FYP Presentation

IMAGE DEBLURRING

Image Upsampling via Tight Frames

LEE Sing Chun

Faculty of Engineering
Department of Information Engineering
The Chinese University of Hong Kong

Agenda

Agenda

- Introduction

Agenda

- Introduction
- Tight frames

Agenda

- Introduction
- Tight frames
- Multi-resolution analysis

Agenda

- Introduction
- Tight frames
- Multi-resolution analysis
- Proposed Algorithm

Agenda

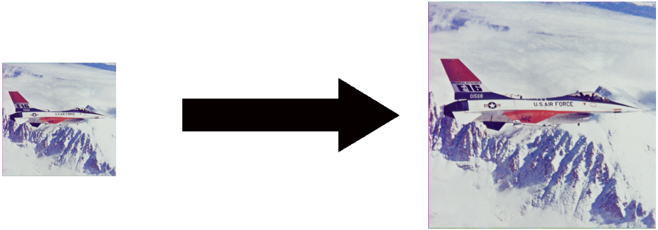
- Introduction
- Tight frames
- Multi-resolution analysis
- Proposed Algorithm
- Results

Agenda

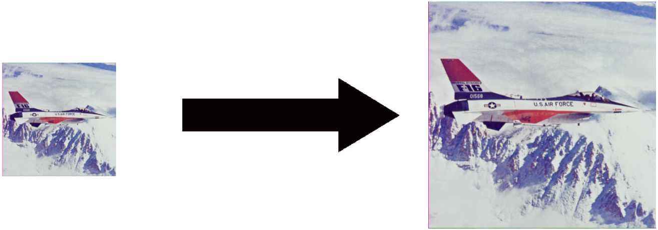
- Introduction
- Tight frames
- Multi-resolution analysis
- Proposed Algorithm
- Results
- Conclusion and future directions

Single Image Upsampling

Single Image Upsampling

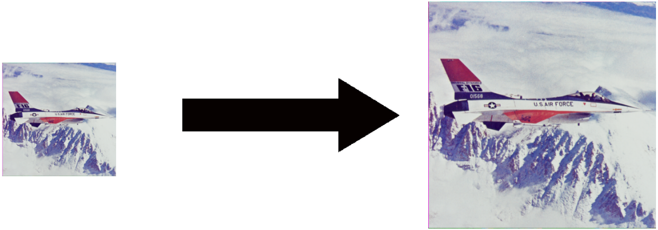


Single Image Upsampling



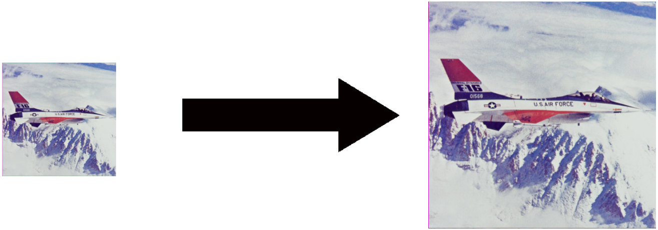
- Given an input image $\mathbf{f}_{input}$,

Single Image Upsampling



- Given an input image $\mathbf{f}_{input}$,
- let say, of 256 x 256 pixels;

Single Image Upsampling



- Given an input image $\mathbf{f}_{input}$,
- let say, of 256 x 256 pixels;
- Reconstruct a 512 x 512 pixels image $\mathbf{f}_{desired}$.

- Commonly found in all image editing software.

- Commonly found in all image editing software.
- Performed by bicubic interpolation,

- Commonly found in all image editing software.
- Performed by bicubic interpolation,
- because of its implementation efficient.

- Commonly found in all image editing software.
- Performed by bicubic interpolation,
- because of its implementation efficient.
- However, the resulting image are blurry.

- Commonly found in all image editing software.
- Performed by bicubic interpolation,
- because of its implementation efficient.
- However, the resulting image are blurry.
- Therefore, our aims is:

- Commonly found in all image editing software.
- Performed by bicubic interpolation,
- because of its implementation efficient.
- However, the resulting image are blurry.
- Therefore, our aims is:

Perform single image upsampling that preserves ***natural details***.

- Commonly found in all image editing software.
- Performed by bicubic interpolation,
- because of its implementation efficient.
- However, the resulting image are blurry.
- Therefore, our aims is:

Perform single image upsampling that preserves ***natural details***.

- In this thesis, we have used

- Commonly found in all image editing software.
- Performed by bicubic interpolation,
- because of its implementation efficient.
- However, the resulting image are blurry.
- Therefore, our aims is:

Perform single image upsampling that preserves ***natural details***.

