Adaptive Transform Domain Image Super-resolution Via Orthogonally Regularized Deep Networks

Supplementary Document

Tiantong Guo, Hojjat Seyed Mousavi, Vishal Monga
The Pennsylvania State University

In this supplementary document, we provide details of the DCT and IDCT using the CDCT layer as well as the associated proof. We also provide image validation of the proof. The training details such as the data augmentation methods and training records are also provided. Hardware, software, and project pages are pointed out in this document. Furthermore, the network size and speed is reported. We also included a discussion over the CDCT filter size choice. The content is indexed as below:

CONTENTS

I. IMAGE DCT AND IDCT

A. Discrete cosine transform

Given an image $\mathbf{x}$ of size $H \times W$, it can be decomposed into $H/N \times W/N$ blocks of size $N \times N$ [1]. For the $(m,n)^{th}$ block $\mathbf{x}_{m,n} \in \mathbb{R}^{N\times N}$, the DCT coefficients are computed as:

$$\mathbf{X}_{m,n}(k_1,k_2) = \sum_{n_2=0}^{N-1}\sum_{n_1=0}^{N-1} \mathbf{x}_{m,n}(n_1,n_2) \times \mathbf{w}^{\text{dct}}_{k_1,k_2}(n_1,n_2) \tag{1}$$

where $k_1, k_2 \in \{0,\ldots,N-1\}$, and $\mathbf{w}_{k_1,k_2} \in \mathbb{R}^{N\times N}$ is the DCT basis function defined as:

$$\mathbf{w}^{\text{dct}}_{k_1,k_2}(n_1,n_2) = C_{k_1,k_2} cos\left[(\frac{\pi}{N}\left(n_1+\frac{1}{2}\right)k_1\right] cos\left[(\frac{\pi}{N}\left(n_2+\frac{1}{2}\right)k_2\right] \tag{2}$$

where $C_{k_1,k_2} = \frac{\sqrt{1+\delta_{k_1}}\sqrt{1+\delta_{k_2}}}{N}$ and

$$\delta_k = \begin{cases} 1, & \text{if } k = 0, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

For $N = 8$, there are 8×8 DCT bases and each basis $\mathbf{w}^{\text{dct}}_{k_1,k_2} \in \mathbb{R}^{N\times N}$.

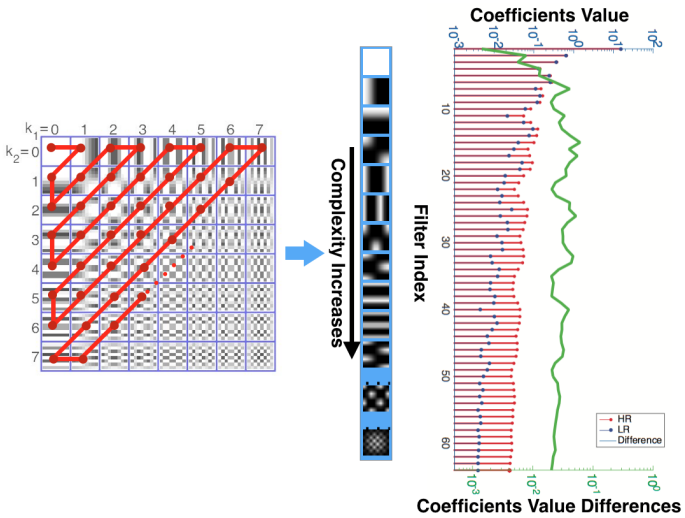
B. Inverse discrete cosine transform

Corresponding to the DCT, the inverse DCT (IDCT) for the $(m,n)^{th}$ block is computed as:

$$\mathbf{x}_{m,n}(n_1,n_2) = \sum_{k_2=0}^{N-1}\sum_{k_1=0}^{N-1} \mathbf{X}_{m,n}(k_1,k_2) \times \mathbf{w}^{\text{dct}}_{k_1,k_2}(n_1,n_2) \tag{4}$$

II. CDCT LAYER STRUCTURE

We treat DCT basis functions as filters and organize them in zig-zag order:



- Re-index $\mathbf{w}^{\text{dct}}_{k1,k2}$ to $\mathbf{w}_i$ with zig-zag mapping function Zig: $Zig(k_1,k_2) = i$ where $(k_1,k_2) \in \{0,\ldots,N-1\}\times\{0,\ldots,N-1\} \to i \in \{1,\ldots,N\times N\}$
- $Zig(k_1,k_2) = i$ and $Zig^{-1}(i) = (k_1,k_2)$
- Now here are $N \times N$ DCT basis filters $\{\mathbf{w}_i\}_{i=1}^{N\times N}$, each of size $N \times N$
- As i increasing, $\mathbf{w}_i$'s complexity also increases.

