Micro-lens image stack upsampling for hyperspectral light fields

Paul Hasenbusch

October 21, 2025

Contents

1 Models

1.1 Prelimanries

1.1.1 Neural Networks

Definition 1 (Fully Connected Layer) *Let $A \in \mathbb{R}^{n \times m}$ and $b \in \mathbb{R}^m$. We call the mapping*

$$F : \mathbb{R}^n \to \mathbb{R}^m \ , \quad F(x) = Ax + b \ ,$$

a fully connected layer.

In order to refer to the architecture, thats is a fully connected layer with input dimension $n \in \mathbb{N}$ and output dimension $m \in \mathbb{N}$, whose weights are not fixed but subject to optimization, we write $F(n, m)$.

1.1.2 Convolutional Layers

Definition 2 (Convolution over multiple feature maps) *Let $X \in \mathbb{R}^{C \times n_1 \times ... \times n_d}$ and $k \in \mathbb{R}^{C \times d_1 \times ... \times d_d}$. The convolution of X with the kernel k, denoted by $k * X$ is given by*

$$(k * X)(p_0) = \sum_{i=1}^{c} \sum_{p \in R} k(c, p) X(c, p_0 + p) \ ,$$

where $R = \prod_{i=1}^{d} [0, d_i] \cap \mathbb{N}$, for all $p_0 \in \prod_{i=1}^{d} \mathbb{N} \cap [1, n_i - d_i]$.

Definition 3 (Convolutional Layer) *Let $X \in \mathbb{R}^{C \times n_1 \times ... \times n_d}$, $k_1, ..., k_{C'} \in \mathbb{R}^{C \times d_1 \times ... \times d_d}$ and $b \in \mathbb{R}^{C'}$. We call the mapping*

$$C : \mathbb{R}^{C \times n_1 \times ... \times n_d} \to \mathbb{R}^{C' \times n_1 \times ... \times n_d} \ , \quad C(X) = [k_1 * X, ..., k_{C'} * X] + b \ ,$$

a convolutional layer.

In order to refer to the architecture, thats is a convolutional layer with input channel dimension $C \in \mathbb{N}$ and output channel dimension $C' \in \mathbb{N}$, kernels $k_1, ..., k_{C'} \in \mathbb{R}^{C \times d_1 \times ... \times d_d}$ and $b \in \mathbb{R}^{C'}$, which are not fixed but subject to optimization, we write

$$C(n, m, \text{kernel-size} = (d_1, ..., d_d), \text{padding} = p, \text{padding-mode} = m) \ .$$

Definition 4 (Residual Connection) *Let X be some set, $F = \{f : X \to X | f \, \text{is a function}\}$ be the set of functions on X mapping back on X. The operation*

$$R : F \to F \ , \quad R(f)(x) = f(x) + x \ ,$$

for all $x \in X$, is called the residual mapping.

Pooling operations, as for example max-pooling, reducing the spatial dimension of feature maps by utilizing strides larger than 1 are widely known.

On the other side there are also variations of convolutional operations to increase the size of feature maps. Most notably transposed convolutions and sub-pixel convolutions.

1.1.3 Transformers

Transformers operate on sequences of data $(x_k)_{k=1}^n$, where $x_k \in \mathbb{R}^d$. In the literature the elements of the input sequence are commonly referred to as tokens. Central to Transformer models is the so-called attention mechanism. The tokens x_k are embedded into three different subspaces using linear mappings $Q, K, V \in \mathbb{R}^{d',d}$. The mappings Q and K are used to compute the attention scores among the members of the sequence, measuring the level of relevance of their respective information for each other

$$A_{ij} = \frac{\exp(x_i^T K^T Q x_j)}{\sum_{k=1}^n \exp(x_k^T K^T Q x_j)} \ . \tag{1}$$

The outputs are then computed for all $j = 1, ..., n$ by

$$y_j = \sum_{i=1}^n A_{ij} V x_i \ . \tag{2}$$

Note that by construction for all $j = 1, ..., n$ holds

$$\sum_{i=1}^n A_{ij} = 1 \ .$$

Definition 5 (Self-Attention) *Let $Q, K, V \in \mathbb{R}^{d',d}$. The operation described in equations (1), (2)*

$$SA : \mathbb{R}^d \times ... \times \mathbb{R}^d \to \mathbb{R}^d \times ... \times \mathbb{R}^d \ , \quad SA(Q, K, V)(x_1, ..., x_k) = [y_1, ..., y_k] \ ,$$

is called self-attention.

