Supplementary file for "Learning Spatial Dependent Dictionary with Efficient Multicative Gaussian Process"

1 Gibbs sampling procedures

1.1 GP-FA model

We denote i th $q \times q$ patch as $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\} \in \mathbb{R}^{J \times N}$ $(J = q \times q)$, $\boldsymbol{D} = \{\boldsymbol{d}_1, \ldots, \boldsymbol{d}_M\} \in \mathbb{R}^{J \times M}$ is dictionary, $\boldsymbol{z}_i = \{z_{i1}, \ldots, z_{iM}\} \in \{0, 1\}^M$ is the factor binary scores, indicating whether a dictionary is present in this patch. $\boldsymbol{w}_i = \{w_{i1}, \ldots, w_{iM}\} \in \mathbb{R}^M$ are the real-valued weights complement the binary scores.

$$\begin{aligned}
\boldsymbol{x}_i &= \boldsymbol{D}(\boldsymbol{w}_i \odot \boldsymbol{z}_i) + \boldsymbol{\varepsilon}_i, \\
\boldsymbol{d}_m &\sim \mathcal{N}(0, \boldsymbol{I}_J) \\
\boldsymbol{w}_i &\sim \mathcal{N}(0, \sigma_w^2 \boldsymbol{I}_M) \\
\boldsymbol{\varepsilon}_i &\sim \mathcal{N}(0, \sigma_\varepsilon^2 \boldsymbol{I}_J) \\
\sigma_w^{-2} &\sim \Gamma(c_0, d_0) \\
\sigma_\varepsilon^{-2} &\sim \Gamma(e_0, f_0)
\end{aligned}$$

We assign 2D grid Gaussian process prior over patches.

$$\begin{aligned}
z_{im} &\sim \text{Bernoulli}(\sigma(y_{im})) \\
y_{im} &= \boldsymbol{f}_m(\boldsymbol{l}_i) \\
\boldsymbol{f}_m(\cdot) &\sim \mathcal{GP}(b_m, K_m(\cdot)) \\
b_m &\sim \mathcal{N}(\lambda_m, \sigma_b^2) \\
\lambda_m &\sim \mathcal{N}_-(0, \sigma_\lambda^2)
\end{aligned}$$

For y_i, using polya-gamma data augmentation.

$$\gamma_{im} \sim \mathcal{PG}(1, y_{im} + b_m)$$

For clarity we omit dictionary index m. Denoting $\boldsymbol{y}_{(-i)} \triangleq \boldsymbol{y} \backslash \{y_i\}$,

$$\begin{aligned}
y_i|- &\sim \mathcal{N}(\mu_*, \sigma_*), \qquad (1) \\
\mu_* &= \left(\frac{\boldsymbol{k}_{i,\backslash i} \boldsymbol{K}_{\backslash i,\backslash i}^{-1} \boldsymbol{y}_{\backslash i}^T}{k_{i,i} - \boldsymbol{k}_{i,\backslash i} \boldsymbol{K}_{\backslash i,\backslash i}^{-1} \boldsymbol{k}_{i,\backslash i}^T} + z_i - \frac{1}{2} - \gamma_i b \right) \sigma_*^2, \\
\sigma_*^2 &= \left(\frac{1}{k_{i,i} - \boldsymbol{k}_{i,\backslash i} \boldsymbol{K}_{\backslash i,\backslash i}^{-1} \boldsymbol{K}_{i,\backslash i}^T} + \gamma_i \right)^{-1}, \\
\boldsymbol{K} &= \begin{bmatrix} k_{i,i} & \boldsymbol{k}_{i,\backslash i} \\ \boldsymbol{k}_{i,\backslash i}^T & \boldsymbol{K}_{\backslash i,\backslash i} \end{bmatrix},
\end{aligned}$$

For b,

$$\begin{aligned} b &\sim \mathcal{N}(\mu_{b*}, \sigma_{b*}^2) \\ \sigma_{b*}^2 &= \left[\sum_i \gamma_i + \sigma_b^{-2}\right]^{-1} \\ \mu_{b*} &= \sigma_{b*}^2 \left[\sum_{i,n} \left(z_{i,m}^n - \frac{1}{2} - \gamma_{1,m}^{(i,n)} y_i^n\right) + \lambda\sigma_b^{-2}\right] \end{aligned}$$

For λ,

$$\begin{aligned} \lambda &\sim \mathcal{N}_-(\mu_{\lambda*}, \sigma_{\lambda*}^2) \\ \sigma_{\lambda*}^2 &= [\sigma_\lambda^{-2} + \sigma_b^{-2}]^{-1} \\ \mu_{\lambda*} &= \lambda\sigma_{\lambda*}^2\sigma_b^{-2} \end{aligned}$$