III. DCT BY CDCT LAYER

A. For stride $S = N$ without overlapping

First we show the DCT by CDCT layer without overlapping and stride of $S = N = 8$. Convolve image $\mathbf{x} \in \mathbb{R}^{W\times H}$ with $\{\mathbf{w}_i\}_{i=1}^{N\times N}$ DCT basis filters:

- Convolve without overlapping, with shift of N.
- For $(m,n) = \{1,...,H/N\} \times \{1,...,W/N\}$, $\mathbf{X}^{\text{dct}}_{m,n}(k_1,k_2)$ is the $(m,n)^{th}$ DCT coefficients block indexed by (k_1,k_2).
- Use zig-zag reorder function: $\mathbf{X}^{\text{dct}}_{m,n}(i) := \mathbf{X}^{\text{dct}}_{m,n}(Zig(k_1,k_2))$.
- We claim: for a fixed i, $\mathbf{x} * \mathbf{w}_i$ gives DCT coefficients $\mathbf{X}^{\text{dct}}_{m,n}(i)$ for every $(m,n)^{th}$ blocks.
- $\mathbf{X}_i := \mathbf{x} * \mathbf{w}_i$, where $\mathbf{X}_i \in \mathbb{R}^{\frac{W}{N}\times\frac{H}{N}}$
- As i increases, smaller details are captured by $\mathbf{x} * \mathbf{w}_i$

Proposition DCT by CDCT layer generates the same DCT coefficients $\mathbf{X}^{\text{dct}}_{m,n}$ for $\mathbf{x}_{m,n}$, where (m,n) denotes one $N \times N$ block's index.

Proof For a fixed block (m,n), $k_1,k_2,n_1,n_2 = \{1,\ldots,N\}$

We have DCT:

$$\mathbf{X}^{\text{dct}}_{m,n}(k_1,k_2) = \sum_{n_2=0}^{N-1}\sum_{n_1=0}^{N-1} \mathbf{x}_{m,n}(n_1,n_2) \times \mathbf{w}_{k_1,k_2}(n_1,n_2)$$

For CDCT layer, convolve x with w_i:

$$\mathbf{X}_i^{\text{cdct}}(m,n) = \sum_{n_2=m\times N}^{(m+1)\times N-1} \sum_{n_1=n\times N}^{(n+1)\times N-1} \mathbf{x}(n_1,n_2) \times \mathbf{w}_i(n_1 - n\times N + 1, n_2 - m\times N + 1) \tag{5}$$

For fixed (m,n) and $i \in \{1,...,N\times N\}$, the $\mathbf{X}_i^{\text{cdct}}(m,n) \in \mathbb{R}^{N^2\times 1}$ can be re-indexed as $\mathbf{X}_{m,n}^{\text{cdct}}(i) \in \mathbb{R}^{N\times N}$. And for $n_1 \in \{n\times N,...,(n+1)\times N-1\}; n_2 \in \{m\times N,...,(m+1)\times N-1\}$, $\mathbf{x}(n_1,n_2) \in \mathbb{R}^{N\times N}$ can be re-indexed as $\mathbf{x}_{m,n}(n_1,n_2) \in \mathbb{R}^{N\times N}$ with $n_1 \in \{0,...,N-1\}; n_2 \in \{0,...,N-1\}$:

$$\mathbf{X}_{m,n}^{\text{cdct}}(i) := \mathbf{X}_i^{\text{cdct}}(m,n) = \sum_{n_2=0}^{N-1}\sum_{n_1=0}^{N-1} \mathbf{x}_{m,n}(n_1,n_2) \times \mathbf{w}_i(n_1,n_2) \tag{6}$$

Now for the given i, $\exists(k_1,k_2)$ such that $\mathbf{w}_i = \mathbf{w}_{k_1,k_2}$ where $i = Zig(k_1,k_2)$. Thus,

$$\begin{aligned}\mathbf{X}_{m,n}^{\text{cdct}}(i) &= \sum_{n_2=0}^{N-1}\sum_{n_1=0}^{N-1} \mathbf{x}_{m,n}(n_1,n_2) \times \mathbf{w}_{k_1,k_2}(n_1,n_2) \\ &= \mathbf{X}_{m,n}^{\text{dct}}(k_1,k_2)\end{aligned} \tag{7}$$

where $i = Zig(k_1,k_2)$. ■

B. For stride $S < N$ with overlapping of $N - S$

Now we derive the DCT procedure with arbitrary stride S, and $S \le N$:

- Convolve with stride of S leading to an overlapping of $N - S$ at adjacent convolution operation on $\mathbf{x}$.
- For $(m,n) = \{1,...,H/S\} \times \{1,...,W/S\}$, $\mathbf{X}_{m,n}^{\text{dct}}(k_1,k_2) \in \mathbb{R}^{N\times N}$ is the $(m,n)^{th}$ DCT coefficients block indexed by (k_1,k_2) with overlapping of S with adjacent block.
- Use zig-zag reorder function: $\mathbf{X}_{m,n}(i) := \mathbf{X}_{m,n}^{\text{dct}}(Zig(k_1,k_2))$.
- For a fixed i, $\mathbf{x} * \mathbf{w}_i$ gives DCT coefficients $\mathbf{X}_{m,n}(i)$ for every $(m,n)^{th}$ blocks.
- $\mathbf{X}_i := \mathbf{x} * \mathbf{w}_i$, where $\mathbf{X}_i \in \mathbb{R}^{\frac{W}{S}\times\frac{H}{S}}$
- As i increases, smaller details are captured by $\mathbf{x} * \mathbf{w}_i$

Proposition Convolution between $\mathbf{x}$ and $\mathbf{w}_i$ in CDCT layer generates the same DCT coefficients $\mathbf{X}_{m,n}^{\text{dct}}$ for $\mathbf{x}_{m,n}$, where (m,n) denotes one $N\times N$ block's index.