To increase the capacity of Transformer models, multiple self-attention operations, called heads in the literature, are used in parallel to process the input sequence.

Definition 6 (Multi Headed Self-Attention) *Let $Q_h, K_h, V_h \in \mathbb{R}^{d',d}$ for $h = 1, ..., H$ and let $Q = (Q_1, ..., Q_H), K = (K_1, ..., K_H), V = (V_1, ..., V_H)$. The operation*

$$MSA(Q, K, V)\left((x_k)_{k=1}^n\right) = \left[SA(Q_1, K_1, V_1)\left((x_k)_{k=1}^n\right), ..., SA(Q_H, K_H, V_H)\left((x_k)_{k=1}^n\right)\right] \ ,$$

is called multi headed self-attention.

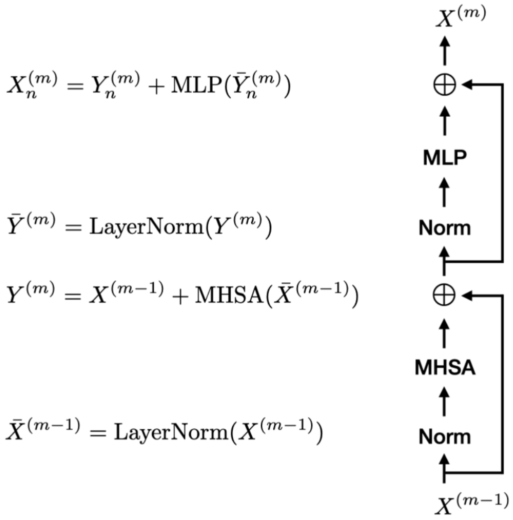

$X^{(m)}$
$X_n^{(m)} = Y_n^{(m)} + \mathrm{MLP}(\bar{Y}_n^{(m)})$
MLP
$\bar{Y}^{(m)} = \mathrm{LayerNorm}(Y^{(m)})$
Norm
$Y^{(m)} = X^{(m-1)} + \mathrm{MHSA}(\bar{X}^{(m-1)})$
MHSA
$\bar{X}^{(m-1)} = \mathrm{LayerNorm}(X^{(m-1)})$
Norm
$X^{(m-1)}$

Figure 1: Image taken from [**?**], Transformer Block architecture.

In order to refer to the architecture, that is Multi Headed Self-Attention with dimension $d \in \mathbb{N}$ and number of heads $H \in \mathbb{N}$, whose weights are not fixed but subject to optimization, we write $\mathrm{MSA}(d, H)$.

Given a number of heads $H \in \mathbb{N}$ generally the embedding dimension of each head is chosen as $\frac{d}{H}$.

Another key ingredient for Transformer models is layer normalization.

Definition 7 (Layer Normalization) *Let $\gamma \in \mathbb{R}$ and $\beta \in \mathbb{R}^d$. The operation given by*

$$LN \colon \mathbb{R}^d \to \mathbb{R}^d \ , \quad LN(x) = \gamma \bar{x} + \beta \ \textit{where} \ \bar{x}_{ki} = \frac{1}{\sqrt{var(x_k)}}(x_{ki} - \sum_{j=1}^{d} x_{kj})$$

is called layer normalization.

Typically Multi Headed Self-Attention is used in transformer blocks, the architecture is outlined in figure 1.

First layer normalization is applied to the inputs before the Multi Headed Self-Attention is being performed. The input is then added back to the outputs via a residual connection. The intermediate throughputs then undergo a second round of layer normalization, thereafter the tokens are processed individually by a neural network, before a second residual connection adds the intermediate results to the outputs. We describe this operation formally in the next definition.

Definition 8 (Transformer Block) *Let $d, H \in \mathbb{N}$ and $\Phi : \mathbb{R}^d \to \mathbb{R}^d$ be some mapping. The architecture*

$$(d, H, \Phi) = R\big(\Phi \circ LN\big) \circ R\big(MSA(d, H) \circ LN\big)$$

is called a Transformer Block.

