Multilook compressive sensing in the presence of speckle noise

Xi Chen
Department of ECE
Rutgers University
xi.chen15@rutgers.edu

Zhewen Hou
Department of Statistics
Columbia University
zh2475@columbia.edu

Christopher A. Metzler
Department of Computer Science
University of Maryland, College Park
metzler@umd.edu

Arian Maleki
Department of Statistics
Columbia University
arian@stat.columbia.edu

Shirin Jalali
Department of ECE
Rutgers University
shirin.jalali@rutgers.edu

Abstract

Multiplicative speckle noise is an inherent part of coherent imaging systems, such as synthetic aperture radar and digital holography. Speckle noise is mitigated by obtaining multiple measurement vectors with independent speckle noise, a technique commonly referred to as "multi-look", followed by appropriate averaging. However, in many applications, even with multi-look, the achievable performance is not satisfactory. Moreover, in this approach, every look (or every set of measurements) is required to be over-determined, which imposes additional costs on the measurement process. In this work, we develop a maximum likelihood based approach for recovering images from a set of under-determined compressive measurements contaminated by speckle noise. We propose an iterative multi-look compressive sensing recovery algorithm, DIP-M^3, that i) requires no training data, ii) is computationally efficient, and iii) generates high-quality reconstruction images from multi-look, where each look is under-determined and corrupted by speckle noise.

1 Problem statement

Speckle noise, or multiplicative noise, is one of the key issues preventing many coherent imaging systems, such as synthetic aperture radar [1] and digital holography [2], from achieving their full potential. The reason is that the speckle noise corruption considerably degrades the acquired images and prevents such systems from producing high-quality images [3, 4].

Let $\mathbf{x}_o \in \mathbb{R}^n$ denote the desired signal. In an imaging system that is affected by speckle noise, the measurements can be represented by $\mathbf{y} = AX_o\mathbf{w}+\mathbf{z}$, where $\mathbf{y} \in \mathbb{R}^m$ is the measurement, $A \in \mathbb{R}^{m\times n}$ denotes the measurement matrix defined by the imaging process, $X_o = \text{diag}(\mathbf{x}_o) \in \mathbb{R}^{n\times n}$ is the diagonal matrix with diagonal entries $\mathbf{x}_o$. Finally, $\mathbf{w} \in \mathbb{R}^n$ and $\mathbf{z} \in \mathbb{R}^m$ denote the multiplicative and additive noise, respectively. Throughout this paper we assume that the entries of $\mathbf{w}$ and $\mathbf{z}$ are i.i.d. $\mathcal{N}(0, \sigma_w^2)$ and $\mathcal{N}(0, \sigma_z^2)$. In the classic setting, $m \geq n$, therefore matrix A is invertible. In such settings, the problem simplifies to a denoising problem with both additive and multiplicative noise.

NeurIPS 2023 Workshop on Deep Learning and Inverse Problems.

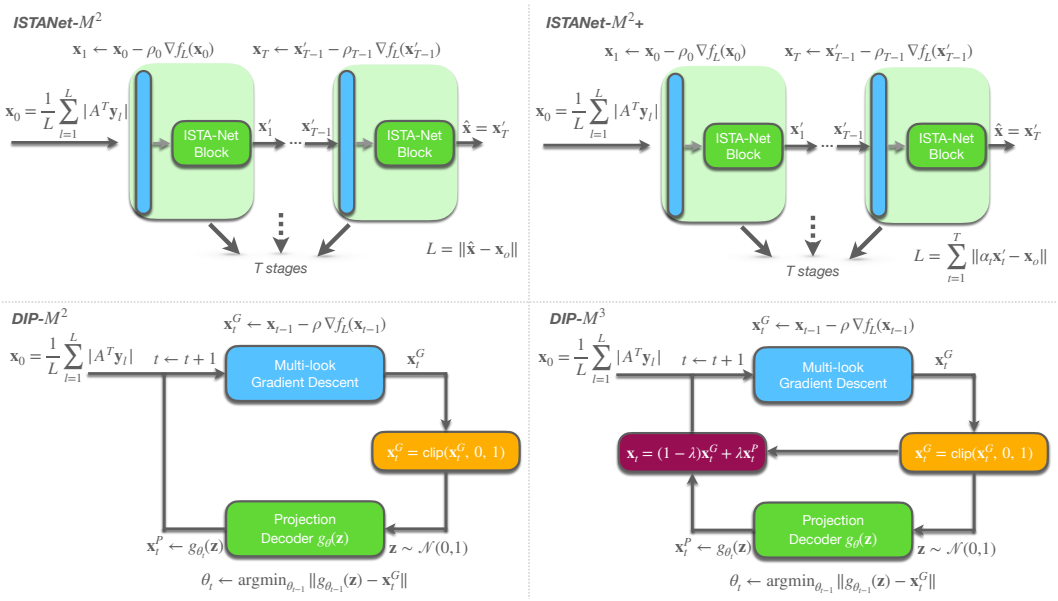


Figure 1: An overview of four model structures we explore in this paper.

In coherent imaging systems, to mitigate the effect of speckle noise, typically multiple measurement vectors with independent speckle noise vectors (referred to multi-looks) are acquired [5–9]. Then, a proper averaging of the measurements is performed [10]. That is, for $l = 1, \dots, L$, where L denotes the number of looks, one acquires $\mathbf{y}_l = A_l X_o \mathbf{w}_l + \mathbf{z}_l$, with $A_l \in \mathbb{R}^{m_l \times n}$ and $m_l \geq n$. Then, inverting the measurement matrices A_l, we derive $\tilde{\mathbf{y}}_l = X_o \mathbf{w}_l + \tilde{\mathbf{z}}_l$, where $\tilde{\mathbf{y}}_l = (A_l^T A_l)^{-1} A_l^T \mathbf{y}_l$ and $\tilde{\mathbf{z}}_i = (A_l^T A_l)^{-1} A_l^T \mathbf{z}_l$. We can further combine the multi-look measurements and derive $\hat{\mathbf{x}}_L = (\frac{1}{L} \sum_{l=1}^{L} \tilde{\mathbf{y}}_l^2)^{\frac{1}{2}}$. It is straightforward to see that as L grows, if there is no additive noise ($\mathbf{z}_l = \mathbf{0}$), then $\hat{\mathbf{x}}_L$ converges to $\mathbf{x}_o$, almost surely.

A key challenge in the described approach is that in many applications A is ill-conditioned, and inverting A leads to amplifying additive noise and adding dependencies. Hence, in such scenarios the current approaches offer sub-optimal performance. Furthermore, in a recent work [11], using maximum likelihood estimation (MLE), it was theoretically shown that it is possible to accurately estimate structured signal $\mathbf{x}_o$ even from *under-determined* measurements $\mathbf{y} = AX_o\mathbf{w}$ with $m < n$. This raises the following question: Given multi-look in which A_l matrix is either under-determined or ill-posed, i.e., $\mathbf{y}_l = A_l X_o \mathbf{w}_l + \mathbf{z}_l$, where $\mathbf{y}_l \in \mathbb{R}^m$ with $m < n$, can we employ MLE to recover $\mathbf{x}_o$ from $(\mathbf{y}_l)_{l=1}^L$ without inverting A and translating the problem to denoising?

In this paper, we address this problem and provide MLE-based algorithms for recovering $\mathbf{x}_o$ from under-determined multi-look measurements. To simplify the presentation of the results, we make the following assumptions: i) We ignore the additive noise and assume that the achievable performance is dominated by speckle noise. This is a reasonable assumption in some coherent imaging applications, unless we amplify the additive noise through inverting matrix A. ii) We assume that the measurement matrix is constant across different looks, that is, $A_l = A$, for $l = 1, \dots, L$. This assumption is valid in some applications. We will relax both assumptions in our future work.