For dictionary element d_{jm}, letting $s_{im} \triangleq w_{im} \times z_{im}$, we can obtain

$$\begin{aligned} p(d_{jm}|-) &\sim \mathcal{N}((\mu_d)_{jm}^*, (\sigma_d^2)_{jm}^*) \\ (\sigma_d^2)_{jm}^* &= \left[1 + \sum_i (s_{im})^2/\sigma_\varepsilon^2\right]^{-1} \\ (\mu_d)_{jm}^* &= \frac{(\sigma_d^2)_{jm}^*}{\sigma_\varepsilon^2} \sum_i s_{im} \left(x_{ij} - \sum_{m' \neq m} d_{jm'} s_{m'i}\right) \end{aligned}$$

For w_{im},

$$\begin{aligned} p(w_{im}|-) &\sim \mathcal{N}((\mu_w)_{im}^*, (\sigma_w^2)_{im}^*) \\ (\sigma_w^2)_{im}^* &= \left[\sigma_w^{-2} + \left(\sum_j d_{jm} z_{im}\right)^2 / \sigma_\varepsilon^2\right]^{-1} \\ (\mu_w)_{im}^* &= \frac{(\sigma_w^2)_{im}^*}{\sigma_\varepsilon^2} z_{im} \sum_j d_{jm} \left(x_{ij} - \sum_{m' \neq m} d_{jm'} s_{m'i}\right) \end{aligned}$$

For z_{im},

$$\begin{aligned} p(z_{im}|-) &\sim \text{Bernoulli}(p_{im}^*/(1 + p_{im}^*)) \\ p_{im}^* &= \exp\left[\frac{1}{2\sigma_\varepsilon^2} \sum_{j=1}^J 2 \left(x_{ij} - \sum_{m' \neq m} d_{jm'} s_{im'}\right) \times d_{jm} w_{im} - \frac{1}{2\sigma_\varepsilon^2} (d_{jm} w_{im})^2 + y_{im} + b_m\right] \end{aligned}$$

For σ_w^{-2},

$$\begin{aligned} \sigma_w^{-2} &\sim \Gamma(c^*, d^*) \\ c^* &= c_0 + \frac{1}{2} MN \\ d^* &= d_0 + \frac{1}{2} \sum_{i=1}^N \sum_{m=1}^M (w_{im})^2 \end{aligned}$$

For σ_ε^{-2},

$$\begin{aligned} \sigma_\varepsilon^{-2} &\sim \Gamma(e^*, f^*) \\ e^* &= e_0 + \frac{1}{2} JN \\ f^* &= f_0 + \frac{1}{2} \sum_{j=1}^J \sum_{i=1}^N \left(x_{ij} - \sum_{m=1}^M d_{jm} s_{im}\right)^2 \end{aligned}$$

1.2 GP-SBN-FA model

Here we provide the gibbs sampling scheme involving only one layer of sigmoid belief network. The multilayer SBN can be easily derived following same strategy.

Denoting $\boldsymbol{H} = \{\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N\} \in \{0,1\}^{N \times K_1}$, where K_1 is the number of nodes in hidden layer. $\boldsymbol{V} = (\boldsymbol{V}_1, \ldots, \boldsymbol{V}_M) \in \mathbb{R}^{K \times K_1}$ is the weight matrix. Using Pólya-gamma data augmentation, we can present the model as below

$$\begin{aligned} p(\boldsymbol{Z}, \boldsymbol{H}, \boldsymbol{\gamma}^{(0,1)}, \boldsymbol{V}, \boldsymbol{b}^{(0,1)}) \quad \propto \quad & \exp\left\{\sum_{i=1}^{N}\sum_{m=1}^{M}\left(z_{im} - \frac{1}{2}\right)(\boldsymbol{V}_m^T\boldsymbol{h}_i + b_m^{(0)}) - \frac{1}{2}\gamma_{im}^{(0)}(\boldsymbol{V}_m^T\boldsymbol{h}_i + b_m^{(0)})^2\right\} \cdot \\ & \exp\left\{\sum_{i=1}^{N}\sum_{m_1=1}^{K_1}\left(h_{im_1} - \frac{1}{2}\right)(b_{m_1}^{(1)})^2 - \frac{1}{2}\gamma_{im_1}^{(1)}(b_{m_1}^{(1)})^2\right\} \\ & \cdot p_0(\boldsymbol{\gamma}^{(0,1)}, \boldsymbol{V}, \boldsymbol{b}^{(0,1)}) \end{aligned}$$

The gibbs update for z_{im} is given by,

$$\begin{aligned} p(z_{im}|-) \quad &\sim \quad \text{Bernoulli}(p_{im}^*/(1+p_{im}^*)) \\ p_{im}^* \quad &= \quad \exp\left[\frac{1}{2\sigma_\varepsilon^2}\sum_{j=1}^{J} 2\left(x_{ij} - \sum_{m' \neq m} d_{jm's_{im'}}\right) \times d_{jm}w_{im} - \frac{1}{2\sigma_\varepsilon^2}(d_{jm}w_{im})^2 + \boldsymbol{V}_m^T\boldsymbol{h}_i + b_m^{(0)}\right] \end{aligned}$$