Typically the mapping Φ is some neural network, with only a few number of layers.

Transformers operate on sequential data, so how to apply these models to images? In their 2020 paper ”An image is worth 16 × 16 words: Transformers for image recognition at scale”, Dosovitskiy et al. [**?**] propose to partition image data $X \in \mathbb{R}^{C \times H \times W}$ into a sequence of patches sized $P \times P$, for some $P \in \mathbb{N}$, which are flattened and then treated as tokens and fed to the transformer. More precisely the sequence is given by $(\hat{x}_k)_{k=1}^N$ where $\hat{x}_k \in \mathbb{R}^{C \cdot P^2}$ and $N = \frac{HW}{P^2}$. Here we implicitly assume that $H \mod P = W \mod P = 0$, i.e. both the height and the width are devisable by the patch size P. Let $w = \frac{W}{P}$, mathematically the partitioning can be described as follows

$$\hat{x}_k(c_i \cdot h_i \cdot w_i) = X\left(c_i, \lfloor \frac{k}{w} \rfloor + h_i, k \mod w + w_i\right) \ , \tag{3}$$

for $k = 1, ..., N$, $c_i = 1, ..., C$ and $h_i, w_i = 1, ..., P$. Lastly the flattened patches are embedded to obtain the final sequence of tokens $(x_k)_{k=1}^N$, that is

$$x_k = E\hat{x}_k \ ,$$

for some $E \in \mathbb{R}^{D \times C \cdot P^2}$. The approach is also visualized in figure 2.

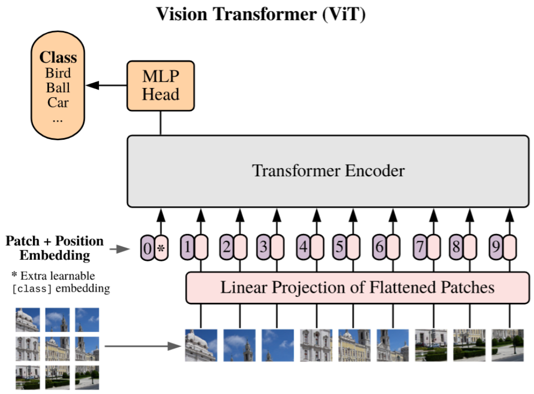

Vision Transformer (ViT)
Class
Bird
Ball
Car
...
MLP Head
Transformer Encoder
Patch + Position Embedding
* Extra learnable [class] embedding
0
*
1
2
3
4
5
6
7
8
9
Linear Projection of Flattened Patches

Figure 2: Image taken from [**?**], Vision Transformer.

A problem that comes with this approach, pointed out by Liu et al. [**?**] that unlike in language, where a word naturally offers itself as the atomic unit, visual elements vary in scale, making the fixed patch sizes unsuitable for tasks requiring predictions at pixel level, as for example semantic segmentation. Simply treating each individual pixel as a token would solve the problem, but at the same time introduce immense computational complexity. For a full HD image of size 1920×1080 this leads to a sequence length of $2.0736 \cdot 10^6$. Thus to reduce computational complexity, but at the same time maintain a global receptive field, Liu et al. [?] propose three concepts.

First the image is partitioned into non-overlapping patches, similarly as described in equation (3), with the difference being that each pixel within the patch is treated as a token itself

$$x_{i \cdot j}^{(p)}(c) = X\left(c, \left\lfloor \frac{p}{P} \right\rfloor + i, p \mod P + j\right) \ , \tag{4}$$

where $p = 1, ..., N$ and $i, j = 1, ..., P$. Self-attention is then performed locally inside of the patch, that is

$$(y_k^{(p)})_{k=1}^P = \mathrm{MSA}(Q, K, V)\left((x_k^{(p)})_{k=1}^P\right) \ , \tag{5}$$

for $p = 1, ..., N$, for some $Q, K, V \in \mathbb{R}^{H \times \frac{d}{H} \times d}$.