2 Method

2.1 Our proposals

To recover $\mathbf{x}_o$ from under-determined measurements $\mathbf{y} = AX_o\mathbf{w}$, we need to take the structure of $\mathbf{x}_o$ into account. In [11], the authors use compression codes to capture the signal's structure. Instead, in this paper, we use the idea of Deep Image Prior (DIP) [12] to represent the source structure. DIP is described by function $g_\theta : \mathbb{R}^k \to \mathbb{R}^n$, which is represented by neural networks. For $\mathbf{x} \in \mathcal{Q}$, where $\mathbf{x} \in \mathcal{Q}$ denotes a subset of $\mathbb{R}^n$ that describes the class of structured signals we are interested in, e.g., the class of natural images, one generates $\mathbf{u} \sim \mathcal{N}(\mathbf{0}, I_k)$ and finds $\hat{\theta}(\mathbf{x})$ (e.g. weights of neural networks) that minimizes $\|g_\theta(\mathbf{u}) - \mathbf{x}\|_2^2$. Intuitively, g_θ is a good fit for the desired class of signals $\mathcal{Q}$ if, for almost every $\mathbf{x} \in \mathcal{Q}$, $\|g_{\hat{\theta}(\mathbf{x})}(\mathbf{u}) - \mathbf{x}\|_2^2$ is small. Existence of such networks for the class of

natural images is empirically shown in [13, 14]. A key advantage of modeling signal structure using DIP, as we will elaborate later, is its convenient integration with our proposed algorithms.

Using MLE to recover $\mathbf{x}_o$ from $\mathbf{y}$ and using DIP instead of compression codes to capture the signal model, the MLE optimization with constraint studied in [11] can be written as

$$\hat{\mathbf{x}} = \underset{X=\text{diag}(\mathbf{x}):\ \mathbf{x}=g_\theta(\mathbf{u})}{\text{argmin}} \big[\log\det(AX^2A^T) + \frac{1}{\sigma_w^2}\mathbf{y}^T(AX^2A^T)^{-1}\mathbf{y}\big]. \tag{1}$$

Note that (1) is for the single-look problem. In the multi-look scenario, $\mathbf{y}_l = AX_o\mathbf{w}_l$, given the independence of the noise vectors, it is straightforward to show that, (1) generalizes as follows

$$\hat{\mathbf{x}}_L = \underset{X=\text{diag}(\mathbf{x}):\ \mathbf{x}=g_\theta(\mathbf{u})}{\text{argmin}} \big[\log\det(AX^2A^T) + \frac{1}{L\sigma_w^2}\sum_{l=1}^{L}\mathbf{y}_l^T(AX^2A^T)^{-1}\mathbf{y}_l\big]. \tag{2}$$

The optimization in (2) involves cost function $f_L : \mathbb{R}^n \to \mathbb{R}$, $f_L(\mathbf{x}) = \log\det(AX^2A^T) + \frac{1}{L\sigma_w^2}\sum_{l=1}^{L}\mathbf{y}_l^T(AX^2A^T)^{-1}\mathbf{y}_l$, $X = \text{diag}(\mathbf{x})$, and constraint $\mathbf{x} = g_\theta(\mathbf{u})$ that are both non-convex and complex. Therefore, solving it is challenging, and one needs efficient methods to approximate the solution of (2). To solve such complex optimization problems, one can potentially consider: 1) End-to-end networks trained to recover $\mathbf{x}_o$ directly from sets of measurements $(\mathbf{y}_l)_{l=1}^L$. For instance, we consider the DnCNN structure [15] and train it to recover $\mathbf{x}_o$ from a proper initialization that combines the measurements. This approach ignores the cost function and constraint set in (2). 2) Unrolled networks consider the gradient of the cost function in (2) and are trained end-to-end to minimize the training loss between the output of the unrolled network and $\mathbf{x}_o$. 3) Iterative approaches based on projected gradient descent (PGD): we use both the cost function and constraint set in (2).

Inspired by these generic approaches, in this paper, we study the following four different algorithms, schematically shown in Figure 1, for solving the Multi-look system with Multiplicative noise (M^2) problem. We will use end-to-end solution for baseline comparison later in our experiment.

1. ISTANet-M^2: An unrolled network [16] that employs ISTA [17] structure, and was previously proposed for compressed sensing recovery [18], where we use the gradient of f_L and the same neural networks structure as in [18].
2. ISTANet-M^2+: ISTANet-M^2+ employs the same neural network structure as ISTANet-M^2. However, we also consider the intermediate losses and add learnable parameters α_t, t denoting the stage index, as explained in Figure 1. The details are presented in Appendix 4.2.
3. DIP-M^2: An iterative algorithm based on PGD [19], where it employs the gradient of cost function f_L and utilization of deep prior in [20] for projection. The implicit prior of the image is introduced by a Deep Decoder [21] $g_\theta(\cdot)$. The parameters θ are optimized by minimizing the mean-squared-error (MSE) between the gradient descent (GD) step results and $g_\theta(\mathbf{z})$, where $\mathbf{z} \sim \mathcal{N}(0, 1)$.
4. DIP-M^3: Inspired by DIP-M^2, we propose DIP with Memory for Multi-look system with Multiplicative noise (DIP-M^3), which involves effective combination of the memory of GD and projection output. As reported in Section 3, this simple modification improves the performance of DIP-M^2 by around $3dB$ on average. The details are presented in Section 2.4.

We elaborate DIP-M^2 based method in Section 2.3. Before that, note that all methods employ the gradient of f_L. So we first derive the gradient and comment on its asymptotic behaviour as $L \to \infty$.

2.2 Gradient of Multi-look Cost Function

The methods described above require access to the gradient of $f_L(\mathbf{x})$. As we show in Appendix 4.1,

$$\frac{\partial f_L}{\partial \mathbf{x}_j} = 2\mathbf{x}_j\Big(\mathbf{a}_j^T(AX^2A^T)^{-1}\mathbf{a}_j - \frac{1}{L\sigma_w^2}\sum_{l=1}^{L}\big(\mathbf{a}_j^T(AX^2A^T)^{-1}\mathbf{y}_l\big)^2\Big). \tag{3}$$

where $\mathbf{a}_j \in \mathbb{R}^m$ denotes column j of matrix A. We show that in Appendix 4.1, as L grows, $\nabla f_L(\mathbf{x}_o)$ converges to zero. That is, as we get more measurements, $\mathbf{x}_o$ becomes a local minima of f_L.

		DnCNN	ISTANet-M^2	ISTANet-M^2+	DIP-M^2	DIP-M^3
ORMM	Sampling rate 25%	18.18	22.06	**23.12**	19.29	22.81
	Sampling rate 50%	21.83	**26.33**	24.23	23.11	25.91
BRMM	Blur kernel	19.94	-	-	25.31	**28.26**

Table 1: Average PSNR (dB) comparison of models in ORMM and BRMM tasks with $L = 50$.

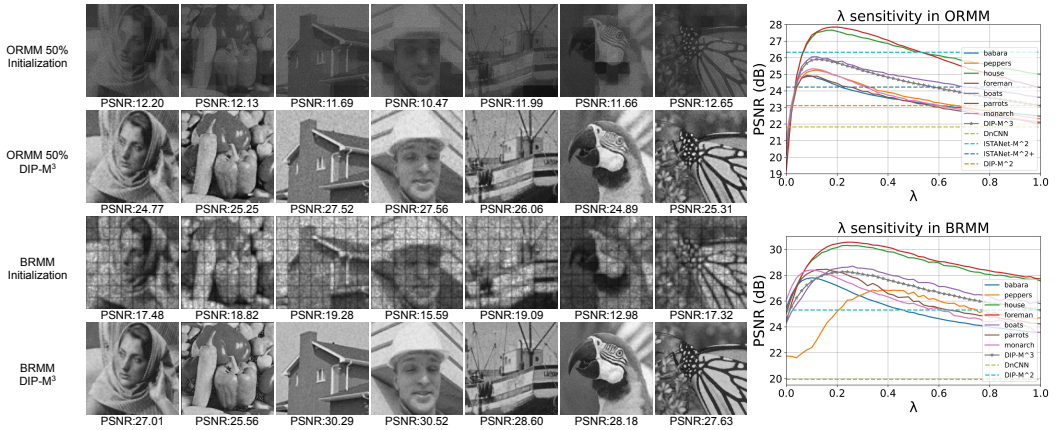


Figure 2: Left: We show the initialization (row 1,3) and reconstruction (row 2,4) for ORMM and BRMM with $L = 50$. Right: We show the sensitivity of each image reconstructed by DIP-M^3 to λ, and compare the average PSNR of all models on ORMM (upper) and BRMM tasks (bottom).

