking which is regarded as the final goal.

Table 3 shows the results of these networks. The median SSIM index for the end-to-end network and the SSIM index when using separated operations are identical (0.85). This means that a neural network is good at solving all operations with one completely integrated model. When applying crosstalk as a separate operator, a slightly better result is achieved (0.86).

The trainable mosaic-to-cube operator MC_{nn} that was introduced in Sect. 4.2 is designed to specialize in converting the image mosaic to a spectral cube by specifying a convolution stride of 4. Each of the 16 convolution filters could learn to select a different pixel from the image mosaic to mimic the handcrafted mosaic-to-cube operator. In Fig. 10, the weights of the 16, 9×9 convolution filters are shown as they have been learned by the end-to-end neural network. As expected each filter specializes in selecting a different, mostly unique part, of the image mosaic. Although the filter size is 9×9, only large weight values for a 4×4 sub matrix are present in the lower-right part of each filter. This is probably due

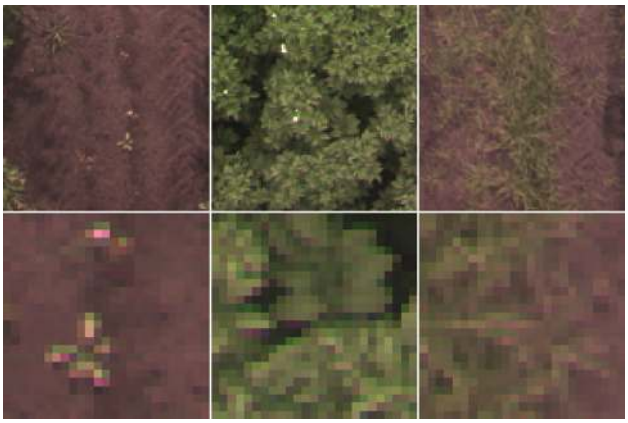

Fig. 11 Images for evaluating upscaling (top row) and demosaicking (bottom row). Soil (left), plants (middle) and grass (right)

to the 4×4 image mosaic and indicates that a filter size of 9×9 is probably not required for the trained mosaic-to-cube operator.

6.3 Visual analysis

Further insight can be gained by visually analyzing the differences between images.[2] This gives an intuition about which SSIM value differences are still perceivable and is the main method for evaluating the demosaicking operator. For this analysis, three images have been selected.

The images used for validating the upscaling operator are shown in the top row of Fig. 11. These images have been downsampled by a factor 16 by the DS$(\cdot)$ operator. The images used for the final demosaicking of an image mosaic are shown in the bottom row of Fig. 11 and have the original spatial resolution (and 16 channels). The left image contains a patch of soil and is used to evaluate the performance on flat surfaces. The middle image contains plants to analyze the performance on images with sharp edges. The right image contains grass which demonstrates performance on small structures. Analyzing these images should give a fair judgment of the performance of our method for different types of images that can be encountered in vegetation inspection with UAVs.

All the images in this subsection are presented in a similar fashion. When visual results of the upscaling models are presented, the first two columns contain the original and downsampled images (Orig and DS) and the rest of the columns contain results for various models, footprints, training images sizes or training image counts. When presenting the results of demosaicking, the downsampled image column is not present because it is not used for demosaicking. The rows of the images can indicate either noCT, preCT or postCT, where noCT shows images without crosstalk correction applied, preCT shows images where crosstalk correction is applied prior to upscaling and postCT shows images where crosstalk correction is applied after upscaling.

The remainder of this subsection discusses the visual results of upscaling and demosaicking using the various models, footprints, images sizes and image counts, as well as the difference in result when applying crosstalk correction either, not, before or after upscaling.

6.3.1 Models

In Fig. 12, it be can clearly seen that the crosstalk-corrected images appear more vivid green because colors are less intermixed. When upscaling the image after applying crosstalk correction (preCT) the resulting images appear slightly more blurry. This shows visually that crosstalk helps upscaling.

In Fig. 12, it is shown that bilinear interpolation (BL) results in a blurry image. The sharpest upscaling result of the potato plant images in that figure is achieved using 32 convolution filters in the first upscaling layer. The images of the soil show an increase in color accuracy when using more convolution filters in the first upscaling layer. The greenish haze in the soil images is least visible when using 16 or more filters and applying crosstalk correction after upscaling (postCT). These visual observations are confirmed by the SSIM of 0.84 for the leaves and 0.79 for the soil patches (when also correcting crosstalk).