For $\gamma_{im}^{(0)}$ and $\gamma_{im_1}^{(1)}$,

$$\begin{aligned} \gamma_{im}^{(0)} \quad &\sim \quad \mathcal{PG}(1, \boldsymbol{V}_m^T\boldsymbol{h}_i + b_m^{(0)}) \\ \gamma_{im_1}^{(1)} \quad &\sim \quad \mathcal{PG}(1, b_{m_1}^{(1)}) \end{aligned}$$

For $\boldsymbol{V}_m$

$$\begin{aligned} \boldsymbol{V}_m \quad &\sim \quad \mathcal{N}(\boldsymbol{\mu}_V, \sigma_V^2) \\ \sigma_V^2 \quad &= \quad \left[\sum_i \gamma_{im}^{(0)}\boldsymbol{h}_i\boldsymbol{h}_i^T + 1\right]^{-1} \\ \boldsymbol{\mu}_V \quad &= \quad \sigma_V^2\left[\sum_i\left(z_{im} - \frac{1}{2} - \gamma_{im}^{(0)}b_m^{(0)}\right)\boldsymbol{h}_i\right] \end{aligned}$$

For $\boldsymbol{b}^{(0)}, \boldsymbol{b}^{(1)}$.

$$\begin{aligned} b_m^{(0)} \quad &\sim \quad \mathcal{N}((\mu_{b_0})_m, (\sigma_{b_0}^2)_m) \\ (\sigma_{b_0}^2)_m \quad &= \quad \left[\sum_i \gamma_{im}^{(0)} + 1\right]^{-1} \\ (\mu_{b_0})_m \quad &= \quad (\sigma_{b_0}^2)_m\sum_i\left(z_{im} - \frac{1}{2} - \gamma_{im}^{(0)}\boldsymbol{V}_m^T\boldsymbol{h}_i\right) \\ b_{m_1}^{(1)} \quad &\sim \quad \mathcal{N}((\mu_{b_1})_{m_1}, (\sigma_{b_1}^2)_{m_1}) \\ (\sigma_{b_1}^2)_{m_1} \quad &= \quad \left[\sum_i \gamma_{im_1}^{(1)} + 1\right]^{-1} \\ (\mu_{b_1})_{m_1} \quad &= \quad (\sigma_{b_1}^2)_{m_1}\sum_i\left(h_{im_1} - \frac{1}{2}\right) \end{aligned}$$

For h_{im_1},

$$\begin{aligned} h_{im_1} &\sim Bernoulli(\sigma(u_{im_1})) \\ u_{im_1} &= b_{m_1}^{(1)} + \sum_{m=1}^{M} \left(z_{im} - \frac{1}{2} \right) V_{mm_1} - \frac{1}{2} \gamma_{im}^{(0)} (2 t_{imm_1} V_{mm_1} + V_{mm_1}^2) \\ t_{imm_1} &= \boldsymbol{V}_m^T \boldsymbol{h}_i + b_m^{(0)} - V_{mm_1} h_{im_1} \end{aligned}$$

2 Kronecker method

Suppose our aim is to employ Kronecker method for fast inference of $\boldsymbol{u}\boldsymbol{K}_{\setminus i,\setminus i}^{-1}$, where $\boldsymbol{u}$ is a vector of size $N-1$.