2.3 DIP-M^2 based Method

The proposed DIP-M^2 based method requires T iterations until the optimization converges. Each iteration has a GD step and projection step. We denote the GD step result, obtained by using (3), as $\mathbf{x}_t^G$. The projection step enforces an implicit prior on $\mathbf{x}_t^G$. The implicit prior is introduced by $g_\theta(\mathbf{z})$ with fixed Gaussian noise input $\mathbf{z}$ throughout all iterations. The $g_\theta(\cdot)$ structure is described in Appendix 4.2 and Figure 3. The optimization over θ, i.e. minimizing $\|g_\theta(\mathbf{z}) - \mathbf{x}_t^G\|$, requires a number of T' nested iterations until convergence. We denote the output of trained $g_{\theta_t}(\cdot)$ as $\mathbf{x}_t^P = g_{\theta_{T'}}(\mathbf{z})$.

2.4 "Residual" Connection between $\mathbf{x}_t^G$ and $\mathbf{x}_t^P$

In DIP-M^3, we propose a simple yet effective "Residual" structure to better enhance the fusion of $\mathbf{x}_t^G$ and $\mathbf{x}_t^P$. Considering the gradient (3) calculated with multi-look system is more stable, instead of using $\mathbf{x}_t^P$ as next iteration input $\mathbf{x}_t$, we introduce a hyperparameter λ, which balance the contribution from $\mathbf{x}_t^G$ and $\mathbf{x}_t^P$ as shown in Figure 1. The hyperparameter $\lambda \in [0, 1]$ is used for generating $\mathbf{x}_t$ as:

$$\mathbf{x}_t = (1 - \lambda)\mathbf{x}_t^G + \lambda \mathbf{x}_t^P.$$

The images $(\mathbf{x}_t)_{t=1}^T$ are generated throughout T iterations, and final reconstructed image is $\hat{\mathbf{x}} = \mathbf{x}_T$. We describe our method details in Algorithm 1. To keep our algorithm simple, we assume the hyperparameter λ is time invariant. However, one can consider time-dependent choices for λ.

3 Experiments

We compare the four approaches with representative end-to-end solution DnCNN [15] on Recovering signal from Multi-look systems in the presence of Multiplicative noise (RMM) tasks, with degradation model A to be row-subsampled random orthogonal (ORMM) or Gaussian matrix (GRMM), and blur kernel matrix (BRMM). (The details of the experiment setup are in Appendix 4.2.) **i) ORMM:** We set $A \in \mathbb{R}^{m \times n}$ such that $AA^T = I$. (In Appendix 4.3, we present our results from GRMM and compare them with ORMM results.) We consider 25% and 50% sampling rates, and initialization $\mathbf{x}_0 = \frac{1}{L}\sum_{l=1}^{L} |A^T \mathbf{y}_l|$. The quantitative and qualitative results are in Table 1 and Figure 2. (More results are in Appendix Table 2 and Figure 5, 6.) We find ISTANet-M^2 based methods slightly outperform DIP-M^3, but training the former one is much more expensive as each training image involves matrix inversion computation. **ii) BRMM:** In image deblurring, we assume the blur kernel is separable, i.e., $A = A_r \otimes A_c \in \mathbb{R}^{n \times n}$ is the Kronecker product of 1D convolutional matrices. (Details of the blurring kernel can be found in Appendix 4.2.) The results are shown in Table 1 and Figure 2. (More results can be found in Appendix 4.3, Table 2 and Figure 7.) In this case, the unrolled

models fail to converge. Note that, they require gradient calculation (3) during training, involving computation of $(AX^2A^T)^{-1} \in \mathbb{R}^{m\times m}$ which on one hand is computationally very demanding since $m = n$, and on the other hand introduces instability issues to the algorithms because of AX^2A^T being ill-conditioned. We also study the hyperparameters used in DIP-M^3. **i) Choice of memory control strength λ:** We find the sensitivity of reconstruction performance to the hyperparameter λ in DIP-M^3 is similar across all test images in Figure 2. Note that when $\lambda = 1$, DIP-M^3 simplifies to DIP-M^2, when $\lambda = 0$, no prior is enforced by the decoder network. **ii) Convergence iteration T:** We plot the PSNR and visualize the intermediate reconstructed images in Figures 10 and 11, respectively, presented in Appendix 4.3. The convergence with different L and λ values are in Figures 9 (b)-(f). It can be observed that the algorithm typically converges within 50 iterations. **iii) Number of looks L:** We show the reconstruction performance with different L in Figure 8. We study the effect of L on the choice of λ, and show the 50% CS reconstruction with different L in Figure 9 (a). Specifically, we see that the reconstruction with larger L benefits more from smaller λ.

We find that training DnCNN is more efficient than unrolled networks, but the PSNR is downgraded given that gradient in (3) is not explicitly considered. ISTANet-M^2 based methods achieve good performance but the training is very computationally expensive since for each training image it requires inversion of large matrices. The proposed DIP-M^3, however, requires no training data, and provides an efficient and effective solution.

Acknowledgements

X.C., S.J., Z.H. and A.M. were supported in part by ONR award no. N00014-23-1-2371. C.A.M. was supported in part by SAAB, Inc., AFOSR Young Investigator Program Award no. FA9550-22-1-0208, and ONR award no. N000142312752.

References

[1] Moreira, A., P. Prats-Iraola, M. Younis, et al. A tutorial on synthetic aperture radar. *IEEE Geoscience and remote sensing magazine*, 1(1):6–43, 2013.

[2] Schnars, U., C. Falldorf, J. Watson, et al. *Digital holography*. Springer, 2015.

[3] Goodman, J. W. *Speckle phenomena in optics: theory and applications*. Roberts and Company Publishers, 2007.

[4] Argenti, F., A. Lapini, T. Bianchi, et al. A tutorial on speckle reduction in synthetic aperture radar images. *IEEE Geoscience and remote sensing magazine*, 1(3):6–35, 2013.

[5] Baumbach, T., E. Kolenovic, V. Kebbel, et al. Improvement of accuracy in digital holography by use of multiple holograms. *Applied Optics*, 45(24):6077–6085, 2006.

[6] Quan, C., X. Kang, C. J. Tay. Speckle noise reduction in digital holography by multiple holograms. *Optical Engineering*, 46(11):115801–115801, 2007.

[7] Kuratomi, Y., K. Sekiya, H. Satoh, et al. Speckle reduction mechanism in laser rear projection displays using a small moving diffuser. *JOSA A*, 27(8):1812–1817, 2010.

[8] Memmolo, P., V. Bianco, M. Paturzo, et al. Encoding multiple holograms for speckle-noise reduction in optical display. *Optics Express*, 22(21):25768–25775, 2014.

[9] Bianco, V., P. Memmolo, M. Paturzo, et al. Quasi noise-free digital holography. *Light: Science & Applications*, 5(9):e16142–e16142, 2016.

[10] Bianco, V., P. Memmolo, M. Leo, et al. Strategies for reducing speckle noise in digital holography. *Light: Science & Applications*, 7(1):48, 2018.

[11] Zhou, W., S. Jalali, A. Maleki. Compressed sensing in the presence of speckle noise. *IEEE Transactions on Information Theory*, 68(10):6964–6980, 2022.

[12] Ulyanov, D., A. Vedaldi, V. Lempitsky. Deep image prior. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9446–9454. 2018.

[13] Van Veen, D., A. Jalal, M. Soltanolkotabi, et al. Compressed sensing with deep image prior and learned regularization. *arXiv preprint arXiv:1806.06438*, 2018.

[14] Mataev, G., P. Milanfar, M. Elad. Deepred: Deep image prior powered by red. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0. 2019.

[15] Zhang, K., W. Zuo, Y. Chen, et al. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE transactions on image processing*, 26(7):3142–3155, 2017.

[16] Monga, V., Y. Li, Y. C. Eldar. Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing. *IEEE Signal Processing Magazine*, 38(2):18–44, 2021.

[17] Beck, A., M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

[18] Zhang, J., B. Ghanem. Ista-net: Interpretable optimization-inspired deep network for image compressive sensing. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1828–1837. 2018.

[19] Boyd, S., N. Parikh, E. Chu, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.

[20] Jagatap, G., C. Hegde. Algorithmic guarantees for inverse imaging with untrained network priors. *Advances in neural information processing systems*, 32, 2019.

[21] Heckel, R., P. Hand. Deep decoder: Concise image representations from untrained non-convolutional networks. *arXiv preprint arXiv:1810.03982*, 2018.

[22] Kulkarni, K., S. Lohit, P. Turaga, et al. Reconnet: Non-iterative reconstruction of images from compressively sensed measurements. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 449–458. 2016.