Figure 13 shows the results when demosaicking the original images. Here it is shown visually that more structure appears in the image objects like the leaves and the small plant in the soil image. This means that the upscaling operator not only performs well on reconstructing images but also achieves good results when demosaicking the images beyond their original resolution. If crosstalk correction is applied after demosaicking, the results are better. This manifests as a smoother upscaling result for the images containing leaves, and this manifests as a strong reduction in chromatic aberrations alongside strong edges in the soil images. The striped background pattern in the bottom row of Fig. 13 keeps diminishing when adding more convolution filters (up to 256).

6.3.2 Image size and image count

The top row of Fig. 14 shows that applying crosstalk correction before upscaling and using a 1×1 pixel hyperspectral training image does not yield any result, just a green image. As crosstalk has been corrected, the model can use neither spectral correlations to reconstruct the image nor spatial correlations because the training images are just one spectral pixel. If the size of the training images is subsequently increased to 5×5 and 10×10 the reconstruction sharpness increases because spatial correlations can be exploited.

[2] This part of the paper is best read on a screen.

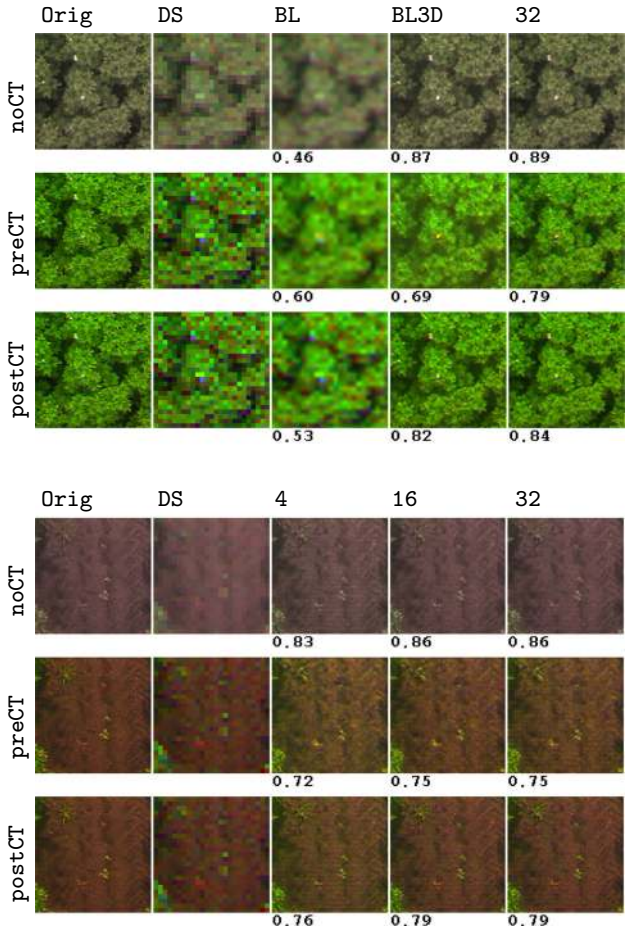

Fig. 12 Upscaled images using different models. Using 32 filters and applying crosstalk correction after upscaling (postCT) shows the sharpest result on the potato leaf images and the best color reconstruction on the soil images. Images with crosstalk correction (preCT and postCT) appear more vivid green (color figure online)

