# Neural Distributed Image Compression Using Common Information

Nitish Mital[1,*], Ezgi Özyılkan[1,†], Ali Garjani[1,‡], and Deniz Gündüz[*]

2022 Data Compression Conference (DCC)

March 22-24, 2022

1 Equal contribution.
*Dept. of Electrical and Electronics Engineering, Imperial College London.
†Dept. of Electrical and Computer Engineering, New York University.
‡Section of Mathematics, EPFL.

# System Model: Point-to-point

Source coding

# System Model: Point-to-point

Source coding

- Lossless

# System Model: Point-to-point

Source coding

- Lossless
- Lossy

# System Model: Point-to-point

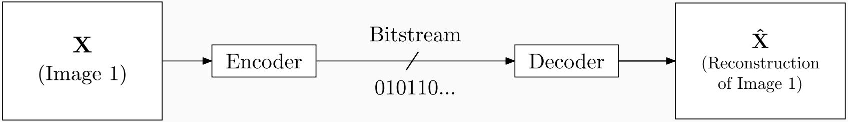
Source coding

- Lossless
- Lossy ✓

# System Model: Point-to-point

Source coding

- Lossless
- Lossy ✓

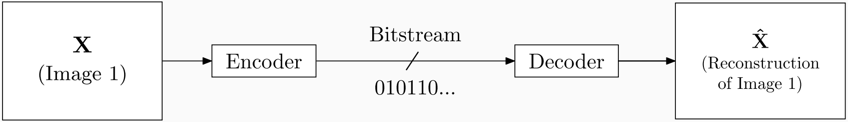



System model for point-to-point source coding.

# System Model: Point-to-point

Source coding

- Lossless
- Lossy ✓



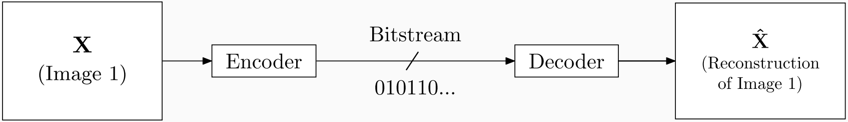

System model for point-to-point source coding.

Two competing goals in lossy compression:

# System Model: Point-to-point

Source coding

- Lossless
- Lossy ✓




System model for point-to-point source coding.

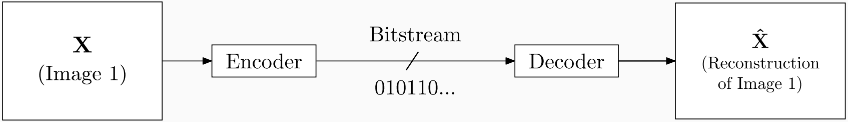
Two competing goals in lossy compression:

- Rate

# System Model: Point-to-point

Source coding

- Lossless
- Lossy ✓



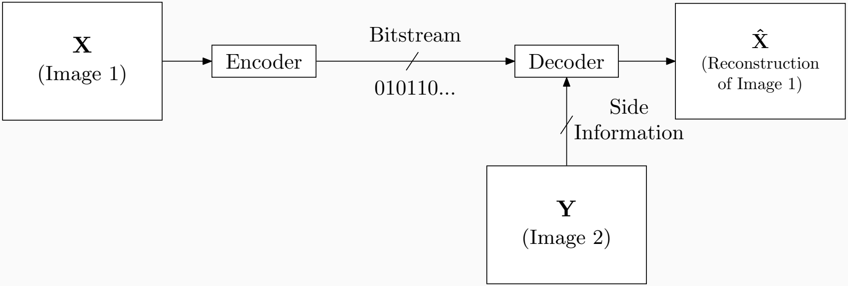

System model for point-to-point source coding.