[23] Kingma, D. P., J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

4 Appendix

4.1 Gradient descent for multi-look systems

We calculate the gradient of objective function (2) in the following way. Define diagonal matrix $\delta_j = \text{diag}([0, \cdots, 1, \cdots, 0])$, with all zeros except the j-th entry to be 1. We denote that the j-th coordinate of X is updated as $X^2 \leftarrow X^2 + \alpha_j \delta_j$, where α_j is the increment. We need to minimize the updated objective function value over α_j:

$$\alpha_j = \underset{\alpha_j}{\text{argmin}} \big[\log\det\big(A(X^2 + \alpha_j\delta_j)A^T\big) + \frac{1}{L\sigma_w^2}\sum_{l=1}^{L} \mathbf{y}_l^T \big(A(X^2 + \alpha_j\delta_j)A^T\big)^{-1}\mathbf{y}_l \big]. \tag{4}$$

this can be further written as:

$$\alpha_j = \underset{\alpha_j}{\text{argmin}} \big[\log\det\big(B + \alpha_j\mathbf{a}_j\mathbf{a}_j^T\big) + \frac{1}{L\sigma_w^2}\sum_{l=1}^{L} \mathbf{y}_l^T \big(B + \alpha_j\mathbf{a}_j\mathbf{a}_j^T\big)^{-1}\mathbf{y}_l \big]. \tag{5}$$

where we denote $B = AX^2A^T$. By matrix inversion Lemma, we know that $\big(B + \alpha_j\mathbf{a}_j\mathbf{a}_j^T\big)^{-1} = B^{-1} - \frac{\alpha_j B^{-1}\mathbf{a}_j\mathbf{a}_j^T B^{-1}}{1+\alpha_j\mathbf{a}_j^T B^{-1}\mathbf{a}_j}$. We also know that from the property of determinant and eigenvalues:

$$\begin{aligned}
\log\det\big(B + \alpha_j\mathbf{a}_j\mathbf{a}_j^T\big) &= \log\det\big(B^{\frac{1}{2}}(I + \alpha_j B^{-\frac{1}{2}}\mathbf{a}_j\mathbf{a}_j^T B^{-\frac{1}{2}})B^{\frac{1}{2}}\big) &(6)\\
&= \log\det(B) + \log\det\big(I + \alpha_j B^{-\frac{1}{2}}\mathbf{a}_j\mathbf{a}_j^T B^{-\frac{1}{2}}\big) &(7)\\
&= \log\det(B) + \log(1 + \alpha_j\mathbf{a}_j^T B^{-1}\mathbf{a}_j). &(8)
\end{aligned}$$

we note that the last equality comes from that, all but one of the eigenvalues of matrix $(I + \alpha_j B^{-\frac{1}{2}}\mathbf{a}_j\mathbf{a}_j^T B^{-\frac{1}{2}})$ are 1. Then the gradient at $\mathbf{x}_j^2$ can be calculated as:

$$\begin{aligned}
\frac{\partial f_L}{\partial \mathbf{x}_j^2} &= \lim_{\alpha_j\to 0}\frac{f_L(\mathbf{x}_j^2 + \alpha_j) - f_L(\mathbf{x}_j^2)}{\alpha_j} &(9)\\
&= \lim_{\alpha_j\to 0}\frac{\log(1 + \alpha_j\mathbf{a}_j^T B^{-1}\mathbf{a}_j) - \frac{1}{L\sigma_w^2}\sum_{l=1}^{L}\frac{\alpha_j\mathbf{y}_l^T B^{-1}\mathbf{a}_j\mathbf{a}_j^T B^{-1}\mathbf{y}_l}{1+\alpha_j\mathbf{a}_j^T B^{-1}\mathbf{a}_j}}{\alpha_j} &(10)\\
&= \mathbf{a}_j^T B^{-1}\mathbf{a}_j - \frac{1}{L\sigma_w^2}\sum_{l=1}^{L}(\mathbf{a}_j^T B^{-1}\mathbf{y}_l)^2. &(11)
\end{aligned}$$

The second equality comes from using matrix inversion Lemma, and property of determinant and eigenvalues as we described before. The last equality comes from using the fact that $\lim_{x\to 0}\frac{\log(1+x)}{x} = 1$. Thus we can express the gradient of (2) at $\mathbf{x}_j$ as:

$$\begin{aligned}
\frac{\partial f_L}{\partial \mathbf{x}_j} &= 2\mathbf{x}_j\Big(\mathbf{a}_j^T B^{-1}\mathbf{a}_j - \frac{1}{L\sigma_w^2}\sum_{l=1}^{L}\big(\mathbf{a}_j^T B^{-1}\mathbf{y}_l\big)^2\Big) &(12)\\
&= 2\mathbf{x}_j\Big(\mathbf{a}_j^T B^{-1}\mathbf{a}_j - \mathbf{a}_j^T B^{-1}AX_o\frac{1}{L\sigma_w^2}\sum_{l=1}^{L}\mathbf{w}_l\mathbf{w}_l^T X_o A^T B^{-1}\mathbf{a}_j\Big) &(13)\\
&= 2\mathbf{x}_j\mathbf{a}_j^T\Big(B^{-1} - B^{-1}AX_o(\frac{1}{\sigma_w^2 L}\sum_{l=1}^{L}\mathbf{w}_l\mathbf{w}_l^T)X_o A^T B^{-1}\Big)\mathbf{a}_j &(14)\\
&= 2\mathbf{x}_j\mathbf{a}_j^T B^{-\frac{1}{2}}\Big(I - B^{-\frac{1}{2}}AX_o(\frac{1}{\sigma_w^2 L}\sum_{l=1}^{L}\mathbf{w}_l\mathbf{w}_l^T)X_o A^T B^{-\frac{1}{2}}\Big)B^{-\frac{1}{2}}\mathbf{a}_j. &(15)
\end{aligned}$$

where the term $(\frac{1}{\sigma_w^2 L}\sum_{l=1}^{L}\mathbf{w}_l\mathbf{w}_l^T)$ is approximately an identity matrix when L is large enough, remember that $AX^2A^T = B$, so the gradient at optimal point $\nabla f(\mathbf{x}_o) = 0$, $\mathbf{x}_o$ becomes a local minima of f_L. This helps explain the better performance achieved by using larger number of looks.

Algorithm 1 DIP with Memory for Multi-look system with Multiplicative noise (DIP-M^3)

Input: $\{\mathbf{y}_l\}_{l=1}^L, A, \mathbf{x}_0 = \frac{1}{L}\sum_{l=1}^L |A^T\mathbf{y}_l|, \rho, \lambda \in [0,1], T, g_{\theta_0}(\cdot)$.
Output: Reconstructed $\hat{\mathbf{x}}$.
for $t = 1, \ \ldots, \ T$ **do**
  [Gradient Descent Step]
  Gradient at coordinate j as $\nabla f(\mathbf{x}_{t-1,j})$ and update $\mathbf{x}_{t,j}^G$: $\mathbf{x}_{t,j}^G \leftarrow \mathbf{x}_{t-1,j} - \rho\nabla f(\mathbf{x}_{t-1,j})$ in (3).
  Truncate $\mathbf{x}_t^G$ into range $(0,1)$, $\mathbf{x}_t^G = \text{clip}(\mathbf{x}_t^G, 0, 1)$.
  [Deep Prior Projection Step]
  Generate random image given fixed randomly generated noise $\mathbf{z} \sim \mathcal{N}(0,1)$ as $g_{\theta_{t-1}}(\mathbf{z})$.
  Update θ_t by optimizing $\|g_{\theta_{t-1}}(\mathbf{z}) - \mathbf{x}_t^G\|$: $\theta_t \leftarrow \text{argmin}_{\theta_{t-1}} \|g_{\theta_{t-1}}(\mathbf{z}) - \mathbf{x}_t^G\|$ till converges.
  Generate $\mathbf{x}_t^P$ using trained deep decoder as $\mathbf{x}_t^P \leftarrow g_{\theta_t}(\mathbf{z})$.
  Obtain $\mathbf{x}_t$ by adding $\mathbf{x}_t^G$ and $\mathbf{x}_t^P$ with coefficient λ: $\mathbf{x}_t = (1-\lambda)\mathbf{x}_t^G + \lambda\mathbf{x}_t^P$.
end for
Reconstruct image as $\hat{\mathbf{x}} = \mathbf{x}_T$.