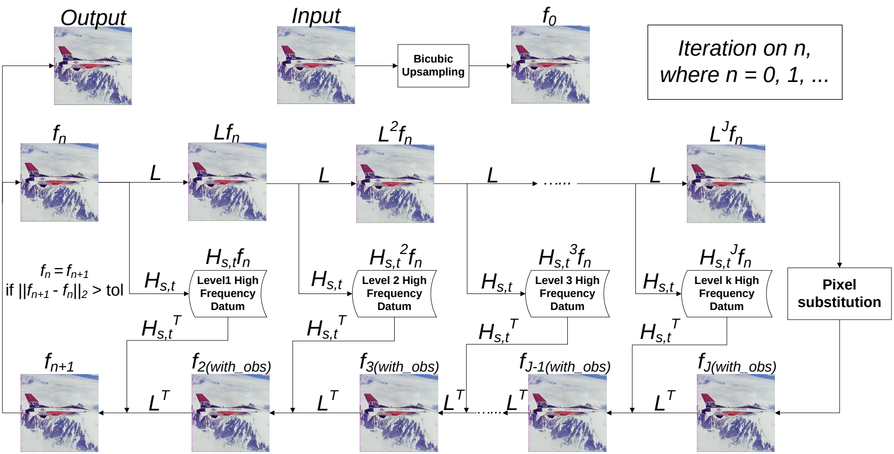
- In this thesis, we have used
- Tight Frame and Multi-resolution Analysis

- Commonly found in all image editing software.
- Performed by bicubic interpolation,
- because of its implementation efficient.
- However, the resulting image are blurry.
- Therefore, our aims is:

Perform single image upsampling that preserves ***natural details***.

- In this thesis, we have used
- Tight Frame and Multi-resolution Analysis
- to design our upsampling algorithm.

Our Proposed Algorithm



- Input is assumed of gray-scale.

- Input is assumed of gray-scale.
- However, real-world images are of RGB scale.

- Input is assumed of gray-scale.
- However, real-world images are of RGB scale.
- To overcome this, using YUV colour space.

- Input is assumed of gray-scale.
- However, real-world images are of RGB scale.
- To overcome this, using YUV colour space.
- RGB is converted to YVU,

- Input is assumed of gray-scale.
- However, real-world images are of RGB scale.
- To overcome this, using YUV colour space.
- RGB is converted to YVU,
- and the Y channel is used as input.

- Input is assumed of gray-scale.
- However, real-world images are of RGB scale.
- To overcome this, using YUV colour space.
- RGB is converted to YVU,
- and the Y channel is used as input.
- V and U channels are upsampled by bicubic interpolation.

- Input is assumed of gray-scale.
- However, real-world images are of RGB scale.
- To overcome this, using YUV colour space.
- RGB is converted to YVU,
- and the Y channel is used as input.
- V and U channels are upsampled by bicubic interpolation.
- Hence, human perception is guaranteed,

- Input is assumed of gray-scale.
- However, real-world images are of RGB scale.
- To overcome this, using YUV colour space.
- RGB is converted to YVU,
- and the Y channel is used as input.
- V and U channels are upsampled by bicubic interpolation.
- Hence, human perception is guaranteed,
- but not image chrominance.

- Input is assumed of gray-scale.
- However, real-world images are of RGB scale.
- To overcome this, using YUV colour space.
- RGB is converted to YVU,
- and the Y channel is used as input.
- V and U channels are upsampled by bicubic interpolation.
- Hence, human perception is guaranteed,
- but not image chrominance.
- In addition, 2-D images are stacked column by column as a 1-D column vector in our algorithm.

Tight Frames

Tight Frames

A system X is called a *tight frames system* of $L_2(\mathbb{R})$ if

$$f = \sum_{g \in X} \langle f, g \rangle \, g, \qquad \forall f \in L^2(\mathbb{R}).$$

Tight Frames

A system X is called a *tight frames system* of $L_2(\mathbb{R})$ if

$$f = \sum_{g \in X} \langle f, g \rangle\, g, \qquad \forall f \in L^2(\mathbb{R}).$$

A tight frames system can be constructed by starting with a scaling function $\phi \in L_2(\mathbb{R})$ and letting it satisfies the dilation equation:

$$\phi(x) = \sum_{n=-l_1}^{l_2} \sqrt{2} h_\phi(n) \phi(2x - n) \qquad \forall x \in \mathbb{R}. \tag{1}$$

- Here h_ϕ are low-pass filter coefficients, with length $(l_2 - l_1)$.