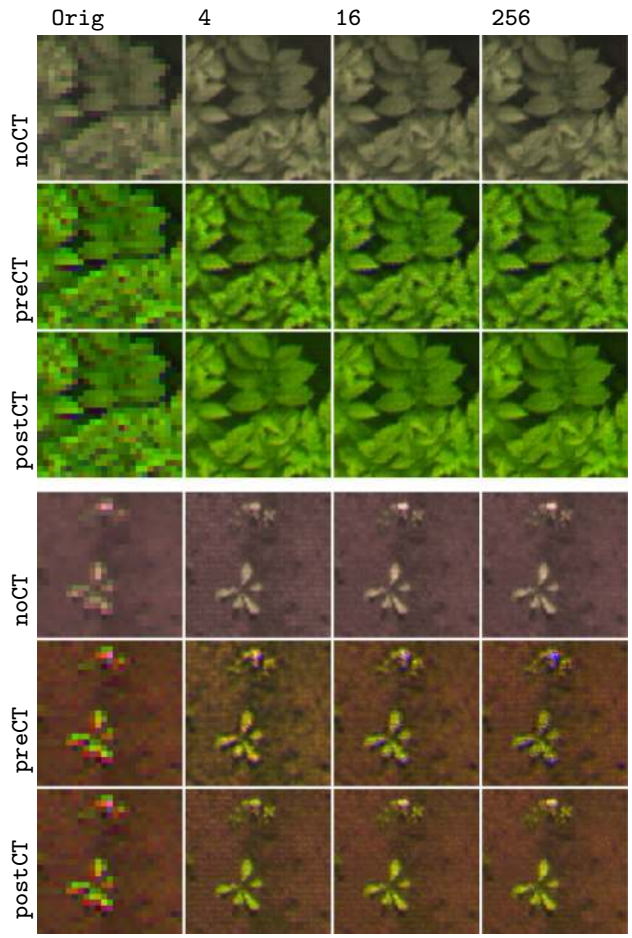

Fig. 13 Demosaicked images using different models. Using 16 or more filters and applying crosstalk after upscaling (postCT) achieves the results with the least noise in the leaf images and the least chromatic aberrations in the soil images

But applying crosstalk after upscaling (the middle row of Fig. 14) shows the best results with training image sizes of 5×5 and larger. The bottom row of Fig. 14 shows that incrementally adding more images to the training set results in higher SSIM index values. However, an SSIM of 0.82 still does not yield satisfying results (there is still a color haze). A slight increase of only 0.02 still represents a great improvement at 1000 training images. This shows that small SSIM differences could still represent visual improvements.

6.3.3 Footprint

The footprint of the convolution filter seems to only marginally affect the upscaling result in Fig. 15 beyond a filter size of 4×4. When the filter is 2×2, the model fails to capture spatial relations and severe striped artifacts are produced which is probably a result of the underlying mosaic pattern.

Interestingly, when applying the trained model for demosaicking, a visual improvement can still be perceived for filters with a footprint of 8×8 (see Fig. 16, bottom row). This shows that an SSIM increase of 0.01 can still represent visual improvements of the striped pattern when looking at the result of the demosaic operator. This also means that a model which has been trained to perform upscaling also generalizes well to perform hyperspectral demosaicking.

6.3.4 End-to-end

The visual results for the various end-to-end models are shown in Fig. 17. This shows a collection of potato leaves, with Orig showing the source image. The columns MC_{hc} and MC_{nn} show demosaicking results of the handcrafted and trained mosaic-to-cube operators. The rows CT_{post} and CT_{nn} show results when crosstalk correction is applied after demosaicking and when crosstalk correction is trained directly into the model. The main conclusion that can be drawn is

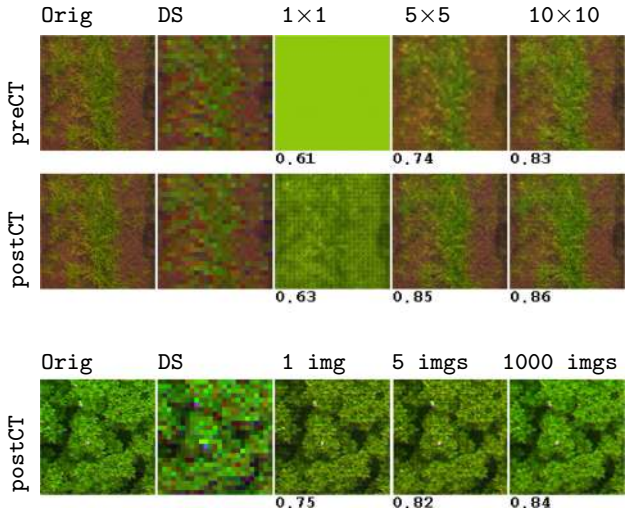

Fig. 14 Upscaled images with different training image sizes (top two rows) and training image counts (bottom row). The model cannot reconstruct the image if the training image size is a 1×1 spectral pixel and when crosstalk is minimized before upscaling. The best result is achieved with large training images (10×10) or many training images (1000 images)

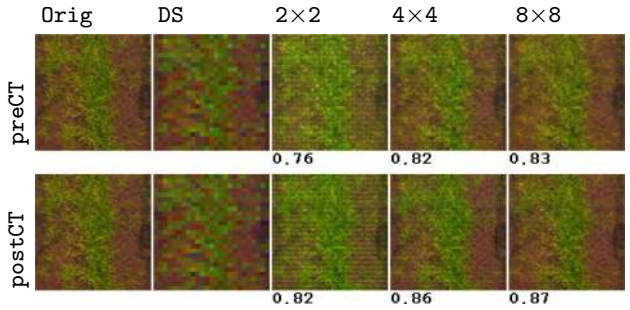

Fig. 15 Upscaled images with different convolution filter footprint sizes. The small details of the grass are clearer when correcting the crosstalk after upscaling. Striped artifacts appear when using a small footprint