If equation (5) would be used repeatedly to update features, information is limited to flow only inside the individual patches. To this end Liu et al. [?] introduce the shifted window mechanism. Consecutive operations of self-attention use different partitionings of the feature map. We consider the partitioning described in equation (4) as the unshifted variant, for the shifted partitioning the borders of the patches are moved down and to the right by $s = \left\lfloor \frac{P}{2} \right\rfloor$ units, that is

$$x_{i \cdot j}^{(p)}(c) = X\left(c, \left(\left\lfloor \frac{p}{P} \right\rfloor + i + s\right) \mod H, (p \mod P + j + s) \mod W\right) \ ,$$

where $p = 1, ..., N$ and $i, j = 1, ..., P$. Applying the modulo to the arguments, leads to a cyclic shift, visualized in figure 3. This is necessary as the shift leads to overshooting the margins of the feature map. Instead also padding techniques could be applied, but Liu et al. [?] report achieving better results, whilst saving computational complexity.

To enable a global receptive field, Liu et al. [?] propose to merge neighboring patches, after the inputs are processed by a certain number of SWin TransformerBlocks. To this end patches $P_1, ..., P_4 \in \mathbb{R}^{C \times P \times P}$, inside a neighborhood of size 2×2 are stacked along the channel dimension, to form a super patch $\hat{P} = [P_1, ..., P_4] \in \mathbb{R}^{4 \cdot C \times P \times P}$. To fuse the features a convolutional layer is applied halving the channel dimension of the super patch

$$P = C(4 \cdot C, 2 \cdot C, \text{kernel-size} = 3, \text{padding} = 1)(\hat{P}) \ .$$

This process is repeated until the entire feature map is of size $P \times P$. The overall architecture of the SWin Transformer is shown in figure 4.

Definition 9 (Shifted window Transformer Block) *Let* $d, H \in \mathbb{N}$ *and* $\Phi : \mathbb{R}^d \to \mathbb{R}^d$

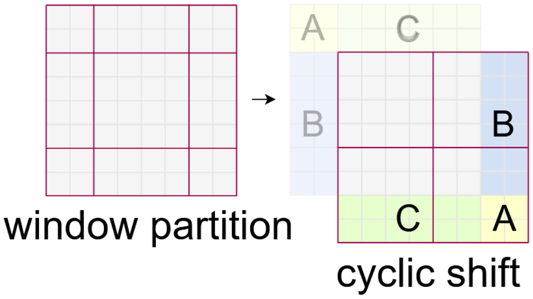


Figure 3: Image taken from [?], visualizing the cyclic shift. Here the pixels in the image are rearranged, and then the unshifted partitioning is used, to obtain the cyclic shift.

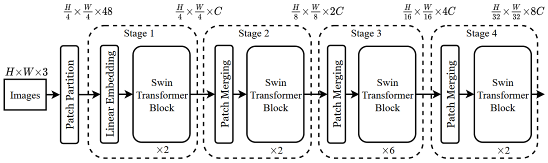


Figure 4: Image taken from [?], SWin Transformer.

Chen et al. [?] introduce overlapping Cross Attention (OCA), a modification of SWin Transformers. Whilst in equation (4) the patches where constructed to partition the feature map,

1.2 General Model Architecture

Independent of the domain of application, a general architectural choice that can be observed in all super resolution models, is that the architecture is made up of three components. A shallow feature extraction module H_S, a deep feature extraction module H_D and an image reconstruction module H_{IR}. Typically the model architecture is conceptualized as follows

$$H = H_{IR} \circ R(H_D) \circ H_S \ . \tag{6}$$

The shallow feature extraction module H_S scales the channel dimension of the input to a higher dimension, which is used throughout the majority of the network. Additionally it extracts low frequency features. The module is usually composed of only one or few convolutional layers.
The deep feature extraction module H_D forms the main part of the model. It is supposed to recover high frequency information. Here is where different architectures proposed in the literature vary the most, convolutional layers, transformer models and various combinations thereof have been tried out.
Note the residual connection in equation (6), the rational behind this being that this way the low frequency features extracted by H_S can bypass the deep feature extraction module H_D. The image reconstruction module H_{IR} maps the input back to the original channel dimension and scales the spatial dimension to the desired size. It has been experimentally confirmed that better results are achieved when scaling is done at the end, rather than processing the already spatially upsampled input. To this end usually transposed convolutional layers or pixel sshuflling layers are employed.