Proof For a fixed block $(m,n) \in \{1,...,H/S\} \times \{1,...,W/S\}$, $k_1,k_2,n_1,n_2 = 1,\ldots,N$

We have DCT:

$$\mathbf{X}_{m,n}^{\text{dct}}(k_1,k_2) = \sum_{n_2=0}^{N-1}\sum_{n_1=0}^{N-1} \mathbf{x}_{m,n}(n_1,n_2) \times \mathbf{w}_{k_1,k_2}^{\text{dct}}(n_1,n_2) \tag{8}$$

For CDCT layer, convolve $\mathbf{x}$ with $\mathbf{w}_i$:

$$\mathbf{X}_i^{\text{cdct}}(m,n) = \sum_{n_2=m\times S}^{m\times S+N-1} \sum_{n_1=n\times S}^{n\times S+N-1} \mathbf{x}(n_1,n_2) \times \mathbf{w}_i(n_1 - n\times S + 1, n_2 - m\times S + 1) \tag{9}$$

For a fixed $(m,n) \in \{1,...,H/S\} \times \{1,...,W/S\}$ and $i = 1,...,N\times N$, the $\mathbf{X}_i^{\text{cdct}}(m,n)$ can be re-indexed as:

$$\mathbf{X}_{m,n}^{\text{cdct}}(i) := \mathbf{X}_i^{\text{cdct}}(m,n) = \sum_{n_2=0}^{N-1}\sum_{n_1=0}^{N-1} \mathbf{x}_{m,n}(n_1,n_2) \times \mathbf{w}_i(n_1,n_2) \tag{10}$$

Now for the given i, $\exists(k_1,k_2)$ such that $\mathbf{w}_i = w_{k_1,k_2}$ where $i = Zig(k_1,k_2)$. Thus,

$$\begin{aligned}\mathbf{X}_{m,n}^{\text{cdct}}(i) &= \sum_{n_2=0}^{N-1}\sum_{n_1=0}^{N-1} \mathbf{x}_{m,n}(n_1,n_2) \times \mathbf{w}_{k_1,k_2}(n_1,n_2) \\ &= \mathbf{X}_{m,n}^{\text{dct}}(k_1,k_2)\end{aligned} \tag{11}$$

where $i = Zig(k_1,k_2)$. Thus, DCT by CDCT layer generates zig-zag arranged blocked DCT. Note that the $\mathbf{X}_i$ with stride S produces rearranged DCT coefficients for S overlapped blocks. For example, $\mathbf{X}_i(m,n)$ and $\mathbf{X}_i(m,n-1)$ are the ith zig-zag reordered DCT coefficients associated with block $\mathbf{x}(x_1,y_1)$ and $\mathbf{x}(x_2,y_2)$ with $x_1,x_2 \in \{m\times S,...,m\times S+N-1\}$, $y_1 \in \{n\times S,...,n\times S+N-1\}$ and $y_2 \in \{(n-1)\times S,...,(n-1)\times S+N-1\}$. ■

IV. IDCT BY CDCT LAYER

A. For stride $S = N$ without overlapping

First we show the IDCT by CDCT layer with the DCT cube generated by non-overlapping and stride of $S = N$. Transpose-convolve features $\mathbf{X}_i \in \mathbb{R}^{\frac{W}{N} \times \frac{H}{N}}$, $i = 1, ..., N \times N$ with $\{\mathbf{w}_i\}_{i=1}^{N \times N}$ DCT basis: Padding $\mathbf{X}_i$ with padding function $g_s(\cdot)$. For a given location $(p, q) \in \{1, ..., W\} \times \{1, ..., H\}$:

$$\bar{\mathbf{X}}_i(p, q) = \begin{cases} \frac{1}{(N/S)^2}\mathbf{X}_i(k, l), & \text{if } p = k \times N \text{ and } q = l \times N \\ 0, & \text{Otherwise} \end{cases}, \tag{12}$$

where $k = \{1, \dots \frac{W}{N}\}, l = \{1, \dots \frac{H}{N}\}$ and $S = N$. Note that with $S = N$, the term $\frac{1}{(N/S)^2} = 1$ means the $g_s(\cdot)$ keeps the $\mathbf{X}_i$ value as given and does not apply reweighing. This is a different case from next section for $S \leq N$. We denote $g_s(\mathbf{X}_i) := \bar{\mathbf{X}}_i \in \mathbb{R}^{W \times H}$ which is a zero padded version of $\mathbf{X}_i$.

Transpose convolve: convolve $g_s(\mathbf{X}_i)$ with $\mathbf{w}_i$ with a shifting of 1. For a fixed i, $g_s(\mathbf{X}_i) * \mathbf{w}_i$ gives all i-th spatial component for $\mathbf{x}$.

$$\mathbf{x}_i = g(\mathbf{X}_i) * \mathbf{w}_i, \tag{13}$$

where $\mathbf{x}_i \in \mathbb{R}^{W \times H}$. The final spatial results:

$$\mathbf{x} = \sum_{i=1}^{N \times N} g_s(\mathbf{X}_i) * \mathbf{w}_i \tag{14}$$

Proposition Transpose convolve $g_s(\mathbf{X}_i)$ with $\mathbf{w}_i$ and sums together generates the same $\mathbf{x}_{m,n}$ from DCT coefficients $\mathbf{X}_{m,n}$, (m, n) denotes one $N \times N$ block's index.