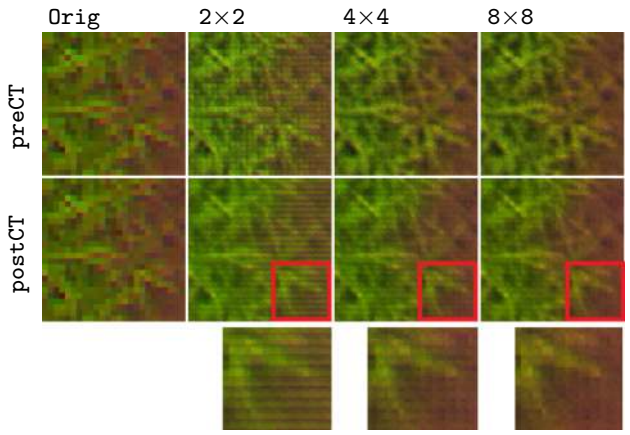

Fig. 16 Demosaicked images with different convolution filter footprint sizes. The striped artifacts diminish with increasing footprint sizes, and this effect is still visible for the 8×8 footprint size (see enlarged square)

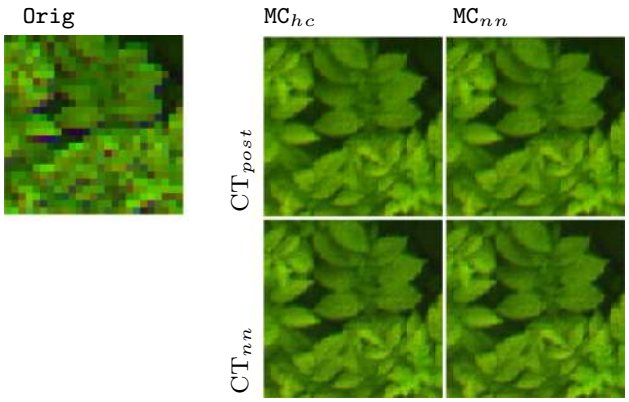

Fig. 17 Collection of demosaicked images of potato leaves with source image (Orig). No noticeable differences between handcrafted mosaic-to-cube (MC$_{hc}$), trained mosaic-to-cube (MC$_{nn}$), post-processed crosstalk correction (CT$_{post}$) and crosstalk correction trained into the model (CT$_{nn}$). The bottom-right image shows the demosaicking result when the network is trained end-to-end

that the end-to-end trained model (bottom-right corner of Fig. 17) shows no noticeable differences in the final result. This in turn means that an end-to-end solution for hyperspectral demosaicking and simultaneous crosstalk correction can be achieved with our similarity maximization framework and a convolutional neural network.

6.4 Spectral analysis

In Sect. 6.1, a spectral analysis of the crosstalk-correction operator showed that crosstalk is actually corrected in the images and has shown why crosstalk-corrected images appear more vivid. In Sect. 6.2, the reconstruction by the upscaling operator has been analyzed using the full-reference SSIM index on the complete spectral cube. This has given an indication of the spectral reconstruction. Further visual analysis has been provided in Sect. 6.3, where an RGB mapping of all original spectral bands was used to visually show the upscaled and demosaicked results of the underlying spectral cubes. In this subsection, an additional spectral analysis is provided to show the effect of the upscaling and demosaicking operator in the spectral domain.

In Fig. 18, the spectral results are shown for the upscaling operator. The top image contains the original RGB mapping. The spectral graphs of the dotted line are provided in the subsequent images where the y-axis indicates the spectral domain with the 16 spectra (ordered from low to high wavelengths, from top to bottom). Each pixel in these images is the crosstalk-corrected intensity for a specific spectral frequency at the dotted line. The images with captions Hyperspectral Original and Hyperspectral Downsampled show the spectral graphs of the original images and the image produced by the downsampling operator DS$(\cdot)$. From this downsampled image, the Hyperspectral Upscaled graphs are provided