- Here h_ϕ are low-pass filter coefficients, with length $(l_2 - l_1)$.
- Therefore, given a suitable low-pass filter, one can solve (1) to find a scaling function ϕ.

- Here h_ϕ are low-pass filter coefficients, with length $(l_2 - l_1)$.
- Therefore, given a suitable low-pass filter, one can solve (1) to find a scaling function ϕ.

With this ϕ and given corresponding high-pass filters h_ψ with length $(s_2 - s_1)$,

- Here h_ϕ are low-pass filter coefficients, with length $(l_2 - l_1)$.
- Therefore, given a suitable low-pass filter, one can solve (1) to find a scaling function ϕ.

With this ϕ and given corresponding high-pass filters h_ψ with length $(s_2 - s_1)$,

one can find the tight frames function ψ by satisfying a similar dilation equation

$$\psi(x) = \sum_{n=-s_1}^{s_2} \sqrt{2} h_\psi(n) \phi(2x - n) \quad \forall x \in \mathbb{R}. \qquad (2)$$

Hence, given suitable low-pass and high-pass filters, one can construct a tight frames system

$$X_\Psi := \left\{ 2^{k/2}\psi(2^k x - j) \mid \psi \in \Psi; k, j \in \mathbb{Z} \right\}.$$

by solving (1) and (2).

Hence, given suitable low-pass and high-pass filters, one can construct a tight frames system

$$X_\Psi := \left\{ 2^{k/2}\psi(2^k x - j) \mid \psi \in \Psi; k, j \in \mathbb{Z} \right\}.$$

by solving (1) and (2).

In this thesis, piecewise linear B-spline is used to construct tight frames system:

Hence, given suitable low-pass and high-pass filters, one can construct a tight frames system

$$X_\Psi := \left\{ 2^{k/2}\psi(2^k x - j) \mid \psi \in \Psi; k, j \in \mathbb{Z} \right\}.$$

by solving (1) and (2).

In this thesis, piecewise linear B-spline is used to construct tight frames system:

- low-pass: $h_0 = \frac{1}{4}[1, 2, 1]$;

Hence, given suitable low-pass and high-pass filters, one can construct a tight frames system

$$X_\Psi := \left\{ 2^{k/2}\psi(2^k x - j) \mid \psi \in \Psi; k, j \in \mathbb{Z} \right\}.$$

by solving (1) and (2).

In this thesis, piecewise linear B-spline is used to construct tight frames system:

- low-pass: $h_0 = \frac{1}{4}[1, 2, 1]$;
- high-pass: $h_1 = \frac{\sqrt{2}}{4}[1, 0, -1]$ and $h_2 = \frac{1}{4}[-1, 2, -1]$.

For 2-D images, tensor product is used, hence the 2-D filters $h_{s,t}$, $s, t = 0, 1, 2$, are defined by

$$h_{s,t} = h_t \otimes h_s.$$

For 2-D images, tensor product is used, hence the 2-D filters $h_{s,t}$, $s, t = 0, 1, 2$, are defined by

$$h_{s,t} = h_t \otimes h_s.$$

In this thesis, periodic boundary condition is used and filters are represented in their matrix form.

For 2-D images, tensor product is used, hence the 2-D filters $h_{s,t}$, $s, t = 0, 1, 2$, are defined by

$$h_{s,t} = h_t \otimes h_s.$$

In this thesis, periodic boundary condition is used and filters are represented in their matrix form.

e.g. H_0 for h_0 and $H_{0,0}$ for $h_{0,0}$ are written as

$$H_0 = \frac{1}{4}\begin{bmatrix} 2 & 1 & 0 & \cdots & 0 & 1 \\ 1 & 2 & 1 & \ddots & 0 & 0 \\ 0 & 1 & 2 & \ddots & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \ddots & 2 & 1 \\ 1 & 0 & 0 & \cdots & 1 & 2 \end{bmatrix} \quad \text{and} \quad H_{0,0} = H_0^T \otimes H_0.$$

Then, we have this perfect reconstruction identity:

$$\sum_{s,t=0}^{2} H_{s,t}^{T} H_{s,t} = \textit{identity matrix} \tag{3}$$

Then, we have this perfect reconstruction identity:

$$\sum_{s,t=0}^{2} H_{s,t}^{T} H_{s,t} = \textit{identity matrix} \tag{3}$$

Our algorithm is basically designed by using (3).

Multi-resolution Analysis

Multi-resolution Analysis

This tight frames system can generate a multi-resolution analysis.