Proof For a fixed block (m, n), $k_1, k_2, n_1, n_2 \in \{1, \dots, N\}$

IDCT:

$$\mathbf{x}^{\text{dct}}_{m,n}(n_1, n_2) = \sum_{k_2=0}^{N-1} \sum_{k_1=0}^{N-1} \mathbf{X}_{m,n}(k_1, k_2) \times \mathbf{w}_{k_1,k_2}(n_1, n_2) \tag{15}$$

IDCT by CDCT layer:

$$\mathbf{x}^{\text{cdct}}_{m,n}(n_1, n_2) = \sum_{i=1}^{N \times N} \sum_{p=0}^{N-1} \sum_{q=0}^{N-1} \bar{\mathbf{X}}_i(m \times N + n_1 - p, n \times N + n_2 - q) \times \mathbf{w}_i(p, q) \tag{16}$$

where $\bar{\mathbf{X}}_i(m \times N + n_1 - p, n \times N + n_2 - q) \neq 0$, while $n_1 - p + m \times N = m \times N$ and $n_2 - q + n \times N = n \times N$ (based the definition of $\bar{\mathbf{X}}$ from Eq. (12)), reordering the index we have:

$$\mathbf{x}^{\text{cdct}}_{m,n}(n_1, n_2) = \sum_{i}^{N \times N} \mathbf{X}_i(m, n) \times \mathbf{w}_i(n_1, n_2) \tag{17}$$

Since $Zig^{-1}(\mathbf{w}_i) = \mathbf{w}_{k_1,k_2}$ and $Zig^{-1}(\mathbf{X}_i) = \mathbf{X}_{k_1,k_2}$, we have:

$$\begin{aligned} \mathbf{x}^{\text{cdct}}_{m,n}(n_1, n_2) &= \sum_{i}^{N \times N} \mathbf{X}_i(m, n) \times \mathbf{w}_i(n_1, n_2) \\ &= \sum_{k_1=0}^{N-1} \sum_{k_2=0}^{N-1} \mathbf{X}_{k_1,k_2}(m, n) \times \mathbf{w}_{k_1,k_2}(n_1, n_2) \\ &= \sum_{k_1=0}^{N-1} \sum_{k_2=0}^{N-1} \mathbf{X}_{m,n}(k_1, k_2) \times \mathbf{w}_{k_1,k_2}(n_1, n_2) \end{aligned} \tag{18}$$

IDCT by neural network generates same $\mathbf{x}_{m,n}(n_1, n_2)$ for a given block. ■

B. For stride $S < N$ with overlapping of $N - S$

Now we derive the IDCT procedure with DCT cube associated with arbitrary stride S, and $S \leq N$. The transpose-convolve features $\mathbf{X}_i \in \mathbb{R}^{\frac{W}{S} \times \frac{H}{S}}$, $i \in \{1, ..., N \times N\}$ with $\{\mathbf{w}_i\}_{i=1}^{N \times N}$ DCT basis. We first pad $\mathbf{X}_i$ with padding function $g_s(\cdot)$. For a given location $(p, q) \in \{1, ..., W\} \times \{1, ..., H\}$:

$$\bar{\mathbf{X}}_i(p, q) = \begin{cases} \frac{1}{(N/S)^2}\mathbf{X}_i(k, l), & \text{if } p = k \times S \text{ and } q = l \times S \\ 0, & \text{Otherwise} \end{cases}, \tag{19}$$

where $k \in \{1, \dots \frac{W}{S}\}, l \in \{1, \dots \frac{H}{S}\}$. Note that the padding function $g_s(\cdot)$ is averaged by $\frac{1}{(S/N)^2}$. This is due to the $S \leq N$ and coefficients are computed multiple times. We denote $g_s(\mathbf{X}_i) := \bar{\mathbf{X}}_i \in \mathbb{R}^{W \times H}$ which is a weighted zero padded version of $\mathbf{X}_i$.

Transpose convolve: convolve $g_s(\mathbf{X}_i)$ with $\mathbf{w}_i$ with a shifting of 1. For a fixed i, $g_s(\mathbf{X}_i) * \mathbf{w}_i$ gives all i-th spatial component for $\mathbf{x}$:

$$\mathbf{x}_i = g_s(\mathbf{X}_i) * \mathbf{w}_i, \tag{20}$$

where $\mathbf{x}_i \in \mathbb{R}^{W \times H}$. The final spatial SR results: $\mathbf{x} = \sum_{i=1}^{N \times N} g_s(\mathbf{X}_i) * \mathbf{w}_i$

Proposition Transpose convolve $g_s(\mathbf{X}_i)$ with $\mathbf{w}_i$ and sums together generates the same $x_{m,n}$ from DCT coefficients $X_{m,n}$, where (m, n) denotes one $N \times N$ block's index.

Proof For a fixed block (m, n), $k_1, k_2, n_1, n_2 \in \{1, \dots, N\}$

IDCT:

$$\mathbf{x}^{\text{dct}}_{m,n}(n_1, n_2) = \sum_{k_2=0}^{N-1} \sum_{k_1=0}^{N-1} \mathbf{X}_{m,n}(k_1, k_2) \times \mathbf{w}_{k_1,k_2}(n_1, n_2) \tag{21}$$

IDCT by CDCT layer: From previous proof, we have that

$$\bar{\mathbf{x}}^{\text{cdct}}_{m,n}(n_1, n_2) = \sum_{i}^{N \times N} \frac{1}{(N/S)^2} \mathbf{X}_i(m, n) \times \mathbf{w}_i(n_1, n_2) \tag{22}$$

Now $S \leq N$, $(n_1, n_2) \in \{0, 1, ..., N\} \times \{0, 1, ..., N\}$, for every $\mathbf{X}_i(m, n)$, Eq. (22) produces a $\bar{\mathbf{x}}_{m,n} \in \mathbb{R}^{N \times N}$:

$$\bar{\mathbf{x}}^{\text{cdct}}_{m,n} = \sum_{i}^{N \times N} \frac{1}{(N/S)^2} \mathbf{X}_i(m, n) \odot \mathbf{w}_i, \tag{23}$$

where $\odot$ denotes scaler multiple with every element in the matrix.