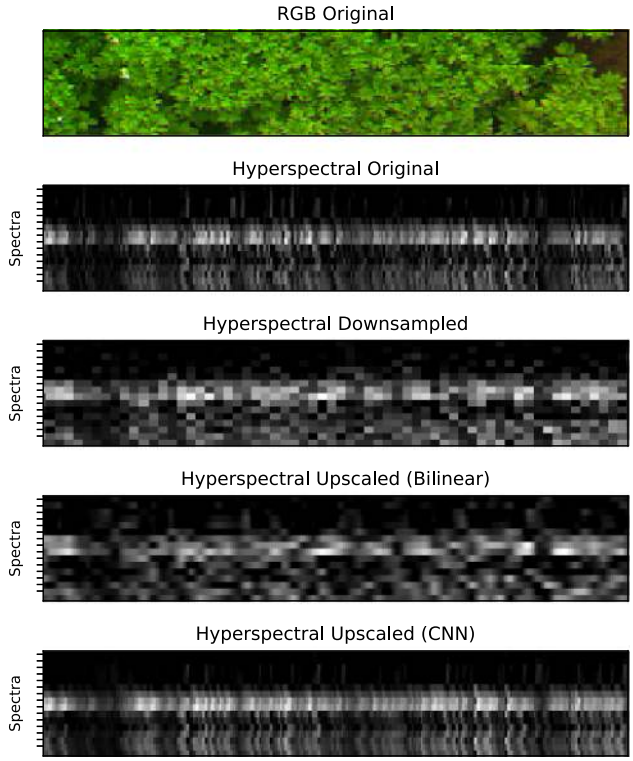

Fig. 18 Spectral graphs for the upscaling result. The top image shows the RGB mapping of the original image and the second image shows the spectral profile of the dotted line of the original image. The bottom graph shows that a good reconstruction is achieved with our convolutional neural network

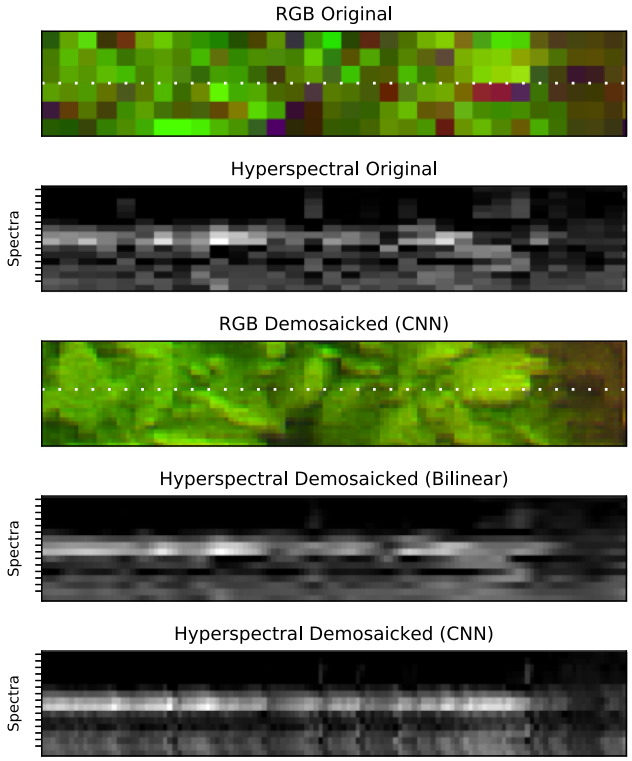

Fig. 19 Spectral graphs for the demosaicking result. The top image shows the RGB mapping of the original image and the second image shows the spectral profile of the dotted line of the original image. The bottom two graphs show that demosaicking with our convolutional neural network uncovers more details compared to demosaicking using bilinear interpolation

for both the bilinear model as well as for our convolutional (CNN) model. Here it is shown that the CNN model provides a more detailed reconstruction for the upscaled result compared to the bilinear model. This also shows that the upscaling operator interpolates spatial structures for each spectral band. It does not actually interpolate spectral information as the number of spectral bands in the downsampled image and the upscaled image are both 16 while the spatial resolution is increased by a factor 4 in one direction.

In Fig. 19, the spectral results for demosaicking using the upscaling operator are shown to investigate if the upscaling operator generalizes to provide an apt demosaicking result. The image with caption: RGB Original and RGB Demosaicked (CNN) show the RGB mapping of the original and the demosaicked image, respectively. The image with caption Hyperspectral Original shows the response of each of the 16 spectral frequencies on the dotted line in the original images. In the bottom two images of Fig. 19, it is clearly shown that the demosaicking process using our convolutional model uncovers more detailed structures in the spectral domain compared to bilinear interpolation. While no full-reference quantitative comparison has been made for demosaicking, the facts that relevant spatial structures like the leaves appear in the result and that additional spectral structures are uncovered provide arguments that our upscaling operator which has been trained on downsampled images actually generalized well to performing demosaicking.