Algorithm 1 Preconditioned conjugate gradient for multiplicative GP inference

Input: Gram matrix for dimension up to S: $\boldsymbol{K}^{(1)} \dots \boldsymbol{K}^{(S)}$, current patch index $\boldsymbol{l}_i = (l_i^{(1)} \dots l_i^{(S)})$ and an input vector of size $N-1$: $\boldsymbol{u}$
Output: $\boldsymbol{u}\boldsymbol{K}_{\setminus i,\setminus i}^{-1}$, where $\boldsymbol{K}_{\setminus i,\setminus i}$ is obtained by removing i th row and column from the Kronecker product gram matrix $\boldsymbol{K} = \boldsymbol{K}^{(1)} \otimes \dots \otimes \boldsymbol{K}^{(s)}$.
Set conditioner matrix $\boldsymbol{C} = \text{diag}(c_1 \dots c_N)$, where $c_i = \lambda^{-1/2}, c_{\setminus i} = \eta$; For numerical reason, we set η and λ to be large number, such as 10^{10}
Initialize $\boldsymbol{x}_0$, $\boldsymbol{r}_0 = \boldsymbol{u}$, $\boldsymbol{z}_0 = \boldsymbol{C}\boldsymbol{u}$ and index $k = 0$
while $\boldsymbol{r}_k >$ tolerance **do**
  $k = k + 1$
  if $k == 1$ **then**
    $\boldsymbol{p}_1 = \boldsymbol{z}_0$
  else
    $\boldsymbol{p}_k = \boldsymbol{z}_{k-1} + \frac{\boldsymbol{z}_{k-1}^T \boldsymbol{r}_{k-1}}{\boldsymbol{z}_{k-2}^T \boldsymbol{r}_{k-2}} \boldsymbol{p}_{k-1}$
  end if
  $\boldsymbol{L} = \boldsymbol{p}_k^T$
  for $s = 1$ to S do **do**
    $\boldsymbol{L}$ =reshape($\boldsymbol{L}$, size($\boldsymbol{K}^{(s)}$),N/ size($\boldsymbol{K}^{(s)}$))
    $\boldsymbol{L} = (\boldsymbol{K}^{(s)} \boldsymbol{L})^T$
    $\boldsymbol{L} = \text{vec}(\boldsymbol{L})$
  end for
  $L_i = L_i + \lambda v_i$
  Compute $\alpha = \frac{\boldsymbol{z}_{k-1}^T \boldsymbol{r}_{k-1}}{\boldsymbol{p}_k^T \boldsymbol{L}}$, $\boldsymbol{x}_k = \boldsymbol{x}_{k-1} + \alpha \boldsymbol{p}_k$, $\boldsymbol{r}_k = \boldsymbol{r}_{k-1} - \alpha \boldsymbol{L}$, $\boldsymbol{z}_k = \boldsymbol{C}\boldsymbol{r}_k$
end while
Return $\boldsymbol{x}_k$

Where reshape$(\cdot)$ and vec$(\cdot)$ represent reshaping a matrix and vectorize a matrix, respectively.

3 Interpolation results for 20% observed data

Here we provide the interpolation results for 20% observed data (Table 1), where GP-FA and GP-SBN-FA showed better result than BPFA, while cannot outperform dHBP.

4 BM3D results for denoising task

Here we provide an comparison with BM3D [Danielyan et al.(2012)Danielyan, Katkovnik, and Egiazarian], which represents the state-of-the-art algorithm for denoising tasks (Table 2).

PSNR(dB)	C.man	House	Pepper	Lena	Barbara
BP(512)	24.11	30.12	25.92	31.00	24.80
dHBP(512)	24.43	**32.23**	**27.06**	**32.00**	**29.51**
GP-FA(256)	24.36	31.02	26.13	31.41	27.68
GP-SBN-FA(256)	**24.49**	30.93	26.62	31.39	27.49
PSNR(dB)	Boats	F.print	Man	Couple	Hill
BP(512)	27.81	26.03	28.24	27.72	29.33
dHBP(512)	**28.66**	26.80	**28.86**	**28.55**	**29.94**
GP-FA(256)	27.95	**26.82**	28.64	27.89	29.43
GP-SBN-FA(256)	27.85	26.73	28.48	27.87	29.37

Table 1: Inpainting task (20% data observed). Parenthesis after methods indicates the number of dictionary elements.

$\sigma = 25$					
Method	C.man	House	Pepper	Lena	Barbara
BM3D	29.45	32.86	30.16	32.08	30.72
GP-FA	28.70	32.22	29.65	31.42	29.11
GP-SBN-FA	28.99	32.23	29.78	31.51	29.18
Method	Boats	F.print	Man	Couple	Hill
BM3D	29.91	27.7	29.62	29.72	29.85
GP-FA	29.49	27.55	29.27	29.04	29.49
GP-SBN-FA	29.56	27.54	29.23	29.15	29.52
$\sigma = 50$					
Method	C.man	House	Pepper	Lena	Barbara
BM3D	26.12	29.69	26.68	29.05	27.23
GP-FA	24.66	28.12	25.71	27.80	25.44
GP-SBN-FA	24.66	28.15	25.67	27.83	25.39
Method	Boats	F.print	Man	Couple	Hill
BM3D	26.78	24.53	26.81	26.46	27.19
GP-FA	25.99	23.91	26.22	25.51	26.48
GP-SBN-FA	26.03	23.89	26.18	25.45	26.45

Table 2: Denoising comparison with BM3D for 2 noise levels $\sigma = \{25, 50\}$. Performance is measured as PSNR in dB.

5 Hyper-parameters for dHBP

For all experiments, we set the hyper-parameters as default setting in dHBP. The kernel width is set as 5, radius of spatial neighbor for consideration is set as 3. The hyper-parameters for Gamma prior is 10^{-6}. $c_0 = 10, \eta_0 = 0.5, c_1 = 1, \eta = 10^{-3}$.