Multi-resolution Analysis

This tight frames system can generate a multi-resolution analysis.
It can be illustrated by this graph:

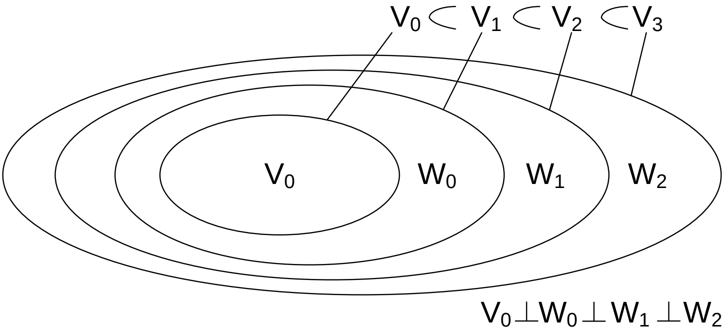


Image Upsampling via Tight Frames

Image Upsampling via Tight Frames

- Assume $\mathbf{f}_{desired}$ is band limited in $L_2(\mathbb{R})$,

Image Upsampling via Tight Frames

- Assume $\mathbf{f}_{desired}$ is band limited in $L_2(\mathbb{R})$,
- and it has a representation in V_J for some $J \in \mathbb{Z}$.

Image Upsampling via Tight Frames

- Assume $\mathbf{f}_{desired}$ is band limited in $L_2(\mathbb{R})$,
- and it has a representation in V_J for some $J \in \mathbb{Z}$.
- After J times convolution with low-pass filter $h_{0,0}$,

Image Upsampling via Tight Frames

- Assume $\mathbf{f}_{desired}$ is band limited in $L_2(\mathbb{R})$,
- and it has a representation in V_J for some $J \in \mathbb{Z}$.
- After J times convolution with low-pass filter $h_{0,0}$,
- $(H_{0,0})^J \mathbf{f}_{desired}$ is in V_0.

Image Upsampling via Tight Frames

- Assume $\mathbf{f}_{desired}$ is band limited in $L_2(\mathbb{R})$,
- and it has a representation in V_J for some $J \in \mathbb{Z}$.
- After J times convolution with low-pass filter $h_{0,0}$,
- $(H_{0,0})^J \mathbf{f}_{desired}$ is in V_0.
- Using the downsampling operator, $D_{m,n}$, $m, n = 0, 1$, defined by [Bose, 1998],

Image Upsampling via Tight Frames

- Assume $\mathbf{f}_{desired}$ is band limited in $L_2(\mathbb{R})$,
- and it has a representation in V_J for some $J \in \mathbb{Z}$.
- After J times convolution with low-pass filter $h_{0,0}$,
- $(H_{0,0})^J \mathbf{f}_{desired}$ is in V_0.
- Using the downsampling operator, $D_{m,n}$, $m, n = 0, 1$, defined by [Bose, 1998],
- We assume $D_{0,0}^T \mathbf{f}_{input} = D_{0,0}^T D_{0,0} (H_{0,0})^J \mathbf{f}_{desired}$.

Image Upsampling via Tight Frames

- Assume $\mathbf{f}_{desired}$ is band limited in $L_2(\mathbb{R})$,
- and it has a representation in V_J for some $J \in \mathbb{Z}$.
- After J times convolution with low-pass filter $h_{0,0}$,
- $(H_{0,0})^J \mathbf{f}_{desired}$ is in V_0.
- Using the downsampling operator, $D_{m,n}$, $m, n = 0, 1$, defined by [Bose, 1998],
- We assume $D_{0,0}^T \mathbf{f}_{input} = D_{0,0}^T D_{0,0} (H_{0,0})^J \mathbf{f}_{desired}$.
- By multi-resolution analysis,

$$V_J = W_J \oplus W_{J-1} \oplus \cdots \oplus W_0 \oplus V_0.$$

- $\mathbf{f}_{desired} \in V_J$ can be split into

- $\mathbf{f}_{desired} \in V_J$ can be split into
  1. high frequency parts in W_k, $0 \leq k \leq J$;

- $\mathbf{f}_{desired} \in V_J$ can be split into
  1. high frequency parts in W_k, $0 \leq k \leq J$;
  2. and $\mathbf{f}_{input} \in V_0$.