4.2 Experiment setup

Degradation model setup We partition the image of (256×256) into patches of (32×32). The number of looks is denoted as L, so the patches shape becomes $(64 \times 1 \times 32 \times 32 \times L)$, where 64 is the batch size, 1 is the channel number. The shape of multiplicative noise $\mathbf{w}$ is also $(64 \times 1 \times 32 \times 32 \times L)$, which indicates the noise $\mathbf{w}_i \sim \mathcal{N}(0, \sigma_w^2)$ is generated independently for each pixel across every batch and look. The measurement matrix A is consistent in each batch and look.

Decoder $g_\theta(\cdot)$ structure The input of $g_\theta(\cdot)$ is randomly generated Gaussian noise $\mathbf{z} \sim \mathcal{N}(0,1)$, $\mathbf{z}$ is fixed in each *Deep Prior Projection Step* of Algorithm 1. Each layer of the deep decoder consists of 1) a pixel-wise linear combinations (1×1 conv layer); 2) activation function ReLU; 3) Batch Normalization (BN); 4) Bi-linear upsampling with scale factor $= 2$. The final output layer consists of 1) a pixel-wise linear combinations (1×1 conv layer); 2) Sigmoid function. The $g_\theta(\cdot)$ we use has 4 layers and 1 output layer. The structure of the decoder is shown in Figure 3. The input and output channel numbers for all layers are $[100, 50, 25, 10]$ and $[50, 25, 10, 1]$ respectively. Since the test image size is $(1 \times 256 \times 256)$ with single channel, we partition it into small patches as (32×32) without overlaps, thus the image size becomes $(64 \times 1 \times 32 \times 32)$ with batch size 64 and channel number 1. Given the decoder structure and image size, the size of the input noise $\mathbf{z}$ is $(64 \times 100 \times 4 \times 4)$. The total number of parameters in $g_\theta(\cdot)$ is 13,990.

ORMM The random orthogonal matrix $A \in \mathbb{R}^{m \times n}$ with $m < n$ satisfies $AA^T = I$, we also compare the performance of our method on ORMM and GRMM in Appendix 4.3. Given the computational resource limitation, we crop the image into smaller patches as input of models. Since the ISTANet-M^2-based and DIP-M^2-based methods demand the matrix inverse computation of $AX^2A^T \in \mathbb{R}^{m \times m}$, which is computationally intensive, we crop image into patches of size (32×32).

BRMM In the image deblurring mode, the blur mechanism is assumed to be linear and is modeled as a convolution with a 2D kernel $k(x,y)$. We further assume that the kernel is seperable and is written as $k(x,y) = r(x)c(y)$. we consider the blurring kernel k denote 1D convolutional matrices $A_r, A_c \in \mathbb{R}^{n' \times n'}$ with kernel r and c respectively. The blurring process on raw image $x_o \in \mathbb{R}^{n' \times n'}$ with multiplicative noise is $Y = A_r(x_o \odot w)A_c^T$, where $Y \in \mathbb{R}^{n' \times n'}$ and $w \in \mathbb{R}^{n' \times n'}$ are the blurred noisy image and speckle noise respectively. We denote $n = n' \times n'$, the vectorized $\mathbf{y} \in \mathbb{R}^n$ can be alternatively represented as $\mathbf{y} = AX_o\mathbf{w}$, where $A = A_r \otimes A_c \in \mathbb{R}^{n \times n}$ is the Kronecker product of the two 1D convolutional matrix, $X_o = \text{diag}(\mathbf{x}_o) \in \mathbb{R}^{n \times n}$ is the diagonal matrix where diagonal entry $\mathbf{x}_o \in \mathbb{R}^n$ is the vectorized $x_o \in \mathbb{R}^{n' \times n'}$, $\mathbf{w} \in \mathbb{R}^n$ is the vectorized $w \in \mathbb{R}^{n' \times n'}$. As we mentioned before, we consider separable blurring kernel as $k = rc^T$, we next introduce how we construct r and c. We define Gaussian PDF as $f(x) = e^{-x^2/2\sigma^2}$, where $\sigma = 10$, then $r = c = [f(-10), f(-8), f(-6), f(-4), f(-2), f(0), f(2), f(4), f(6), f(8), f(10)]^T$, and the 1D convolutional matrices A_r, A_c with kernel r, c are made accordingly.

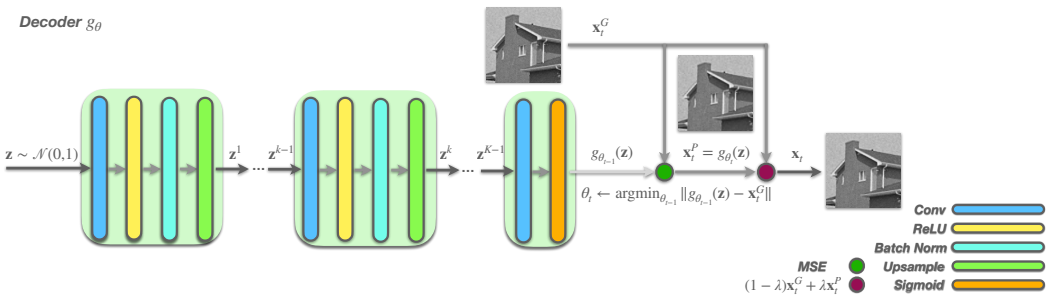


Figure 3: We visualize the detailed structure of decoder g_θ, and show how we use it in DIP-M^3.

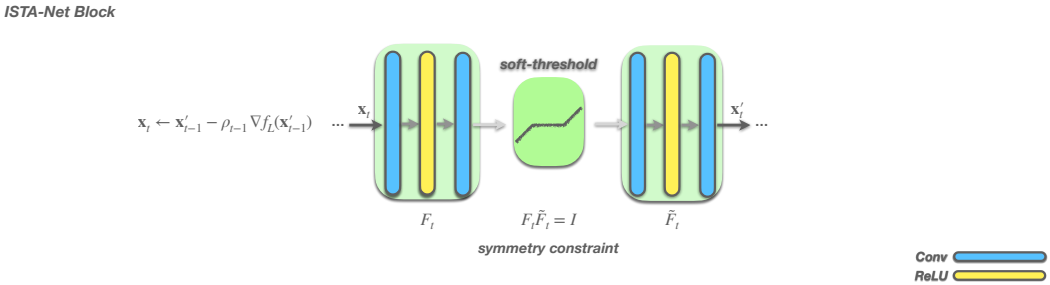


Figure 4: We visualize the Figure 1 ISTA-Net Block structure of ISTANet-M^2 and ISTANet-M^2+. F denotes the forward networks, and $\tilde{F}$ is designed to be symmetric to the structure of F.

Baseline models setup We use supervised learning strategy for DnCNN and ISTANet-M^2-based methods, the training data are 91 images that are commonly used in previous works [18, 22]. The degradation process and initialization are the same for every model. Given that DnCNN does not involve matrix inverse $(AX^2A^T)^{-1}$ calculation, the patch size and stride step when cropping the training images are $(128 \times 128), 64$. We choose $(32 \times 32), 14$ for ISTANet-M^2. The number of blocks (stages) used in DnCNN and ISTANet-M^2 is 9. The block (stage) structures of DnCNN and ISTANet-M^2-based method are kept the same as stated in [15, 18]. We present the block structure of ISTANet-M^2-based method in Figure 4. We also modify the ISTANet-M^2 and name the modified version as ISTANet-M^2+. In the ISTANet-M^2, only the final output x_T is used in calculating the loss $\|\mathbf{x}_T - \mathbf{x}_o\|$ for backpropagation. In ISTANet-M^2+, however, we also consider the intermediate outputs $(\mathbf{x}_t)_{t=1}^{T-1}$ for each stage (block), and compute the intermediate losses together with the final loss as $\sum_{t=1}^{T} \|\mathbf{x}'_t - \mathbf{x}_o\|$ for backpropagation. Furthermore, we introduce a set of learnable parameters $(\alpha_t)_{t=1}^{T}$ into the losses as $\sum_{t=1}^{T} \|\alpha_t\mathbf{x}'_t - \mathbf{x}_o\|$, since we find the performance is improved by adding α into the loss. The structures of ISTANet-M^2 and ISTANet-M^2+ are shown in Figure 1.