6 Supplements for depth restoration

6.1 Different patch extraction strategies

We compared the PSNR results of two approaches. One approach is to extract patches of $8*8*4$ size, where each patch consists pixels of all 4 channels. Our second approach is to extract patches of $8*8$ size, where each patch only contains pixels of single channel. Since the patches of first approach only have location covariate, we put a 2-D Gaussian process prior. For the second approach, we put a 3-D Gaussian prior. We found that the second approach significantly improved the PSNR for recovering both corrupted RGB image and depth information ($\sim$ 1dB improvement). Presumably, the first approach treat each dictionary element as a "bundle" of 4 channel-specific dictionary elements (i.e. the size of dictionary element is $8*8*4$, can be seen as 4 $8*8$ channel specific dictionary elements), which impose strong assumption to the model (channel specific dictionary elements must occur at each location). While the second approach lend more flexibility to the dictionary element, to let each channel freely choose channel specific dictionary elements to recruit. The dependency of the dictionary activation is imposed by the GP across channels.

6.2 Binary activation pattern in depth restoration task

We adopt the second approach to jointly learn the dictionary elements from channel-specific patches. As shown in FIGURE 1, the binary dictionary score shows high level of similarity among RGB channels. The dictionary activation binaries in depth channel have relatively weaker correlation to RGB, compared with the correlation within RGB channels. This motivate us to set θ_d greater than one.

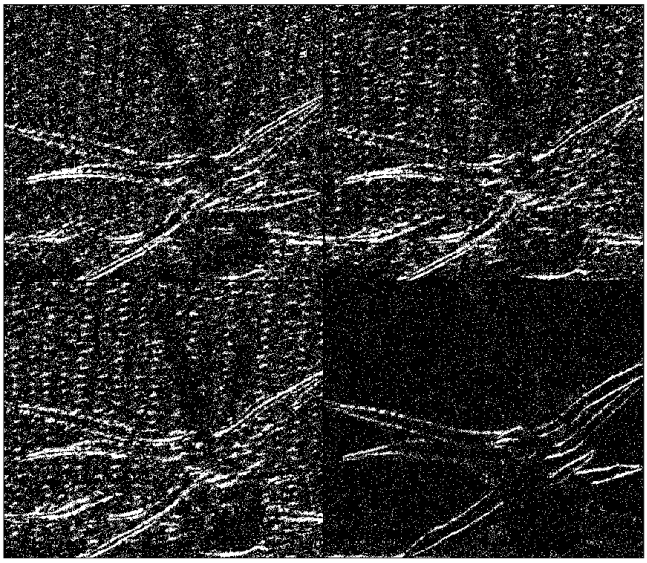

Figure 1: From top-left to bottom-right: binary activation pattern of R,G,B,D channels

6.3 Depth restoration result for 30 images

We compared dHBP, BPFA and our method with 30 RGB images with depth information, the results for each image is as below. Figure 2 shows the RGB channels recovery PSNR and Figure 3 shows the depth channel restoration. In general, GP-FA performs best over these 30 datasets.

As a more detailed comparison with several other methods, we include the results from [Lu et al.(2014)Lu, Ren, and Liu]. The figures 4 are from their original paper.

6.4 Adding the channel covariance kernel leads to performance improvement

We conduct two experiments to demonstrate the effectiveness of RGB channel information, and the channel covariance matrix in improving PSNR results. All methods used 64 dictionary elements and was run for 500 MC iterations. As shown in Figure 5 and Figure 6, using depth information alone (GP-FA depth alone/ GP-SBN-FA depth alone) for interpolation led to much lower performance than taking advantage of corresponding noisy RGB image. Moreover, if the covariance kernel function $\boldsymbol{K}^c$ is removed from the $\boldsymbol{K}$, i.e. only spatial correlation is considered, the results is approximately 0.5 less than those capturing channel. This validate our belief that the across channel information borrowing would potentially leads to better characterization of dictionary elements.

6.5 Inter-dictionary correlation

Here we show the inter-dictionary correlation of a gray-scale image (Barbara). As shown in Figure.7, we found that the hidden nodes in SBN controls a "group" of dictionary elements, to simultaneously turn on (with positive V_{mm_1}) or turn off (with negative V_{mm_1}). The activated (or deactivated) dictionary elements are co-captured by one hidden node, thus display similarity between each other.