7 Discussion and conclusion

This paper has presented an end-to-end trainable method for demosaicking and simultaneous crosstalk correction of images taken with a hyperspectral mosaic sensor, based on deep learning. All experiment have been performed with an image mosaic of 4×4 but our similarity framework can easily be adjusted to incorporate larger sensor mosaics. A general rule of thumb is that the dimension of the mosaic should not be a prime number so that two deconvolution (upscaling) layers can be used to introduce nonlinearity.

The quantitative and qualitative analyses show that our similarity maximization framework for demosaicking outperforms standard bilinear interpolation or Bayer demosaicking. Even when directly plugging bilinear interpolation into our framework and training the convolutional filters, a good result is achieved. By increasing the number of layers and adding nonlinearity, the demosaicking results can be further

improved to achieve a median structural similarity (SSIM) index of 0.86 between original and upscaled images. When just using bilinear interpolation, an SSIM index of only 0.55 is achieved.

In the introduction of this paper, three research questions were introduced. We will now reflect on these questions. At the end of this section, we provide additional arguments why these upscaling models can directly be used for demosaicking.

1. How much does hyperspectral demosaicking benefit from spectral and spatial correlations?

The experimental results show that upscaling benefits from spectral correlations. This has been shown by the fact that upscaling after crosstalk correction consistently produces inferior results compared to upscaling before crosstalk correction.

The results also show that without crosstalk correction the reconstruction results are better (SSIM is 0.90). This is most likely explained by the fact that crosstalk correction is destructive and irreversible.

By forcing the models to train with samples of single spectral pixels (effectively disabling exploitation of spatial correlations), the results have shown that without crosstalk the upscaling cannot be trained. When only using spatial information the upscaling performs reasonably well. A combination of spectral and spatial information in the training data shows the best results.

2. What are good practices for designing hyperspectral demosaicking neural networks?

Good practices in relation to models, image size, image count and convolution footprint have been investigated. Our proposed nonlinear upscaling models show the best results compared to bilinear interpolation and trainable linear upscaling. A two-layer upscaling model with 16 convolutional filters of size 4×4 seems to be sufficient when training with 100 images of size 20×20 pixels. An SSIM index difference of 0.02 seems significant when visually perceiving the result of the upscaling operator.

The final demosaicking results further improve when using an 8×8 convolution filter for the first upscaling layer. So when the results of the upscaling operators in the similarity framework do not seem to improve much the demosaicking results are still affected and an SSIM difference of 0.01 is still visually perceivable after demosaicking.

3. How well can hyperspectral demosaicking sub-tasks be integrated for end-to-end training?

The acclaimed power of convolutional neural networks is the ability to learn problems end-to-end. Our results show that a custom-designed deep learning model can be trained to directly take a raw mosaic image and produce a crosstalk-corrected and full-resolution demosaicked hyperspectral cube. We also observed that by design the first layer of this network specializes in converting the raw image mosaic to an initial spectral cube.

We have chosen a similarity maximization method which shares many properties with single image super resolution (SISR) methods. A point of discussion is whether this method of using downsampling and upscaling to train a network for demosaicking (upscaling beyond the original image resolution) is the correct approach. Could the upscaling models not just be learning to invert the downscaling operator? Would a comparison with a ground-truth for quantitatively validating the demosaicking results not be a better approach? Our approach deals with a practical situation of a UAV applied in precision agriculture. It would be very difficult to produce accurate additional ground-truth images using a multi-camera-single-shot or a single-camera-multi-shot system. The pixel-precise alignment needed for validation would be virtually impossible because of moving objects and parallax errors as noted in the introduction.

While a ground-truth could help, we argue that such a setup is not necessary for our approach due to the following reasons. The downsampling operator has been carefully and specifically designed to retain the spectral and spatial information in the same way that the actual raw mosaic image contains this information. This helps to generalize the upscaling operator to perform demosaicking. The demosaicked images show recognizable reconstructed image objects like leaves and plants and also additional spectral structures are uncovered which confirms the generalizing behavior of the upscaling operator. Because the convolution filters are very small, the types of features that they respond to are limited to basic image features like edges, corners, etc. This prevents the upscaling operator from mistakenly learning large object structures, specific to the downsampled image, like complete leaves or other macro-scale objects. Finally, the size of the image mosaic (4×4) is identical for the downsampled and original images which means that the same trained crosstalk-correction operator is applicable for both upscaling as well as for demosaicking.