Training details of DIP-M^3 The algorithm of DIP-M^3 is presented in Algorithm 1. In DIP-M^3 training, the step size ρ in *Gradient Descent Step* is fixed to $\rho = 1e-2$. In *Deep Prior Projection Step*, we use Adam [23] optimizer to optimize the decoder parameters θ, the learning rate is $1e-4$, weight decay is $5e-4$, the number of iterations used for achieving convergence of $\|g_\theta(\mathbf{z}) - \mathbf{x}_t^G\|$ is $T' = 200$. The choice of λ value was discussed before, and we pick $\lambda = 0.12$ for ORMM, and 0.22 for BRMM.

4.3 Additional experiments

Quantitative and qualitative results of baseline models To better compare the reconstruction performance of all test images, we provide the PSNR of each test image reconstructed by all the methods discussed above on ORMM and BRMM tasks in Table 2. We also show the qualitative results of the baseline models on ORMM Figure 5, 6, and BRMM Figure 7 tasks.

	Images	DnCNN	ISTANet-M^2	ISTANet-M^2+	DIP-M^2	DIP-M^3
ORMM (25%)	Babara	17.94	21.60	**22.56**	19.72	22.49
	Peppers	17.83	21.45	**21.98**	17.85	21.41
	House	18.56	22.46	24.49	21.07	**24.85**
	Foreman	17.46	22.42	23.85	20.42	**24.49**
	Boats	18.97	22.15	**23.54**	19.53	23.10
	Parrots	17.29	21.99	**22.51**	19.49	22.12
	Monarch	19.23	22.35	**22.90**	16.95	21.21
ORMM (50%)	Babara	21.26	**25.49**	23.39	22.47	24.77
	Peppers	21.67	24.95	23.48	22.32	**25.25**
	House	22.16	**27.69**	24.17	25.00	27.52
	Foreman	20.57	27.03	24.99	24.20	**27.56**
	Boats	22.82	**26.64**	25.32	23.60	26.06
	Parrots	19.94	**25.84**	24.16	22.08	24.89
	Monarch	22.67	**26.70**	24.13	22.12	25.31
BRMM	Babara	19.36	-	-	23.57	**27.01**
	Peppers	19.74	-	-	24.73	**25.56**
	House	21.54	-	-	27.56	**30.29**
	Foreman	20.09	-	-	27.73	**30.52**
	Boats	20.56	-	-	25.78	**28.60**
	Parrots	19.47	-	-	24.23	**28.18**
	Monarch	18.83	-	-	23.56	**27.63**

Table 2: PSNR (dB) of each test image on ORMM and BRMM tasks. We take $\lambda = 0.12$ and $\lambda = 0.22$ for DIP-M^3 on ORMM and BRMM tasks respectively.

Effects of number of looks We compare the performance of DIP-M^3 with different number of looks in Figure 8. We also show that the number of looks L has a strong effect on the choice of λ value in Figure 9 (a), this is reasonable since larger number of looks provides better gradient calculation in the *Gradient Descent Step* in Algorithm 1, which makes $\mathbf{x}_t^G$ to be more optimal. Note that $\mathbf{x}_t^P$ is generated to be as close as possible to $\mathbf{x}_t^G$, and at the same time to be more like a natural image given the implicit prior enforced by $g_\theta(\cdot)$. However, the optimization of $\|g_\theta(\mathbf{z}) - \mathbf{x}_t^G\|$ is not always guaranteed to be optimal given different iteration t, number of nested iterations T', and even the structure choice of $g_\theta(\cdot)$. In these cases, the information from a good reconstructed $\mathbf{x}_t^G$ can help compensate the generated $\mathbf{x}_t^P$, and yield a better fusion $\mathbf{x}_t$ compared with only using $\mathbf{x}_t^P$ as $\mathbf{x}_t$.

Number of iterations for convergence We plot the intermediate reconstructed images on both ORMM and BRMM tasks in Figure 10 and Figure 11. We also show the convergence curves of ORMM with 50% sampling rate in Figure 9 (b)-(f). We can find that the optimization of DIP-M^3 generally converges within 50 iterations on both ORMM and BRMM tasks. In ORMM task with patch size (32×32), it takes 1 to 1.5 seconds for DIP-M^3 to complete one iteration depending on the sampling rate m/n. In BRMM task, it takes about 2 seconds for each iteration.

Measurement matrix in RMM We also compare the effect of different measurement matrices in RMM. We consider two choices of measurement matrix A: row-subsampled (1) random orthogonal matrix $AA^T = I$; (2) random Gaussian matrix $A_{ij} \sim \mathcal{N}(0, 1)$. Table 3 and Figure 12 compare the results we obtain for these two choices. We can find that the choice of the measurement matrix does not have much effect on the performance of our model DIP-M^3.

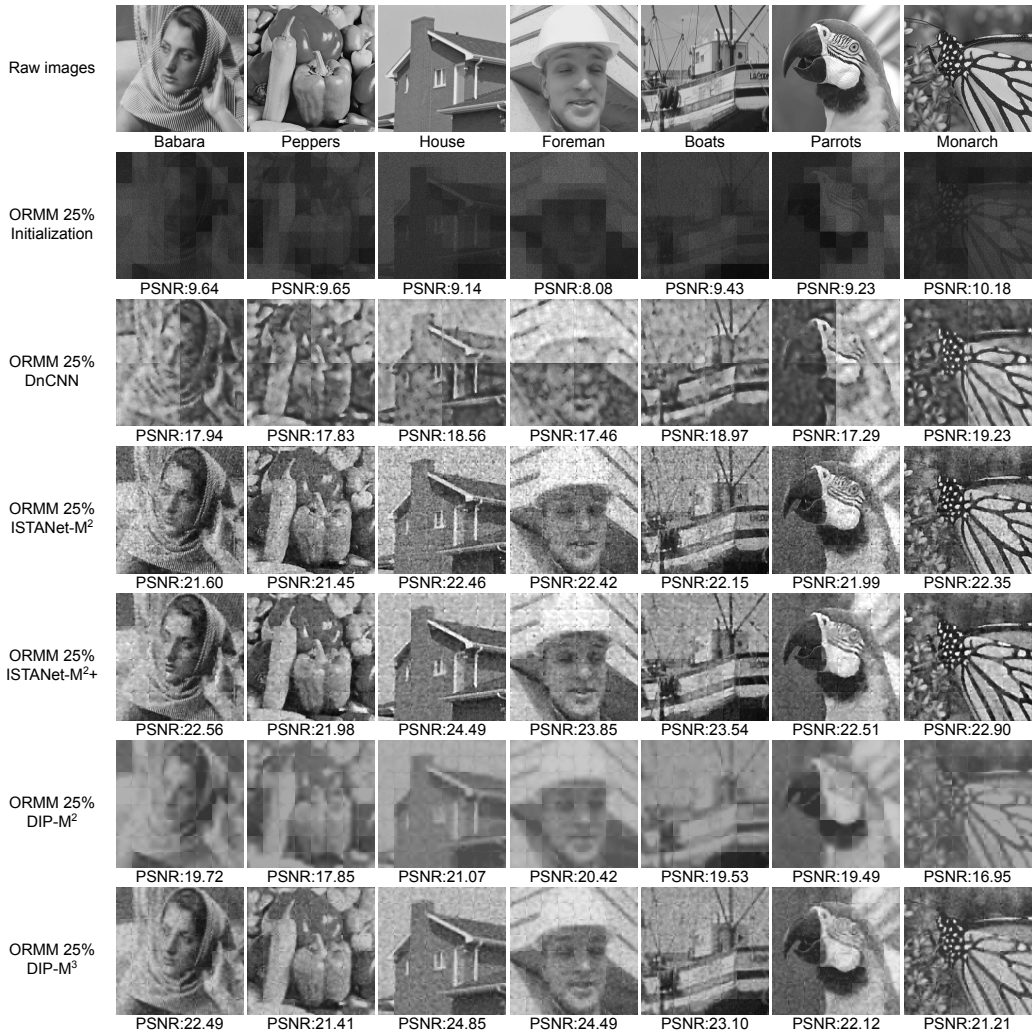


Figure 5: Qualitative comparison of the five models DnCNN, ISTANet-M^2, ISTANet-M^2+, DIP-M^2, DIP-M^3 on ORMM. In this figure, the sampling rate is chosen to be 25% and we collect 50 looks. The first row shows the raw test images. The second row is initialization $\frac{1}{L}\sum_{l=1}^{L}|A^T\mathbf{y}_l|$, which is the same for all models. From the third to the last row, we show the results of all models respectively.