The $\frac{1}{(N/S)^2}$ is generated by the $g_s(\cdot)$ weighting the $\mathbf{X}_i$ as in Eq. (19). At any given block (m, n)'s location (n_1, n_2), $\mathbf{x}^{\text{cdct}}_{m,n}(n_1, n_2)$ is computed as[1]:

$$\mathbf{x}^{\text{cdct}}_{m,n}(n_1, n_2) = \sum_{k=-N/2S}^{N/2S} \sum_{l=-N/2S}^{N/2S} \bar{\mathbf{x}}^{\text{cdct}}_{m-k,n-l}(n_1 - k \times S, n_2 - l \times S) \tag{24}$$

With stride S, $\bar{\mathbf{x}}^{\text{cdct}}_{m,n}$ are corresponding to patches that are overlapping with width of $N - S$, the following entries are equal to each other:

$$\bar{\mathbf{x}}^{\text{cdct}}_{m,n}(n_1, n_2) = \bar{\mathbf{x}}^{\text{cdct}}_{m-k,n-l}(n_1 - k \times S, n_2 - l \times S), \text{ where } k \in [-N/2S, ..., N/2S],\ l \in [-N/2S, ..., N/2S] \tag{25}$$

Thus Eq. (24) is rewritten as:

$$\mathbf{x}^{\text{cdct}}_{m,n}(n_1, n_2) = (N/S)^2 \sum_{i}^{N \times N} \frac{1}{(N/S)^2} \mathbf{X}_i(m, n) \times \mathbf{w}_i(n_1, n_2) = \sum_{i}^{N \times N} \mathbf{X}_i(m, n) \times \mathbf{w}_i(n_1, n_2) \tag{26}$$

Follow the same proof as Eq. (18), we have shown that the transpose convolution between $g_s(\mathbf{X}_i)$ with $\mathbf{w}_i$ and sums together generates the same $x_{m,n}$ from DCT coefficients $X_{m,n}$. ■

V. Validation of DCT and IDCT by CDCT layer

In this section we provide validations of DCT and IDCT performed by CDCT layer. First with $S = N = 8$, the input $\mathbf{x}$ of the CDCT layer is shown in Fig. 1. The MSE between input and reconstructed image $= 1.2e - 14$ which is in the range of round-off precision error.

With $S = 2$, there are overlapping of $N - S = 6$ between every adjacent reconstructed patch $\mathbf{x}_{m,n}$. This overlapping is averaged by weighting the reconstruct patch. Detailed image shown in Fig. 2. The MSE between the input image and the reconstruction $= 1.5e - 14$ which is in the range of round-off precision error.

[1]For $\mathbf{x}_{m,n}$ where $(m, n) \in \{0, 1, ..., N/S - 1\} \times \{0, ..., N/S - 1\}$ or $(m, n) \in \{0, 1, ..., N/S - 1\} \times \{H/S - N/S \times S, H/S - (N/S - 1) \times S, ..., H/S - S\}$ or $(m, n) \in \{W/S - N/S \times S, W/S - (N/S - 1) \times S, ..., W/S - S\} \times \{0, 1, ..., N/S - 1\}$ or $(m, n) \in \{W/S - N/S \times S, W/S - (N/S - 1) \times S, ..., W/S - S\} \times \{H/S - N/S \times S, H/S - (N/S - 1) \times S, ..., H/S - S\}$, the overlapping region in these patches are not $(S/N)^2$ times. This edge effect is eliminated by either cropping out the N-pixel width edge or re-weight the edge by its according overlapping times. This work choice to crop out the edges is the same operation following the literature as shows in multiple deep learning based studies, Dong *et.al* [2], Kim *et.al* [3], etc.

(a) input image $\mathbf{x}$ (b) reconstructed image

Fig. 1: CDCT layer reconstruction validation, i.e. DCT followed by IDCT with $S = N = 8$

(a) input image $\mathbf{x}$ (b) reconstructed image

Fig. 2: CDCT layer reconstruction validation, i.e. DCT followed by IDCT with $S = 2$

VI. Modified backpropagation for ORDSR

As shown in the main manuscript, the cost function of ORDSR is given as:

$$\begin{aligned}\mathbf{L}(\mathbf{\Theta}, \mathbf{B}) = &\underbrace{\frac{1}{2}\|F(\mathbf{x}) - \mathbf{y}\|_2^2}_{\text{MSE loss}} + \sigma\frac{1}{2}\sum_l \underbrace{\|\mathbf{W}_l\|_2^2}_{\text{weight decay}} \\ &+\gamma\frac{1}{2}\sum_{(i,j),i\neq j}\underbrace{\|vec(\mathbf{w}_i)^T vec(\mathbf{w}_j) - \epsilon\|_2^2}_{\text{orthogonality constraint}} \\ &+\lambda\frac{1}{2}\sum_t \underbrace{\|var(\mathbf{w}_t) - var(\mathbf{w}_t^{\text{dct}})\|_2^2}_{\text{complexity order constraint}}\end{aligned} \tag{27}$$

where the $var(\mathbf{w}_i)$ is given by Bessel's correction version:

$$var(\mathbf{w}) = \frac{1}{N^2 - 1}\sum_m (\mathbf{w}^m - \frac{1}{N^2}\sum_n \mathbf{w}^n)^2 \tag{28}$$