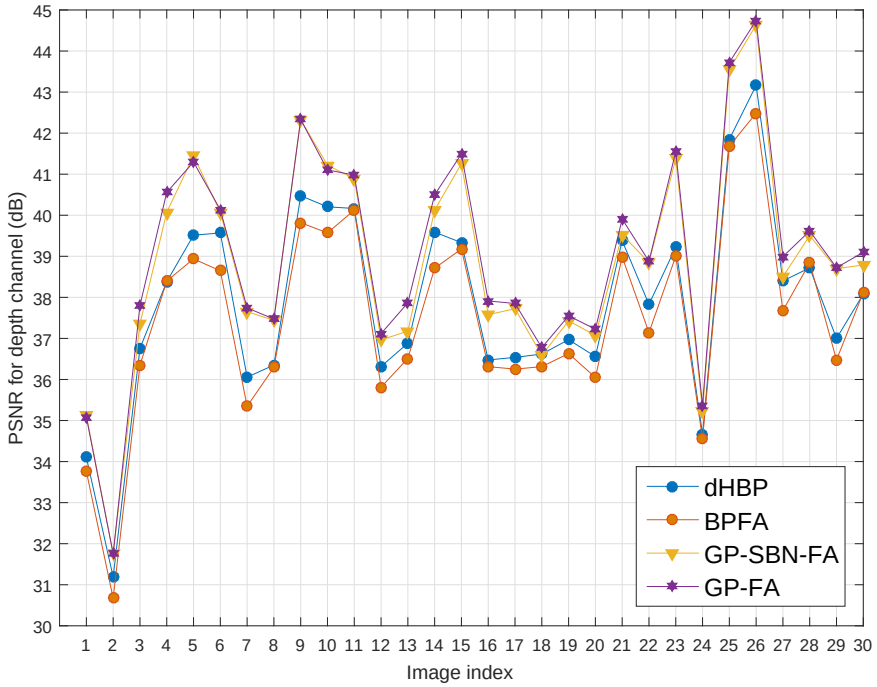

Figure 2: Depth channel recovery comparison. Each line represents a method. The result is evaluated by PSNR (dB)

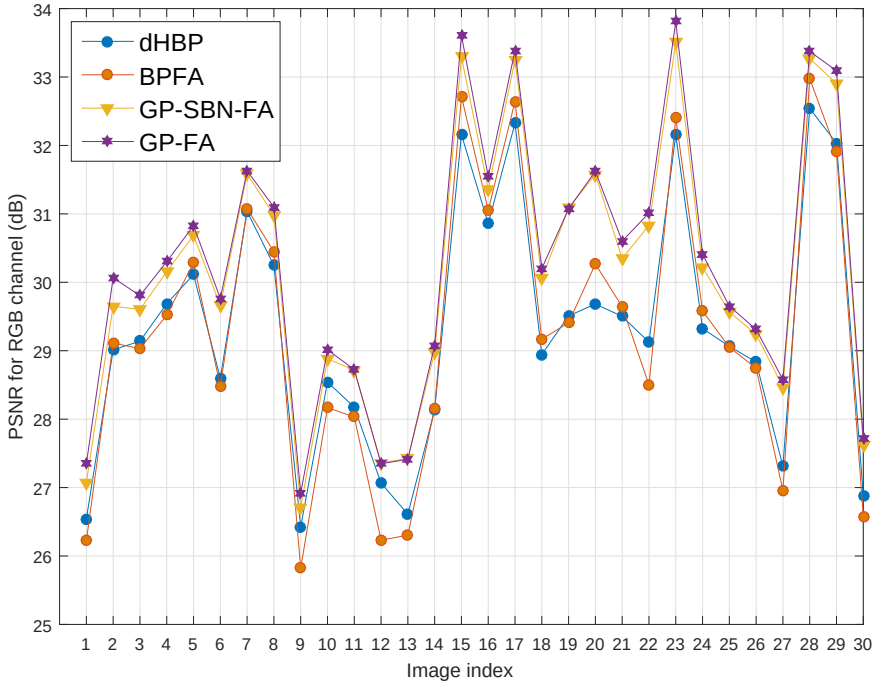

Figure 3: RGB channels recovery comparison. Each line represents a method. The result is evaluated by PSNR (dB)