- $\mathbf{f}_{desired} \in V_J$ can be split into
  1. high frequency parts in W_k, $0 \leq k \leq J$;
  2. and $\mathbf{f}_{input} \in V_0$.
- Deductively applying the perfect reconstruction identity (3), we have

- $\mathbf{f}_{desired} \in V_J$ can be split into
  1. high frequency parts in W_k, $0 \leq k \leq J$;
  2. and $\mathbf{f}_{input} \in V_0$.
- Deductively applying the perfect reconstruction identity (3), we have

$$\mathbf{f}_{desired} = \sum_{s,t=0}^{2} H_{s,t}^T H_{s,t} \mathbf{f}_{desired}$$

- $\mathbf{f}_{desired} \in V_J$ can be split into
  1. high frequency parts in W_k, $0 \le k \le J$;
  2. and $\mathbf{f}_{input} \in V_0$.
- Deductively applying the perfect reconstruction identity (3), we have

$$\begin{aligned}\mathbf{f}_{desired} &= \sum_{s,t=0}^{2} H_{s,t}^T H_{s,t}\mathbf{f}_{desired} \\ &= H_{0,0}^T H_{0,0}\mathbf{f}_{desired} + \sum_{\substack{s,t=0 \\ (s,t)\neq(0,0)}}^{2} H_{s,t}^T H_{s,t}\mathbf{f}_{desired}\end{aligned}$$

$$= \quad (H_{0,0}^T)^2 (H_{0,0})^2 \mathbf{f}_{desired} + \sum_{j=0}^{1} \sum_{\substack{s,t=0 \\ (s,t)\neq(0,0)}}^{2} (H_{0,0}^T)^j H_{s,t}^T H_{s,t} (H_{0,0})^j \mathbf{f}_{desired}$$

$$
\begin{aligned}
&= (H_{0,0}^T)^2 (H_{0,0})^2 \mathbf{f}_{desired} + \sum_{j=0}^{1} \sum_{\substack{s,t=0 \\ (s,t)\neq(0,0)}}^{2} (H_{0,0}^T)^j H_{s,t}^T H_{s,t} (H_{0,0})^j \mathbf{f}_{desired} \\
&= \cdots
\end{aligned}
$$

$$
\begin{aligned}
&= (H_{0,0}^T)^2 (H_{0,0})^2 \mathbf{f}_{desired} + \sum_{j=0}^{1} \sum_{\substack{s,t=0 \\ (s,t)\neq(0,0)}}^{2} (H_{0,0}^T)^j H_{s,t}^T H_{s,t} (H_{0,0})^j \mathbf{f}_{desired} \\
&= \cdots \\
&= (H_{0,0}^T)^J (H_{0,0})^J \mathbf{f}_{desired} + \sum_{j=0}^{J-1} \sum_{\substack{s,t=0 \\ (s,t)\neq(0,0)}}^{2} (H_{0,0}^T)^j H_{s,t}^T H_{s,t} (H_{0,0})^j \mathbf{f}_{desired}.
\end{aligned}
$$

$$
\begin{aligned}
&= (H_{0,0}^T)^2(H_{0,0})^2\mathbf{f}_{desired} + \sum_{j=0}^{1}\sum_{\substack{s,t=0\\(s,t)\neq(0,0)}}^{2}(H_{0,0}^T)^j H_{s,t}^T H_{s,t}(H_{0,0})^j\mathbf{f}_{desired}\\
&= \cdots\\
&= (H_{0,0}^T)^J(H_{0,0})^J\mathbf{f}_{desired} + \sum_{j=0}^{J-1}\sum_{\substack{s,t=0\\(s,t)\neq(0,0)}}^{2}(H_{0,0}^T)^j H_{s,t}^T H_{s,t}(H_{0,0})^j\mathbf{f}_{desired}.
\end{aligned}
$$

By our assumption: $D_{0,0}^T D_{0,0}(H_{0,0})^J\mathbf{f}_{desired} = D_{0,0}^T\mathbf{f}_{input}$,in V_0,

$$
\begin{aligned}
&= (H_{0,0}^T)^2 (H_{0,0})^2 \mathbf{f}_{desired} + \sum_{j=0}^{1} \sum_{\substack{s,t=0 \\ (s,t)\neq(0,0)}}^{2} (H_{0,0}^T)^j H_{s,t}^T H_{s,t} (H_{0,0})^j \mathbf{f}_{desired} \\
&= \cdots \\
&= (H_{0,0}^T)^J (H_{0,0})^J \mathbf{f}_{desired} + \sum_{j=0}^{J-1} \sum_{\substack{s,t=0 \\ (s,t)\neq(0,0)}}^{2} (H_{0,0}^T)^j H_{s,t}^T H_{s,t} (H_{0,0})^j \mathbf{f}_{desired}.
\end{aligned}
$$