where $\mathbf{w}_i^*$ denotes an arbitrary scalar entry in filter $\mathbf{w}_i$ and $\sum_* \mathbf{w}_i^*$ denotes the summation of all the elements inside $\mathbf{w}$. Note that $\hat{\mathbf{y}} := F(\mathbf{x})$. And the network is trained to minimize:

$$\mathbf{\Theta}, \mathbf{B} = \arg\min_{\mathbf{\Theta},\mathbf{B}} \mathbf{L}(\mathbf{\Theta}, \mathbf{B}) \tag{29}$$

where $\mathbf{\Theta} = \{\mathbf{\Theta}^{\text{cnn}}, \{\mathbf{w}_i\}_{i=1}^{64}\}$, $\mathbf{\Theta}^{\text{cnn}} = \{\mathbf{W}_l\}_{l=1}^{D}$ and $\mathbf{B} = \{b_l\}_{l=1}^{D}$ (as in the main manuscript). Note that the $\mathbf{W}_l \in \mathbb{R}^{c_l \times n_l \times n_l}$ is a representative notation of the m_l filters in layer l. Detailed m_l, c_l, n_l please see the main manuscript.
The weights and bias are updates as:

$$\mathbf{\Theta}^{t+1} = \mathbf{\Theta}^t - \eta\nabla_{\mathbf{\Theta}}\mathbf{L}(\mathbf{\Theta}^t),\ \mathbf{B}^{t+1} = \mathbf{B}^t - \eta\nabla_{\mathbf{B}}\mathbf{L}(\mathbf{B}^t) \tag{30}$$

The following terms need to be computed:

$$\frac{\partial \mathbf{L}}{\partial \mathbf{W}_l}, \frac{\partial \mathbf{L}}{\partial \mathbf{w}_i}, \frac{\partial \mathbf{L}}{\partial \mathbf{b}_l} \tag{31}$$

where $\mathbf{W}_l$ denotes a filter at l^{th} layer of the CNN representatively, $\mathbf{w}_i$ denotes the ith filter in the CDCT layer and $\mathbf{b}_l$ denotes a bias at l^{th} layer of the CNN.
For $\mathbf{b}_l$ in D-layer CNN:

$$\frac{\partial \mathbf{L}}{\partial \mathbf{b}_l^a} = - < (\hat{\mathbf{y}} - \mathbf{y}), \frac{\partial \mathbf{y}}{\partial \mathbf{b}_l^a} >_F \tag{32}$$

For $\mathbf{W}_l$ in D-layer CNN:

$$\frac{\partial \mathbf{L}}{\partial \mathbf{W}_l^a} = - < (\hat{\mathbf{y}} - \mathbf{y}), \frac{\partial \mathbf{y}}{\partial \mathbf{W}_l^a} >_F + \sigma < \mathbf{W}_l, \frac{\partial \mathbf{W}_l}{\partial \mathbf{W}_l^a} >_F \tag{33}$$

where $\mathbf{W}_l^a$ denotes an arbitrary scalar entry within the representative filter $\mathbf{W}_l$, and $< \cdot, \cdot >_F$ denotes the real value Frobenius inner product: For two real valued matrix $\mathbf{A}$ and $\mathbf{B}$ with same dimension, $< \mathbf{A}, \mathbf{B} >_F := \sum_{i,j} A_{i,j} B_{i,j}$ where i, j are the indexes of the entries. In Eq. (33), $\frac{\partial \mathbf{y}}{\partial \mathbf{W}_l^a}$ is computed by following the standard backpropagation rule for each layer l.
For the CDCT filter $\mathbf{w}_i$, the gradient w.r.t an arbitrary scalar entry $\mathbf{w}_i^a$ is given by:

$$\begin{aligned}\frac{\partial \mathbf{L}}{\partial \mathbf{w}_i^a} =& - < (\hat{\mathbf{y}} - \mathbf{y}), \frac{\partial \mathbf{y}}{\partial \mathbf{w}_i^a} >_F \\ &\underbrace{+ \gamma \sum_{(j)} \left(vec(\mathbf{w}_i)^T vec(\mathbf{w}_j) - \epsilon \right) \mathbf{w}_j^a}_{\text{gradient of orthogonality constraint w.r.t } \mathbf{w}_i^a} \\ &\underbrace{+ \lambda \frac{\partial var(\mathbf{w}_i)}{\partial \mathbf{w}_i^a} \left(var(\mathbf{w}_i) - var(\mathbf{w}_i^{\text{dct}}) \right)}_{\text{gradient of complexity order constraint w.r.t } \mathbf{w}_i^a}\end{aligned} \tag{34}$$

where $\frac{\partial \mathbf{y}}{\partial \mathbf{w}_l^a}$ is computed following the standard backpropagation rule. $\frac{\partial var(\mathbf{w}_i)}{\partial \mathbf{w}_i^a}$ is the partial derivative of $var(\mathbf{w}_i)$ w.r.t $\mathbf{w}_i^a$ is given by:

$$\begin{aligned}\frac{\partial var(\mathbf{w}_i)}{\partial \mathbf{w}_i^a} &= \frac{\partial \frac{1}{N^2-1} \sum_m (\mathbf{w}_i^m - \frac{1}{N^2} \sum_n \mathbf{w}_i^n)^2}{\partial \mathbf{w}_i^a} \\ &= \frac{2}{N^2-1} \sum_m \left((\mathbf{w_i^m} - \frac{1}{N^2} \sum_n \mathbf{w}_i^n) \frac{\partial (\mathbf{w_i^m} - \frac{1}{N^2} \sum_n \mathbf{w}_i^n)}{\mathbf{w}_i^a} \right) \\ &= \frac{2}{N^2-1} \left[\sum_{m \neq a} \left((\mathbf{w_i^m} - \frac{1}{N^2} \sum_n \mathbf{w}_i^n)(0 - \frac{1}{N^2}) \right) + \left((\mathbf{w_i^a} - \frac{1}{N^2} \sum_n \mathbf{w}_i^n)(1 - \frac{1}{N^2}) \right) \right] \\ &= \frac{2}{N^2-1} \left[-\frac{1}{N^2} \sum_{m \neq a} \left((\mathbf{w_i^m} - \frac{1}{N^2} \sum_n \mathbf{w}_i^n) \right) - \frac{1}{N^2} (\mathbf{w_i^a} - \frac{1}{N^2} \sum_n \mathbf{w}_i^n) + \mathbf{w_i^a} - \frac{1}{N^2} \sum_n \mathbf{w}_i^n) \right] \\ &= \frac{2}{N^2-1} \left[-\frac{1}{N^2} \sum_m \left((\mathbf{w_i^m} - \frac{1}{N^2} \sum_n \mathbf{w}_i^n) \right) + \mathbf{w_i^a} - \frac{1}{N^2} \sum_n \mathbf{w}_i^n) \right] \\ &= \frac{2}{N^2(N^2-1)} \left[N^2 \mathbf{w}_i^a - \sum_n \mathbf{w}_i^n - \sum_m \left(\mathbf{w}_i^m - \frac{1}{N^2} \sum_n \mathbf{w}_i^n \right) \right]\end{aligned} \tag{35}$$

The aforementioned $\frac{\partial \mathbf{y}}{\partial \mathbf{W}_l^a}$, $\frac{\partial \mathbf{y}}{\partial \mathbf{w}_l^a}$ and $\frac{\partial \mathbf{y}}{\partial \mathbf{b}_l^a}$ are following the same backpropagation rule of CNN [4].

VII. Training Details

A. Dataset and augmentations

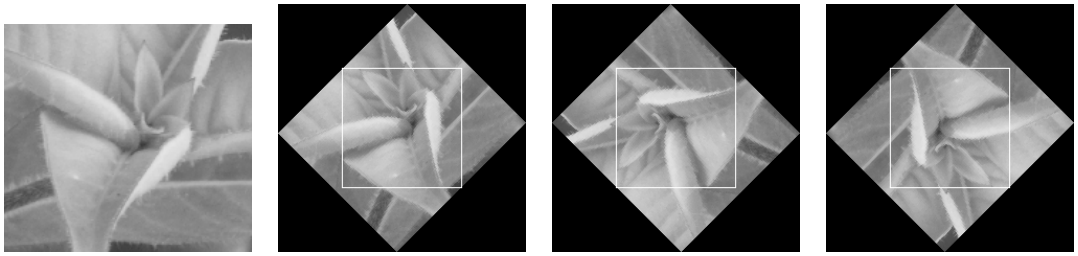
The 291 images dataset [5] is used for training. The images are augmented by a combination of three methods:

1) Rotating the images by $\{45°, 90°, 135°, 180°, 225°, 270°, 315°\}$;
2) Horizontal and vertical flip;
3) Scaling by factors of $\{0.7, 0.8, 0.9\}$.

(a) Original training image (b) Luminance channel of original image (c) Rotation of 90° (d) Rotation of 180° (e) Rotation of 270°

(f) Horizontal flip (g) Vertical flip (h) Scale by 0.7 (i) Scale by 0.8 (j) Scale by 0.9

Fig. 3: Training image *000t16.bmp*, its luminance channel with rotations, flips, and scaling.



(a) Original luminance training image (b) Rotation of 45° (c) Rotation of 135° (d) Rotation of 225°

(e) Copped rotation of 45° (f) Copped rotation of 135° (g) Copped rotation of 225°

Fig. 4: Training image *000t16.bmp* and its luminance channel with rotations. The cropped regions are shown within the white bonding boxes.

The augmentation method will resulting $8 \times 2 \times 3 = 48$ times of the original 291 training images, giving at total $291 \times 48 = 13968$ training images. During the training, at each epoch, instead of taking in all the augmented data, 50% of the 13968 images are selected to form the training patches. Training patches are cropped out as described in the main manuscript. During the training, random patches are selected forming the training batches. This random selection technique combining with Statistic Gradient Descent method speeds the training process.

Fig. 3 and Fig. 4 show the augmentation examples of the training set. Note that for rotation angle $\{45°, 135°, 225°\}$, only the valid image parts are stored as newly augmented images as shown in Fig. 4.

B. Hardware, software, and project page

Both training and test are conducted on one NVIDIA Titan X GPU (12GB).

The implementation of ORDSR and DCT-DSR uses the Tensorflow package [6], version r1.2 with python 2.7.

Project page: http://signal.ee.psu.edu/research/ORDSR.html.

TABLE I: Average PSNR on Set5 with scale factor 3 – different stride size.

S	2	3	4	5	8
PSNR(db)	34.31	34.10	33.85	33.60	32.95
InferenceTime(sec)	0.1	0.08	0.062	0.045	0.02

C. Network size and speed

VDSR uses 2 $3\times3\times64$ and 18 $3\times3\times64\times64$ convolutional layer. Total parameters[2]: $2\times3\times3\times64+18\times3\times3\times64\times64=664704$.