1.3 Single Image Super Resolution

1.3.1 Deep Residual Channel Attention Network

The Deep Residual Channel Attention Network (DRCAN) proposed by Zhang et al. [?], the channel attention mechanism is introduced to single image super resolution. Channel Attention enables the network to dynamically assess which feature maps / channels are more important or need more refinement. This is achieved by processing the globally pooled average of the feature maps using a lightweight network and then reweighing the feature maps based thereon.

The overall model architecture is depicted in figure 5. The input image X is first processed via an initial convolutional layer

$$F_0 = C(3, 64, \text{kernel-size} = 3, \text{padding} = 1)(X) \ .$$

The following convolutional layers used in the architecture of the DRCAN are of the form

$$C = C(64, 64, \text{kernel-size} = 3, \text{padding} = 1, \text{padding-mode} = \text{zero}) \ ,$$

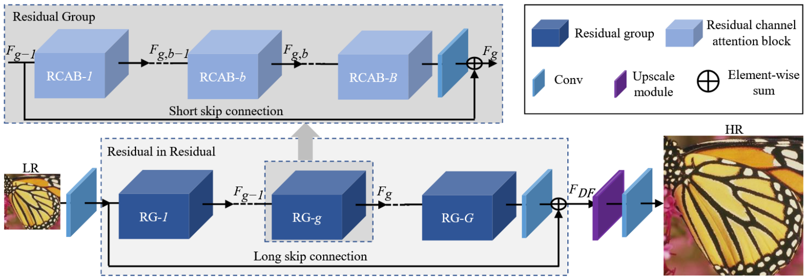


Figure 5: Image taken from [**?**], DRCAN model architecture.

that is 64 ingoing feature maps processed by 64 quadratic kernels of size 3 × 3 with zero-padding of size 1, so that feature map sizes are conserved throughout the model. The inial features F_0 are then further processed by a network with a residual in residual architecture

$$F_1 = H_{RIR}(F_0) \ .$$

For low-frequency features to bypass the deep feature extraction, a residual connection is used before the upsampling is performed

$$F_2 = F_0 + F_1 \ .$$

The final features F_2 are then upsampled using transposed convolutional layers.
The H_{RIR} network is composed of 10 Residual Groups followed by a final convolutional layer, that is

$$H_{RIR} = C \circ H_{RG} \circ ... \circ H_{RG} \ .$$

The Residual Groups (RG) are again composed of 20 Residual Channel Attention Blocks followed as well by a convolutional layer, the structure is encapsuled in a residual connection

$$H_{RG} = R\big(C \circ H_{RCAB} \circ ... \circ H_{RCAB}\big) \ .$$

The Residual Channel Attention Block (RCAB) depicted in figure 6, is made up of two convolutional layer, with a ReLU activation function in between, followed by a channel attention module, the output is then added back to the input again via a residual connection

$$H_{RCAB} = R\big(H_{CA} \circ C \circ \mathrm{ReLU} \circ C\big) \ .$$

The channel attention mechanism depicted in 7. The information of a feature map is first condesated into a single value by using global pooling

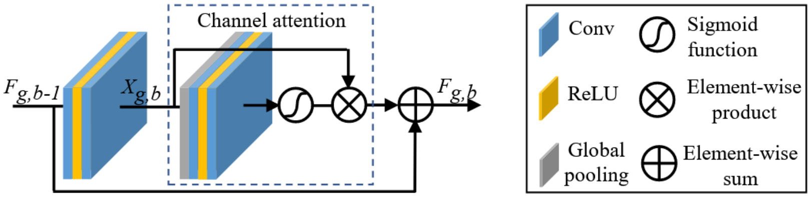


Figure 6: Image taken from [?], architecture of RCAB module.