By our assumption: $D_{0,0}^T D_{0,0} (H_{0,0})^J \mathbf{f}_{desired} = D_{0,0}^T \mathbf{f}_{input}$,in V_0,

$$
\begin{aligned}
\mathbf{f}_{desired} &= (H_{0,0}^T)^J \sum_{m,n=0}^{1} D_{m,n}^T D_{m,n} (H_{0,0})^J \mathbf{f}_{desired} \\
&\quad + \sum_{j=0}^{J-1} \sum_{\substack{s,t=0 \\ (s,t)\neq(0,0)}}^{2} (H_{0,0}^T)^j H_{s,t}^T H_{s,t} (H_{0,0})^j \mathbf{f}_{desired}
\end{aligned}
$$

$$
\begin{aligned}
= \quad & (H_{0,0}^T)^J D_{0,0}^T D_{0,0} (H_{0,0}^T)^J \mathbf{f}_{desired} + (H_{0,0}^T)^J \sum_{\substack{m,n=0 \\ (m,n)\neq(0,0)}}^{1} D_{m,n}^T D_{m,n} (H_{0,0}^T)^J \mathbf{f}_{desired} \\
& + \sum_{j=0}^{J-1} \sum_{\substack{s,t=0 \\ (s,t)\neq(0,0)}}^{2} (H_{0,0}^T)^j H_{s,t}^T H_{s,t} (H_{0,0})^j \mathbf{f}_{desired}
\end{aligned}
$$

$$
\begin{aligned}
= \quad & (H_{0,0}^T)^J D_{0,0}^T D_{0,0} (H_{0,0}^T)^J \mathbf{f}_{desired} + (H_{0,0}^T)^J \sum_{\substack{m,n=0 \\ (m,n)\neq(0,0)}}^{1} D_{m,n}^T D_{m,n} (H_{0,0}^T)^J \mathbf{f}_{desired} \\
& + \sum_{j=0}^{J-1} \sum_{\substack{s,t=0 \\ (s,t)\neq(0,0)}}^{2} (H_{0,0}^T)^j H_{s,t}^T H_{s,t} (H_{0,0})^j \mathbf{f}_{desired} \\
= \quad & (H_{0,0}^T)^J D_{0,0}^T \mathbf{f}_{input} + (H_{0,0}^T)^J \sum_{\substack{m,n=0 \\ (m,n)\neq(0,0)}}^{1} D_{m,n}^T D_{m,n} (H_{0,0}^T)^J \mathbf{f}_{desired} \\
& + \sum_{j=0}^{J-1} \sum_{\substack{s,t=0 \\ (s,t)\neq(0,0)}}^{2} (H_{0,0}^T)^j H_{s,t}^T H_{s,t} (H_{0,0})^j \mathbf{f}_{desired}
\end{aligned}
$$

So,

So,

$$\mathbf{f}_{desired} = (H_{0,0}^T)^J D_{0,0}^T \mathbf{f}_{input} + (H_{0,0}^T)^J \sum_{\substack{m,n=0 \\ (m,n)\neq(0,0)}}^{1} D_{m,n}^T D_{m,n} (H_{0,0})^J \mathbf{f}_{desired}$$
$$+ \sum_{j=0}^{J-1} \sum_{\substack{s,t=0 \\ (s,t)\neq(0,0)}}^{2} (H_{0,0}^T)^j H_{s,t}^T H_{s,t} (H_{0,0})^j \mathbf{f}_{desired} \quad (4)$$

So,

$$\mathbf{f}_{desired} = (H_{0,0}^T)^J D_{0,0}^T \mathbf{f}_{input} + (H_{0,0}^T)^J \sum_{\substack{m,n=0 \\ (m,n)\neq(0,0)}}^{1} D_{m,n}^T D_{m,n} (H_{0,0})^J \mathbf{f}_{desired}$$
$$+ \sum_{j=0}^{J-1} \sum_{\substack{s,t=0 \\ (s,t)\neq(0,0)}}^{2} (H_{0,0}^T)^j H_{s,t}^T H_{s,t} (H_{0,0})^j \mathbf{f}_{desired} \quad (4)$$

And our proposed algorithm is iterating on this equation (4) as:

$$\mathbf{f}_{n+1} = (H_{0,0}^T)^J D_{0,0}^T \mathbf{f}_{input} + (H_{0,0}^T)^J \sum_{\substack{m,n=0 \\ (m,n)\neq(0,0)}}^{1} D_{m,n}^T D_{m,n} (H_{0,0})^J \mathbf{f}_{n}$$
$$+ \sum_{j=0}^{J-1} \sum_{\substack{s,t=0 \\ (s,t)\neq(0,0)}}^{2} (H_{0,0}^T)^j H_{s,t}^T H_{s,t} (H_{0,0})^j \mathbf{f}_{n} \quad (5)$$

The step of using our is:

The step of using our is:

1. Set an initial guess, $\mathbf{f}_0$, as the bicubic upsampling of $\mathbf{f}_{input}$ and *tol* as the tolerance allowed.