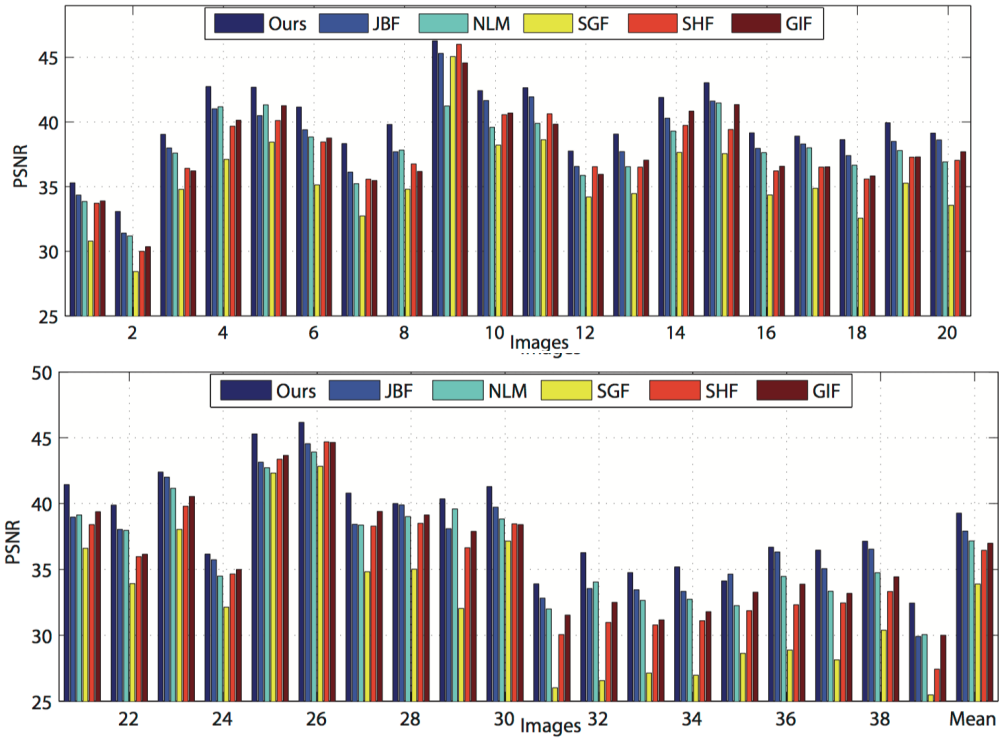

Figure 4: Results for depth restoration images. Methods from left to right: Lu. et al. Joint bilateral filter (JBF), Nonlocal means filter (NLM), Structure-guided fusion (SGF), Spatio- temporal hole filling (SHF), Guided inpainting and filtering (GIF)

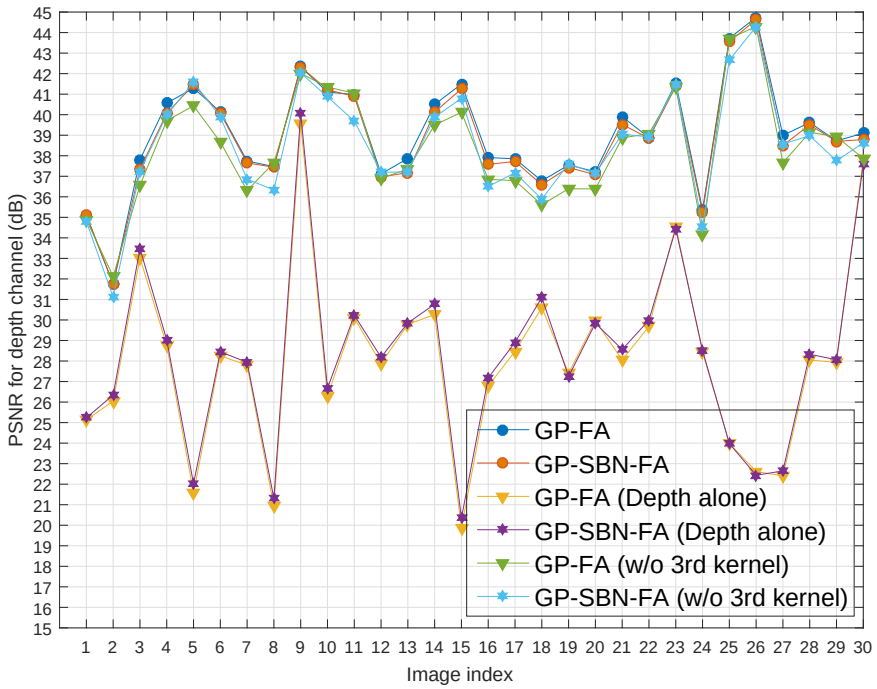

Figure 5: Depth channel recovery comparison. Each line represents a method. The result is evaluated by PSNR (dB)