Images	25% sampling rate		50% sampling rate	
	Orthogonal	Gaussian	Orthogonal	Gaussian
Babara	22.49	22.40	24.77	24.80
Peppers	21.41	21.34	25.25	25.24
House	24.85	24.79	27.52	27.64
Foreman	24.49	24.30	27.56	27.55
Boats	23.10	22.80	26.06	26.03
Parrots	22.12	21.88	24.89	24.90
Monarch	21.21	20.86	25.31	25.36

Table 3: PSNR (dB) of different measurement matrix on RMM tasks with 50-look. We show the PSNR of reconstructed images with random orthogonal and Gaussian matrix using DIP-M^3.

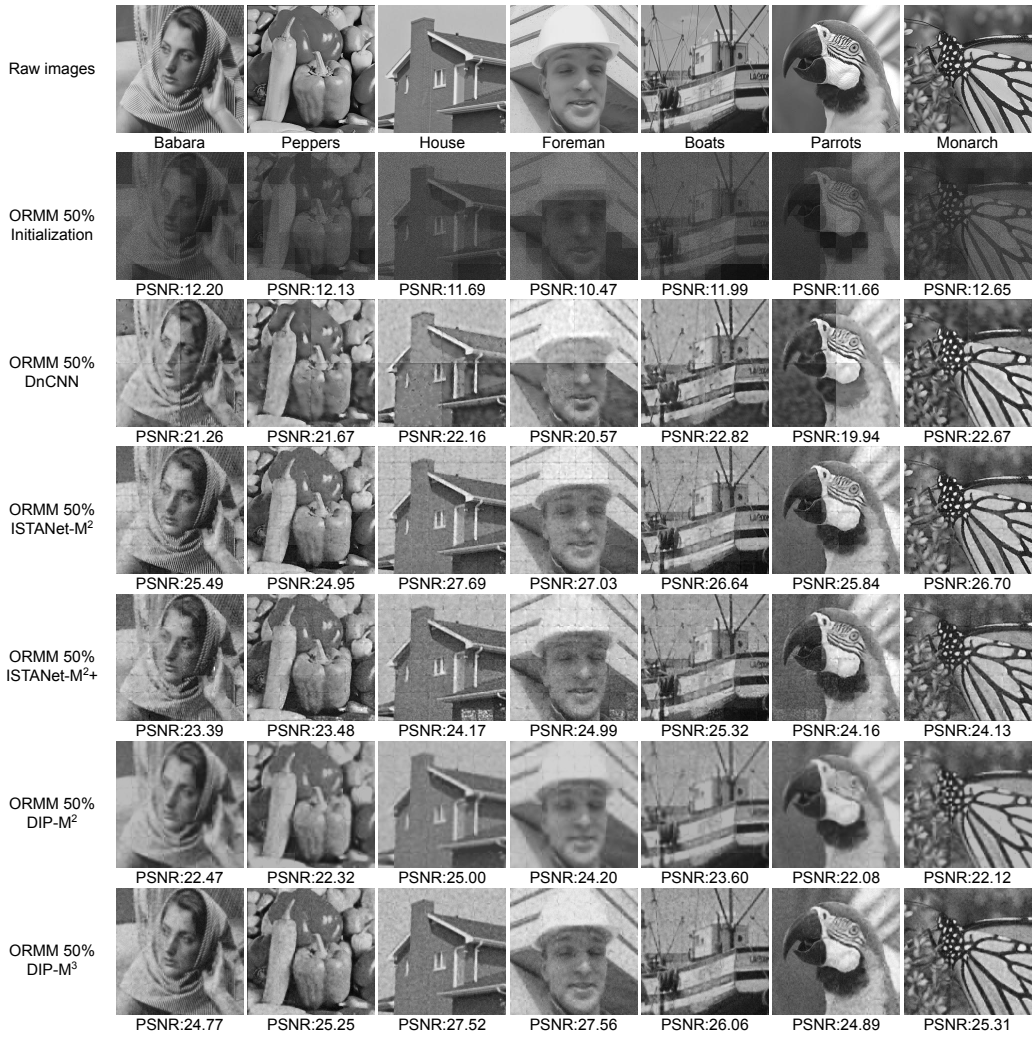
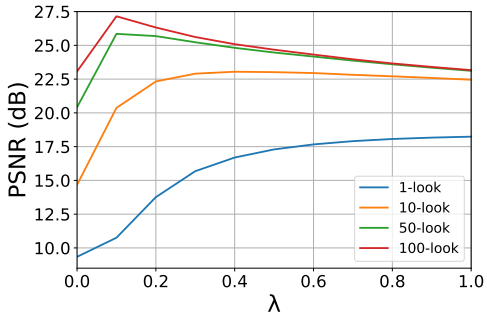


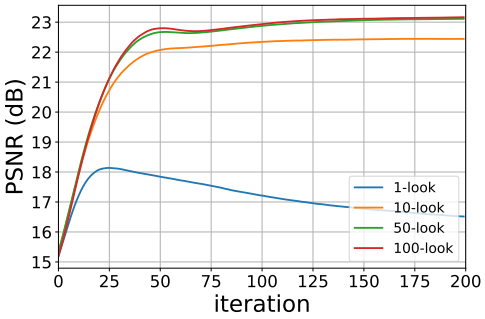
Figure 6: Qualitative comparison of the five models DnCNN, ISTANet-M^2, ISTANet-M^2+, DIP-M^2, DIP-M^3 on ORMM. In this figure, the sampling rate is chosen to be 50% and we collect 50 looks. The first row shows the raw test images. The second row is initialization $\frac{1}{L}\sum_{l=1}^{L}|A^T\mathbf{y}_l|$, which is the same for all models. From the third to the last row, we show the results of all models respectively.

Figure 7: Qualitative comparison on BRMM with 50 looks. The second row visualize the initialization $\frac{1}{L}\sum_{l=1}^{L}|\mathbf{y}_l|$, which is the same for all models. From the third row to the last row, we show the reconstructed results from DnCNN, DIP-M^2 and DIP-M^3 respectively.

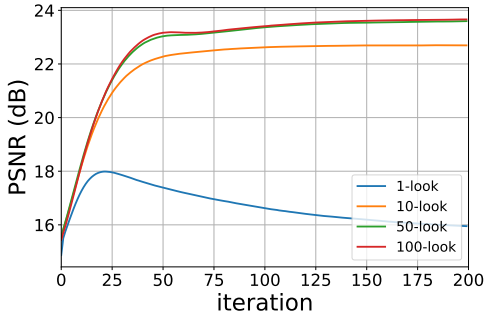
Figure 8: Comparison on ORMM task with 50% sampling rate with different number of looks. The first row shows the raw test images. The second to the last row shows the results of our model DIP-M^3 with 1,10,50,100 looks respectively.



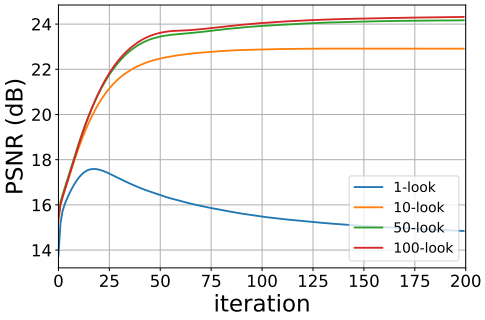
(a) Sensitivity to λ with different looks



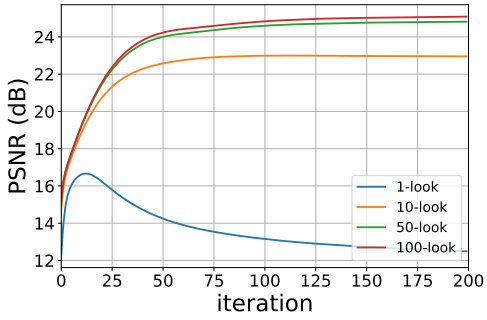
(b) Convergence of different looks $\lambda = 1.0$



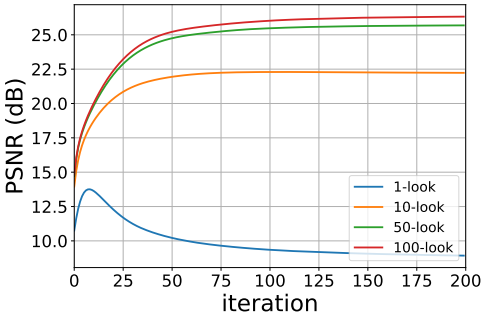
(c) Convergence of different looks $\lambda = 0.8$



(d) Convergence of different looks $\lambda = 0.6$



(e) Convergence of different looks $\lambda = 0.4$



(f) Convergence of different looks $\lambda = 0.2$

Figure 9: Effects of number of looks. We show the average PSNR of our model DIP-M^3 with 1-look, 10-look, 50-look, 100-look across all candidate λ on ORMM task in (a). We also plot the convergence of our model DIP-M^3 under different number of looks when $\lambda = 1.0, 0.8, 0.6, 0.4, 0.2$ in (b)-(f).