The step of using our is:

1. Set an initial guess, $\mathbf{f}_0$, as the bicubic upsampling of $\mathbf{f}_{input}$ and *tol* as the tolerance allowed.
2. Set $\mathbf{f}_n = \mathbf{f}_0$.

The step of using our is:

1. Set an initial guess, $\mathbf{f}_0$, as the bicubic upsampling of $\mathbf{f}_{input}$ and *tol* as the tolerance allowed.
2. Set $\mathbf{f}_n = \mathbf{f}_0$.
3. Find $\mathbf{f}_{n+1}$ by the equation (5)

The step of using our is:

1. Set an initial guess, $\mathbf{f}_0$, as the bicubic upsampling of $\mathbf{f}_{input}$ and *tol* as the tolerance allowed.
2. Set $\mathbf{f}_n = \mathbf{f}_0$.
3. Find $\mathbf{f}_{n+1}$ by the equation (5)
4. Set $\mathbf{f}^* = \mathbf{f}_{n+1}$ if $\|\mathbf{f}_{n+1} - \mathbf{f}_n\|_2 \leq tol$, where $\mathbf{f}^*$ is our upsampled image; else set $\mathbf{f}_n = \mathbf{f}_{n+1}$ and repeat step 3 and 4.

Results

Results

- Click Here (Hyper Link)
- Local Files:

  ->

Bicubic

TF Level 1

Bicubic

TF Level 4

Bicubic

TF Level 7

[Shan, 2009]

TF Level 1

[Shan, 2009]

TF Level 4

[Shan, 2009]

TF Level 7

Figure:
By [Shan,2009], Level 1, Level 2 and Level 3 (Upper: From left to right)
Level 4, Level 5, Level 6 and Level 7 (Lower: From left to right)

Bicubic

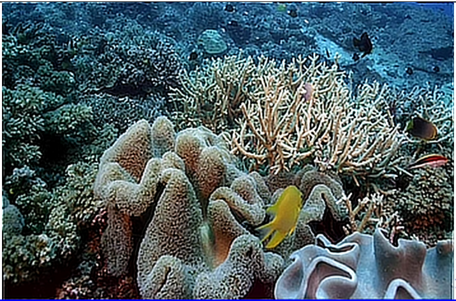
TF Level 1

Bicubic

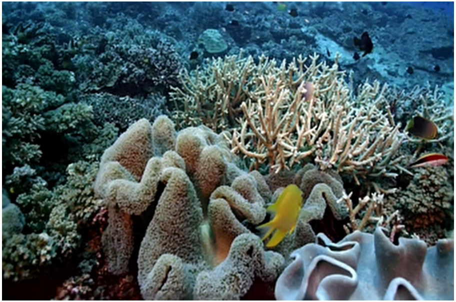
TF Level 4

Bicubic

TF Level 7

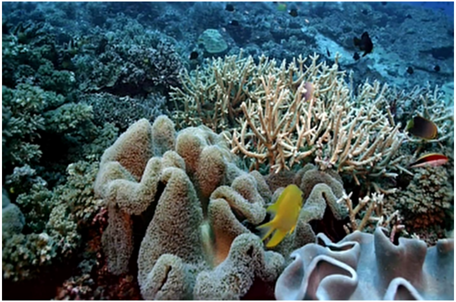


[Shan, 2009]



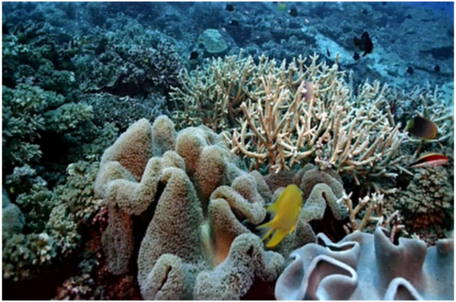
TE Level 1

[Shan, 2009]



TF Level 4

[Shan, 2009]