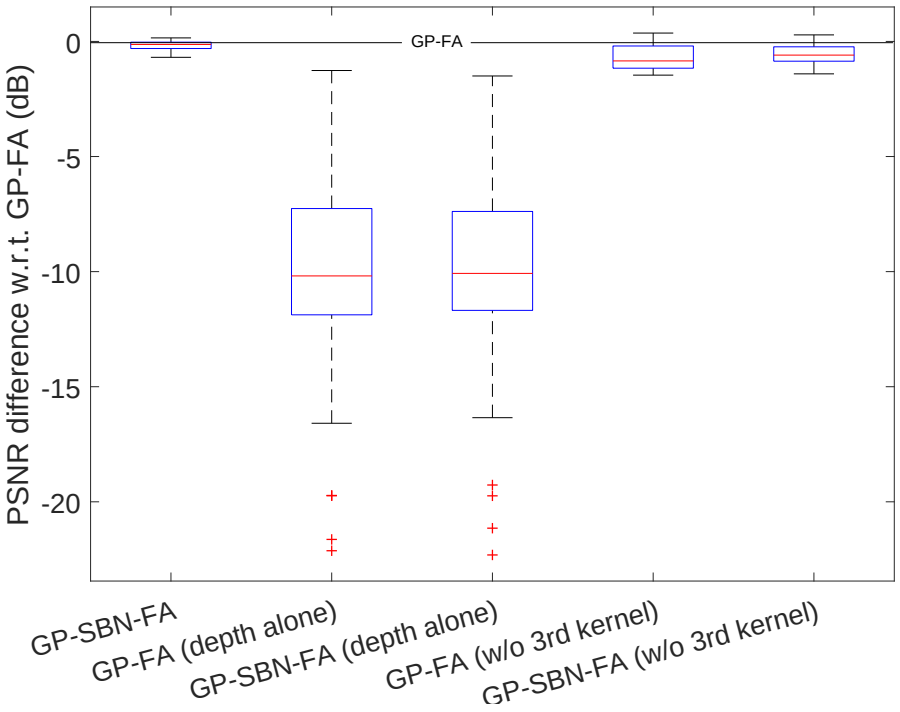

Figure 6: Boxplot shows the difference of PSNR w.r.t. GP-FA

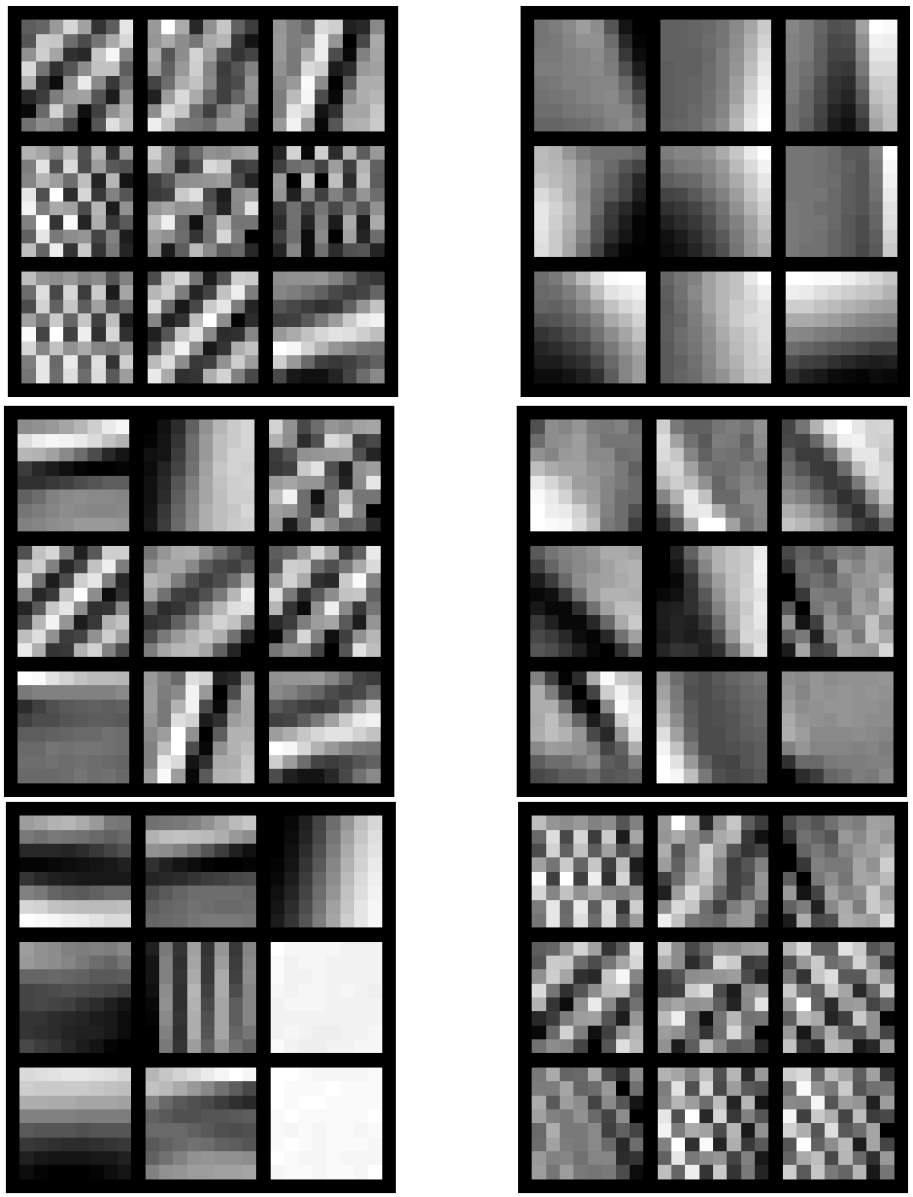

Figure 7: Inter-dictionary dependency captured by three different hidden nodes. Left $3 * 3$ grid shows the dictionary elements with positive weight, i.e. activated by this hidden node. Right $3 * 3$ grid shows the dictionary elements with negative weight, i.e. deactivated by this hidden node.

References

[Danielyan et al.(2012)Danielyan, Katkovnik, and Egiazarian] Aram Danielyan, Vladimir Katkovnik, and Karen Egiazarian. Bm3d frames and variational image deblurring. *Image Processing, IEEE Transactions on*, 21(4): 1715–1728, 2012.

[Lu et al.(2014)Lu, Ren, and Liu] Si Lu, Xiaofeng Ren, and Feng Liu. Depth Enhancement via Low-Rank Matrix Completion. pages 3390–3397, 2014.