8 Future work

Performing crosstalk correction has several advantages for future research. When the signal is untangled, a multivariate spectral analysis of the data could be used to identify important spectral bands. Also with an untangled signal, it is easier to compare spectral outputs to theoretical spectral profiles (for example, the peak reflection wavelength of Chlorophyll). Crosstalk correction and demosaicking mostly reorder and

augment information to a format which better represents the physical world. Future research could focus on the benefits of this representation for disease detection and for measuring soil nutrient concentrations.

In this research, a 4×4 mosaic sensor was used. Future research could apply the proposed similarity framework to other types of mosaic sensor configurations. For example to 3×3 or 5×5 mosaic sensors available from various vendors. Further research could combine single image super resolution (SISR) to demosaic and upscale images beyond the spatial resolution of the original sensor. This could represent a combination of these two fields in the form of hyperspectral single image super resolution (HSISR).

These mosaic sensors seem ideally suited for utilization on UAVs because of their low weight and small size. Our framework could be used to investigate other agricultural applications like classifying diseases, counting and classifying crops and determining soil properties. While this paper focused mainly on precision agriculture applications with unmanned aerial vehicles (UAVs), future research could extend these experiments to a multitude of other applications where hyperspectral mosaic sensors are used. For example: medical imaging, environmental monitoring, food inspection, etc.

References

1. Paredes, J.A., Gonzalez, J., Saito, C., Flores, A.: Multispectral imaging system with UAV integration capabilities for crop analysis. In: 2017 First IEEE International Symposium of Geoscience and Remote Sensing (GRSS-CHILE), pp. 1–4 (2017)
2. van de Loosdrecht, J., Dijkstra, K., Postma, J.H., Keuning, W., Bruin, D.: Twirre: architecture for autonomous mini-UAVs using interchangeable commodity components. In: International Micro Air Vehicle Conference and Competition (2014)
3. Dijkstra, K., van de Loosdrecht, J., Schomaker, L.R.B., Wiering, M.A.: Hyper-spectral frequency selection for the classification of vegetation diseases. In: European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, Bruges (2017)
4. Rebetez, J., Satizabal, H.F., Mota, M., Noll, D., Buchi, L., Wendling, M., Cannelle, B., Perez-Uribe, A., Burgos, S.: Augmenting a convolutional neural network with local histograms—a case study in crop classification from high-resolution UAV imagery. In: ESANN 2016 Proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (2016)
5. Berra, E.F., Gaulton, R., Barr, S.: Commercial off-the-shelf digital cameras on unmanned aerial vehicles for multitemporal monitoring of vegetation reflectance and NDVI. IEEE Trans. Geosci. Remote Sens. **55**(9), 4878–4886 (2017)
6. Pullanagari, R.R., Kereszturi, G., Yule, I.J.: Mapping of macro and micro nutrients of mixed pastures using airborne AisaFENIX hyperspectral imagery. ISPRS J. Photogramm. Remote Sens. **117**, 1–10 (2016)
7. Wang, T., Celik, K., Somani, A.K.: Characterization of mountain drainage patterns for GPS-denied UAS navigation augmentation. Mach. Vis. Appl. **27**(1), 87–101 (2016)
8. De Boer, J., Barbany, M.J., Dijkstra, M.R., Dijkstra, K., van de Loosdrecht, J.: Twirre V2: evolution of an architecture for automated mini-UAVs using interchangeable commodity components. In: International Micro Air Vehicle Conference and Competition (2015)
9. Monno, Y., Tanaka, M., Okutomi, M.: Multispectral demosaicking using guided filter. IS&T/SPIE Electronic. Imaging **8299**, 82990O (2012). https://doi.org/10.1117/12.906168
10. Mustaniemi, J., Kannala, J., Heikkilä, J.: Parallax correction via disparity estimation in a multi-aperture camera. Mach. Vis. Appl. **27**(8), 1313–1323 (2016)
11. Behmann, J., Mahlein, A.K., Paulus, S., Dupuis, J., Kuhlmann, H., Oerke, E.C., Plümer, L.: Generation and application of hyperspectral 3D plant models: methods and challenges. Mach. Vis. Appl. **27**(5), 611–624 (2016)
12. Bayer, B.E.: Color imaging array. U.S. Patent 3971 065, p. 10 (1976)
13. Geelen, B., Blanch, C., Gonzalez, P., Tack, N., Lambrechts, A.: A tiny VIS-NIR snapshot multispectral camera. In: von Freymann, G., Schoenfeld, W.V., Rumpf, R.C., Helvajian, H. (eds.) Advanced Fabrication Technologies for Micro/Nano Optics and Photonics, vol. 9374. International Society for Optics and Photonics, Bellingham (2015)
14. Hirakawa, K.: Cross-talk explained. In: Proceedings—International Conference on Image Processing, ICIP, pp. 677–680. IEEE (2008)
15. Keren, D., Osadchy, M.: Restoring subsampled color images. Mach. Vis. Appl. **11**(4), 197–202 (1999)
16. Wang, D., Yu, G., Zhou, X., Wang, C.: Image demosaicking for Bayer-patterned CFA images using improved linear interpolation. In: 2017 Seventh International Conference on Information Science and Technology (ICIST), pp. 464–469. IEEE (2017)
17. Wang, Y.Q.: A multilayer neural network for image demosaicking. In: 2014 IEEE International Conference on Image Processing, ICIP 2014, pp. 1852–1856 (2014)
18. Degraux, K., Cambareri, V., Jacques, L., Geelen, B., Blanch, C., Lafruit, G.: Generalized inpainting method for hyperspectral image acquisition. In: Proceedings—International Conference on Image Processing, ICIP, pp. 315–319 (2015)
19. Aggarwal, H.K., Majumdar, A.: Single-sensor multi-spectral image demosaicing algorithm using learned interpolation weights. In: International Geoscience and Remote Sensing Symposium (IGARSS), pp. 2011–2014. IEEE (2014)
20. Gharbi, M., Chaurasia, G., Paris, S., Durand, F.: Deep joint demosaicking and denoising. ACM Trans. Graph. **35**(6), 1–12 (2016)
21. Peng, J., Hon, B.Y.C., Kong, D.: A structural low rank regularization method for single image super-resolution. Mach. Vis. Appl. **26**(7–8), 991–1005 (2015)
22. Dong, C., Loy, C.C., He, K., Tang, X.: Image super-resolution using deep convolutional networks. IEEE Trans. Pattern Anal. Mach. Intell. **38**(2), 295–307 (2016)
23. Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: DeepLab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. In: International Conference on Learning Representations (2016)
24. Eichhardt, I., Chetverikov, D., Jankó, Z.: Image-guided ToF depth upsampling: a survey. Mach. Vis. Appl. **28**(3–4), 267–282 (2017)
25. Mihoubi, S., Losson, O., Mathon, B., Macaire, L.: Multispectral demosaicing using intensity-based spectral correlation. In: 5th International Conference on Image Processing, Theory, Tools and Applications 2015, IPTA 2015, pp. 461–466. IEEE (2015)
26. Lin, H.W., Tegmark, M., Rolnick, D.: Why does deep and cheap learning work so well? J. Stat. Phys. **168**(6), 1223–1247 (2017)
27. Al-Waisy, A.S., Qahwaji, R., Ipson, S., et al.: A multimodal deep learning framework using local feature representations for face