Two competing goals in lossy compression:

- Rate
- Distortion

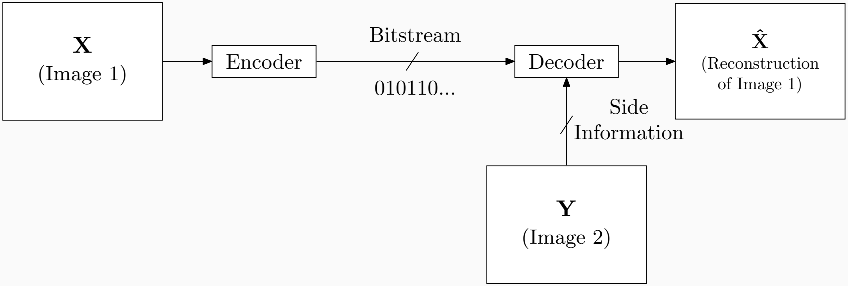
# System Model: Distributed Source Coding (DSC)

# System Model: Distributed Source Coding (DSC)




System model for source coding with decoder-only side information.

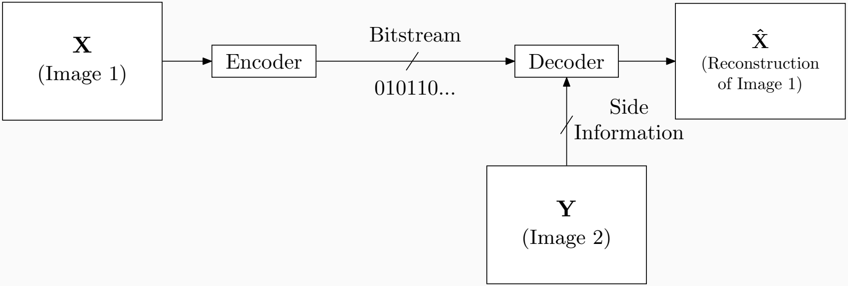
# System Model: Distributed Source Coding (DSC)




System model for source coding with decoder-only side information.

- Slepian and Wolf, 1973 Lossless compression.

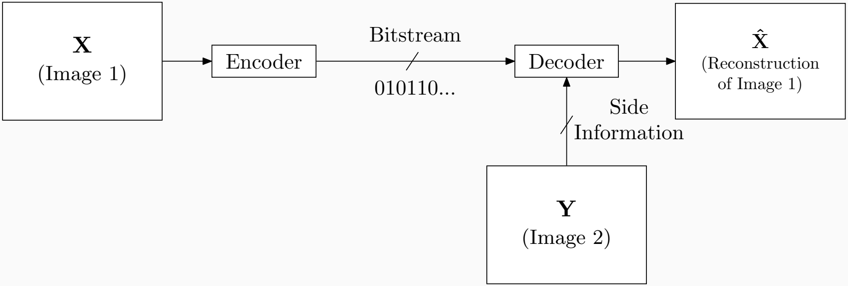
# System Model: Distributed Source Coding (DSC)




System model for source coding with decoder-only side information.

- Slepian and Wolf, 1973 Lossless compression.
- Wyner and Ziv, 1976 Lossy compression.

# System Model: Distributed Source Coding (DSC)

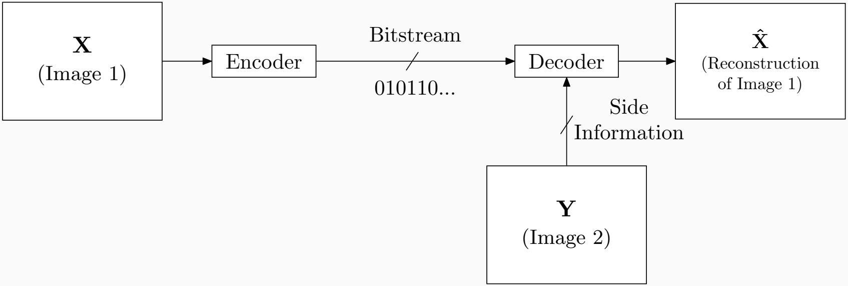



System model for source coding with decoder-only side information.

- Slepian and Wolf, 1973 Lossless compression.
- Wyner and Ziv, 1976 Lossy compression.

- Pradhan and Ramchandran, 2003 (DISCUS) DSC using syndromes.

# System Model: Distributed Source Coding (DSC)




System model for source coding with decoder-only side information.