EDSR uses 32 residual blocks where each block has 2 convolutional layer. Each convolutional layer has size of $3\times3\times256\times256$. Also there are one input layer and one output layer of size $3\times3\times256$ leading the total number of parameters be: $32\times3\times3\times256\times256\times2+2\times3\times3\times256=37753344$.

RDN uses 16 dense residual blocks where each block has 8 convolutional layer. Each convolutional layer has size of $3\times3\times64\times64$. Also there are one input layer and one output layer of size $3\times3\times64\times3$ leading the total number of parameters be: $16\times8\times3\times3\times64\times64+2\times3\times3\times256\times3=4732416$.

ORDSR uses one $8\times8\times64$ CDCT layer, one $5\times5\times64\times64$, thirteen $3\times3\times64\times64$, and one $3\times3\times60\times64$ convolutional layers. Total parameter number is: $8\times8\times64+5\times5\times64\times64+13\times3\times3\times64\times64+3\times3\times60\times64=620288$. This makes ORSR uses 44416 less parameters than VDSR and about 5% of parameters as EDSR.

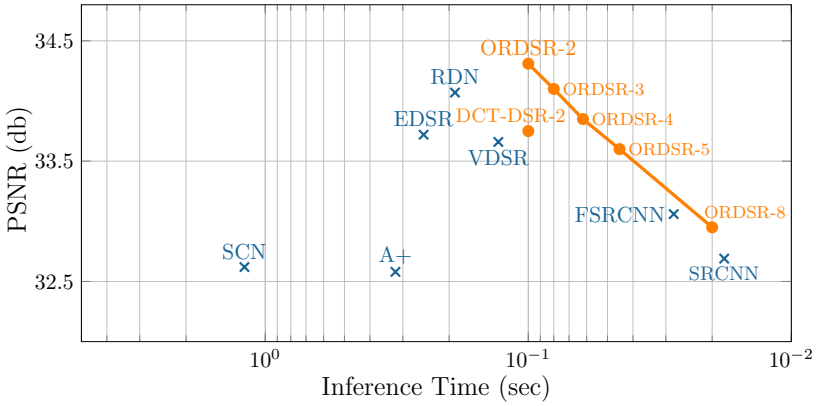


Fig. 5: ORDSR vs. state-of-the-art methods. PSNR on Set5 with scale factor of 3, plotted against inference time.

Fig. 5 plots the inference time vs. PSNR of ORDSR and other competing methods for scale factor 3 on Set5, where numerical results are reported in Table I. According to the guidelines in [7], ORDSR inference time may in fact be considered close to real-time processing. ORDSR-2, -3, -4, -5, and -8 refer to ORDSR with different stride S. Note that using larger stride can further speed up the inferences.

D. CDCT filter size

As we select the CDCT layer to be initialized from the 8×8 conventional DCT basis following the literate, other CDCT filter size may still be feasible. We then initial the CDCT layer from the DCT basis of 6×6, 8×8 (as in ORDSR), and 10×10 and conduct the training as the ORDSR with two constraints in place. Note that during training, other network parameters are setup following the guide of ORDSR (*see main manuscript Section IV-B*).

TABLE II: Average PSNR on Set5 with scale factor 3 – different CDCT layer filter size.

$N\times N$	6×6	8×8	10×10
number of filters	36	64	100
T	3	4	6
PSNR(db)	33.65	34.31	34.34

E. Training record

There are two training phases for training ORDSR. The network is first initialized as DCT-DSR for training the residual network in the DCT domain. Then the trained DCT-DSR is trained with constraints, e.g. Orthogonality Constraints and Complexity Order Constraints. Using abundant training data, following the same setups as described in the main manuscript,

[2]For simplicity, we only count in convolutional filter parameters which has major influence on the network size.

Fig. 6 shows the plot of the cost function evaluation against training epoch with the constraints in place. As it shown in the table, 8×8 CDCT layer gives the best performance. The learning rate is initialized to 10^4 for all layers and decreases half for every 50 epochs. During the training 128 training patch pairs with the size of 40×40 are randomly extracted.

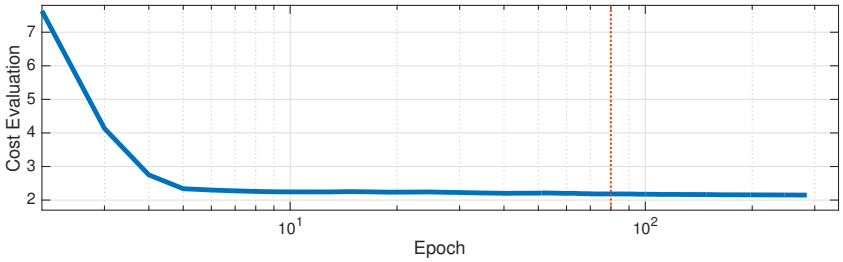


Fig. 6: Cost function evaluation during the training process.

REFERENCES

[1] R. Gonzalez and P. Wintz, "Digital image processing," 1977.

[2] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 38, no. 2, pp. 295–307, 2016.

[3] J. Kim, J. K. Lee, and K. M. Lee, "Accurate image super-resolution using very deep convolutional networks," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2016, pp. 1646–1654.

[4] Y. B. Y. LeCun, L. Bottou and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[5] S. Schulter, C. Leistner, and H. Bischof, "Fast and accurate image upscaling with super-resolution forests," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2015, pp. 3791–3799.

[6] M. Abadi, A. Agarwal, and P. B. et. al., "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. [Online]. Available: http://tensorflow.org/

[7] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2016, pp. 1874–1883.