recognition. Mach. Vis. Appl. **29**, 35 (2018). https://doi.org/10.1007/s00138-017-0870-2
28. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: convolutional architecture for fast feature embedding. arXiv preprint arXiv:1408.5093 (2014)
29. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. **13**(4), 600–612 (2004)
30. Galteri, L., Seidenari, L., Bertini, M., Del Bimbo, A.: Deep generative adversarial compression artifact removal. In: International Conference on Computer Vision (2017)
31. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, Cambridge (2016)
32. Goyal, P., Dollár, P., Girshick, R., Noordhuis, P., Wesolowski, L., Kyrola, A., Tulloch, A., Jia, Y., He, K.: Accurate, large minibatch SGD: training ImageNet in 1 h. arXiv preprint arXiv:1706.02677 (2017)
33. Sauget, V., Hubert, Faiola, A., Tisserand, S.: Application note for CMS camera and CMS sensor users: Post-processing method for crosstalk reduction in multispectral data (2016). https://docs.wixstatic.com/ugd/153fe5_3617a87460ea401a8c0c2a0c04f0443a.pdf
34. Shelhamer, E., Long, J., Darrell, T.: Fully convolutional networks for semantic segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **39**(4), 640–651 (2017)
35. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Pereira, F., Burges, C.J.C., Bottou, L., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems, vol. 25, pp. 1097–1105. Curran Associates Inc., Red Hook (2012)
36. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**, 1929–1958 (2014)
37. Thomas, J.R., Gausman, H.W.: Leaf reflectance versus leaf chlorophyll and carotenoid concentrations for eight crops. Agron. J. **69**(5), 799 (1977)