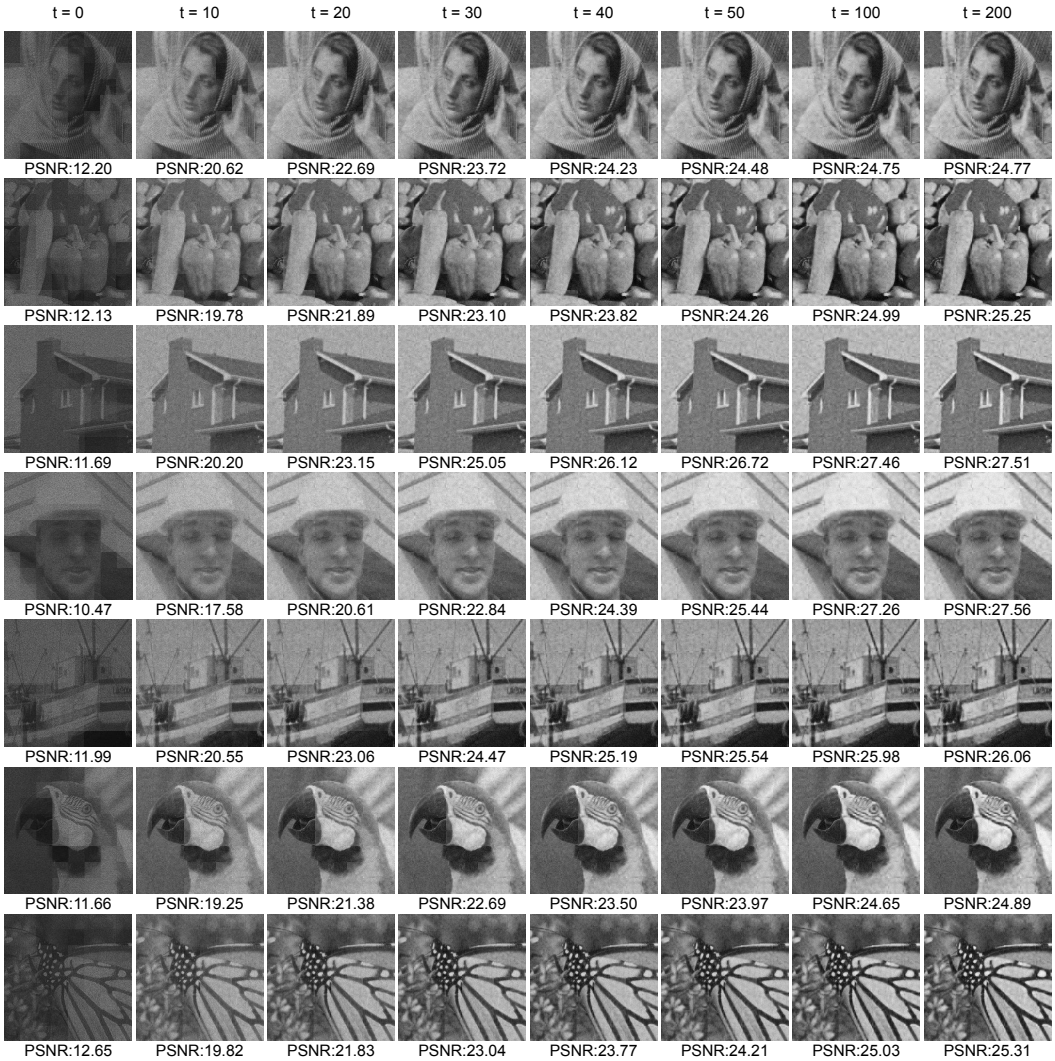


Figure 10: Visualization of reconstructed images by our model DIP-M^3 at iteration $t = 0, 10, 20, 30, 40, 50, 100, 200$ on ORMM. The sampling rate is 50% and $L = 50$.

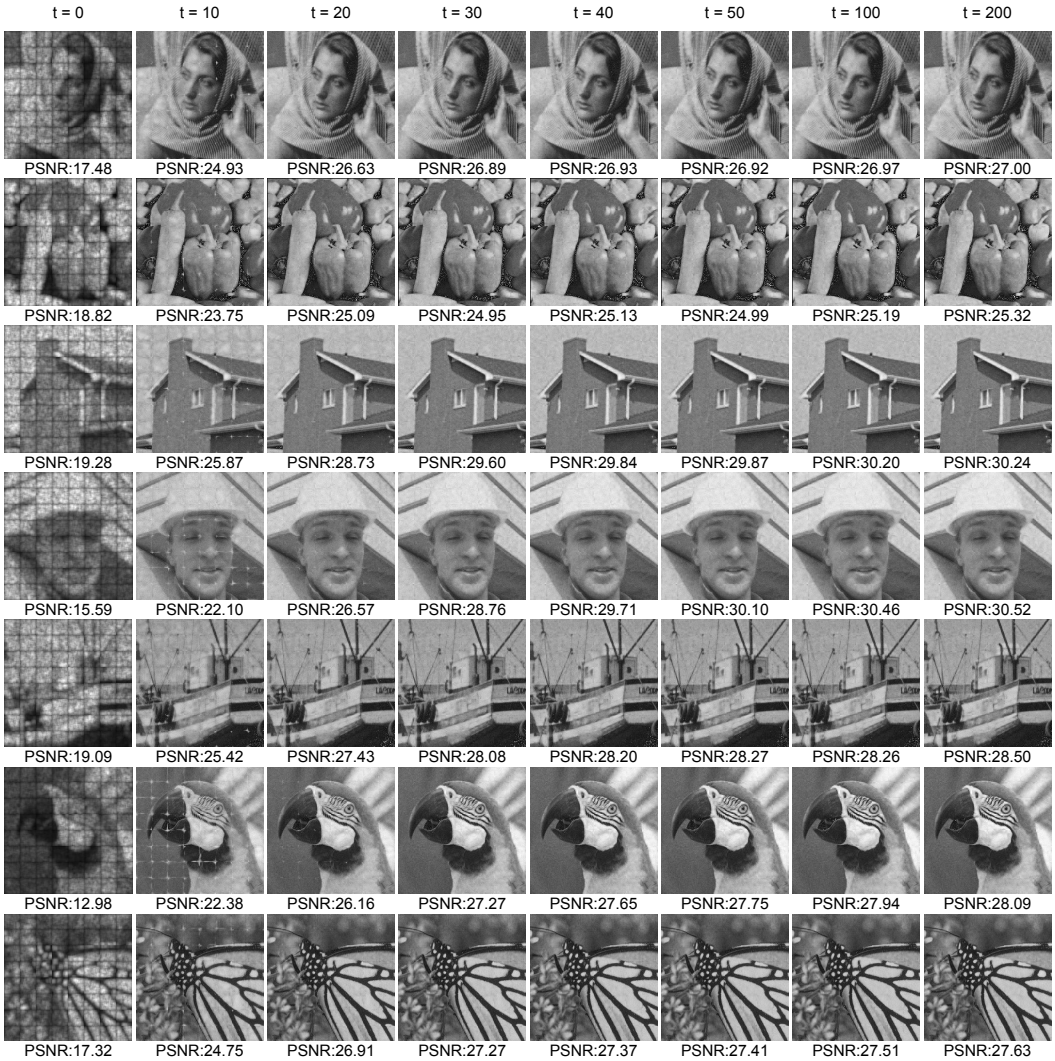


Figure 11: Visualization of reconstructed images by our model DIP-M^3 at iteration $t = 0, 10, 20, 30, 40, 50, 100, 200$ on ORMM, $L = 50$.

Figure 12: Qualitative comparison on different measurement matrix with 50 looks on RMM task. The first row shows the raw test images. The second and third row are RMM (25% sampling rate) with random orthogonal and Gaussian measurement matrix. The fourth and fifth row are RMM (50% sampling rate) with random orthogonal and Gaussian measurement matrix.