TF Level 7

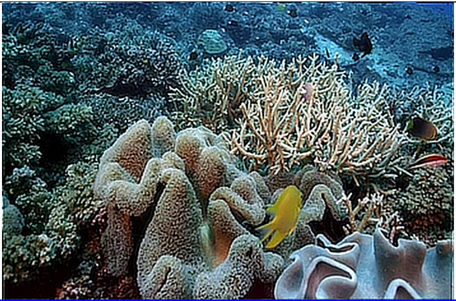


Figure:
By [Shan,2009], Level 1, Level 2 and Level 3 (Upper: From left to right)
Level 4, Level 5, Level 6 and Level 7 (Lower: From left to right)

Bicubic

TF Level 1

Bicubic

TF Level 4

Bicubic

TF Level 7

[Shan, 2009]

TF Level 1

[Shan, 2009]

TF Level 4

[Shan, 2009]

TF Level 7

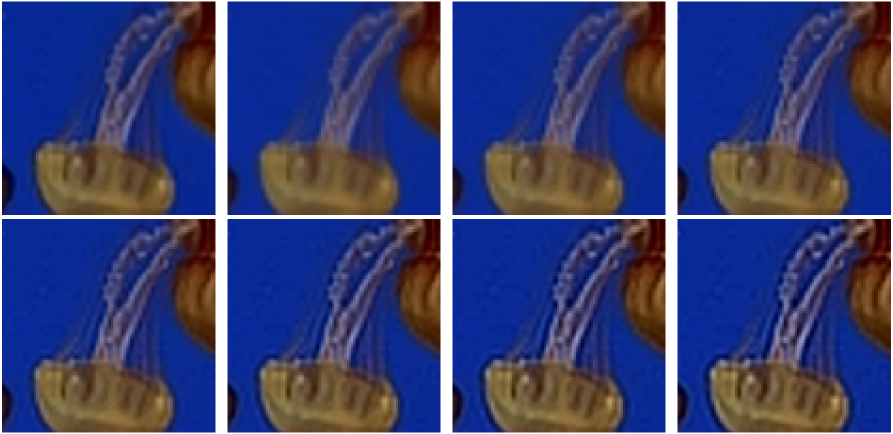


Figure:
By [Shan,2009], Level 1, Level 2 and Level 3 (Upper: From left to right)
Level 4, Level 5, Level 6 and Level 7 (Lower: From left to right)

Conclusions and Future Directions

Conclusions and Future Directions

- For upsampling 512 x 512 to 1024 x 1024

Conclusions and Future Directions

- For upsampling 512 x 512 to 1024 x 1024
- Computation time (on 1.66GHz CPU N280 AUSU EEEPC, Matlab 2009b):

Conclusions and Future Directions

- For upsampling 512 x 512 to 1024 x 1024
- Computation time (on 1.66GHz CPU N280 AUSU EEEPC, Matlab 2009b):

Table: Computation time

Level	1	2	3	4	5	6	7
Time taken	79s	292s	10 mins	19 mins	36 mins	67 mins	112 mins

Conclusions and Future Directions

- For upsampling 512 x 512 to 1024 x 1024
- Computation time (on 1.66GHz CPU N280 AUSU EEEPC, Matlab 2009b):

Table: Computation time

Level	1	2	3	4	5	6	7
Time taken	79s	292s	10 mins	19 mins	36 mins	67 mins	112 mins

- Lower level: faster than [Shan,2009] ($\approx$ 13 mins)

Conclusions and Future Directions

- For upsampling 512 x 512 to 1024 x 1024
- Computation time (on 1.66GHz CPU N280 AUSU EEEPC, Matlab 2009b):

Table: Computation time

Level	1	2	3	4	5	6	7
Time taken	79s	292s	10 mins	19 mins	36 mins	67 mins	112 mins

- Lower level: faster than [Shan,2009] ($\approx$ 13 mins)
- Higher level: enhanced visual perception

- For a fixed level (J) reconstruction,

- For a fixed level (J) reconstruction,
- it is parameter-free.

- For a fixed level (J) reconstruction,
- it is parameter-free.
- Automatically upsampling images.

- For a fixed level (J) reconstruction,
- it is parameter-free.
- Automatically upsampling images.
- In the future,

- For a fixed level (J) reconstruction,
- it is parameter-free.
- Automatically upsampling images.
- In the future,
- Natural Science statistical distribution.

- For a fixed level (J) reconstruction,
- it is parameter-free.
- Automatically upsampling images.
- In the future,
- Natural Science statistical distribution.
- As prior information of images.

- For a fixed level (J) reconstruction,
- it is parameter-free.
- Automatically upsampling images.
- In the future,
- Natural Science statistical distribution.
- As prior information of images.
- Proof of local convergence.

Thanks for your attentions!

Q & A