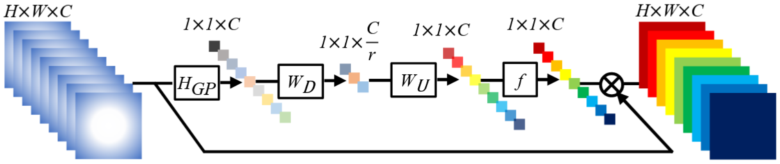


Figure 7: Image taken from [?], Channel Attention machanism.

$$z_c = H_{GP}(x_c) = \frac{1}{HW}\sum_{i=1}^{H}\sum_{j=1}^{W} x_c(i,j) \ ,$$

with the input $X = [x_1, ..., x_C] \in \mathbb{R}^{C\times H\times W}$. The vector $z \in \mathbb{R}^C$ is then processed by a two-layer neural network

$$\Phi = \sigma \circ F(4,64) \circ \text{ReLU} \circ F(64,4) \ ,$$

the sigmoid activation function is applied at last, in order to squash the attention scores into the interval $[0,1]$. Channel attention the weights the inputs according to the attention scores

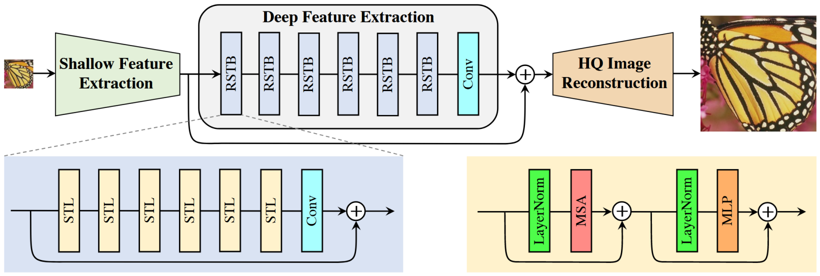
$$H_{CA}(X) = \Phi \circ H_{GP}(X) \cdot X \ .$$

1.3.2 Shifted Window Transfomer Image Restoration

The SWinIR model proposed by Liang et al. [?], makes use of the shifted window transfomer architecture introduced by Liu et al. [?]. While the model does not employ the hierarchical structure of the original architecture, it makes extensive use of the shifted window mechanism. The model architecture is depicted in figure 9.

The broader architectural design follows that described in section 1.2. Given inputs $X \in \mathbb{R}^{3\times H\times W}$, the shallow feature extraction is performed via a single convolutional layers

$$F_0 = C(3, 180, \text{kernel-size} = 3, \text{padding} = 1)(X) \ .$$



(a) Residual Swin Transformer Block (RSTB)

(b) Swin Transformer Layer (STL)

Figure 8: Image taken from [?], architecture of SWinIR model.

The features are then further processed by the deep feature extraction module

$$F_1 = H_D(F_0) + F_0 \ ,$$

before being upsampled using the image reconstruction module $I_{SR} = H_{IR}(F_1)$. To this end the authors employ a sub-pixel convolutional layer.

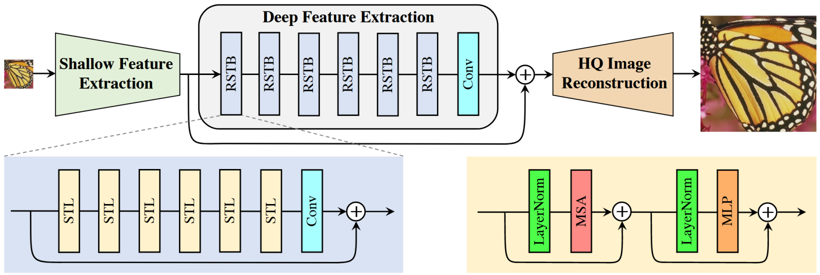
The deep feature extraction module is composed of 6 Residual Swin Transformer Blocks (RSTBs), followed by a last convolutional layer

$$H_D = C(180, 180) \circ H_{RSTB} \circ ... \circ H_{RSTB} \ .$$

Each RSTB is consists of 6 Transformer layers where every second makes use of the shifted window mechanism, these are suceeded by a final convolutional layer

$$H_{RSTB} = C(180, 180) \circ H_{STL} \circ H_{TL} \circ ... \circ \circ H_{STL} \circ H_{TL} \ .$$

1.3.3 Hybrid Attention Transfomer



(a) Residual Swin Transformer Block (RSTB)

(b) Swin Transformer Layer (STL)

Figure 9: Image taken from [?], architecture of SWinIR model.

2 Training

2.1 Preprocessing the Data

2.2 Training Methods

2.2.1 Single Image Super Resolution

2.2.2 Light Field-, Hyperspectral Image- and Spectral Super Resolution Methods

2.2.3 Diffusion Models