- Slepian and Wolf, 1973 Lossless compression.
- Wyner and Ziv, 1976 Lossy compression.

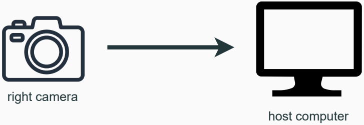
- Pradhan and Ramchandran, 2003 (DISCUS) DSC using syndromes.
- Girod et al., 2005 Distributed Video Coding.

# Motivation for DSC Setup

Pair of correlated images with overlapping fields of view.

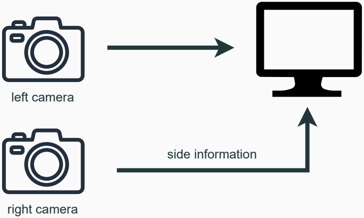
# Motivation for DSC Setup

Pair of correlated images with overlapping fields of view.

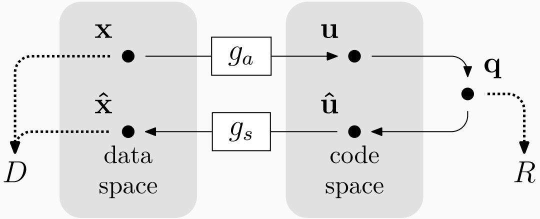


# Motivation for the DSC Setup

Pair of correlated images with overlapping fields of view.



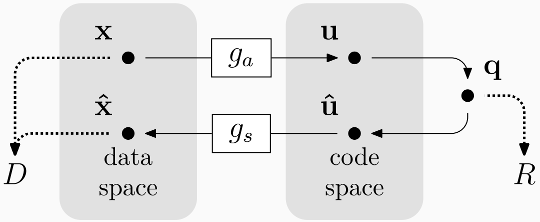
# Lossy Compression: Transform Coding




Transform coding framework[1].

[1]Figure provided is from Ballé et al., 2017.

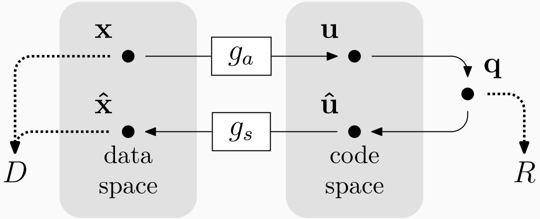
# Lossy Compression: Transform Coding




Transform coding framework[1].

$$\min \; R$$
$$\text{subject to } \mathbb{E}[D] \leq D_c$$

[1]Figure provided is from Ballé et al., 2017.




Transform coding framework[1].

$$\min\ R$$
$$\text{subject to } \mathbb{E}[D] \leq D_c$$

$$L = R + \lambda D$$

[1] Figure provided is from Ballé et al., 2017.

# Lossy Image Compression using DNNs

- Ballé et. al., 2017 Deep end-to-end image compression.

# Lossy Image Compression using DNNs

- Ballé et. al., 2017 Deep end-to-end image compression.

$$L(\mathbf{g}_a, \mathbf{g}_s) = R + \lambda D,$$

$$R = \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[\underbrace{-\log p(\tilde{\mathbf{u}})}_{\text{latent}}]$$

- Ballé et. al., 2017 Deep end-to-end image compression.

$$L(\mathbf{g}_a, \mathbf{g}_s) = R + \lambda D,$$

$$R = \mathbb{E}_{\mathbf{x} \sim p(\mathbf{x})}[\underbrace{-\log p(\tilde{\mathbf{u}})}_{\text{latent}}]$$

- Ballé et. al., 2018 Deep end-to-end compression using hyperpriors to model the prior distribution of the latent variables.

- Ballé et. al., 2017 Deep end-to-end image compression.

$$L(\mathbf{g}_a, \mathbf{g}_s) = R + \lambda D,$$

$$R = \mathbb{E}_{\mathsf{x}\sim p(\mathsf{x})}[\underbrace{-\log p(\tilde{\mathsf{u}})}_{\text{latent}}]$$

- Ballé et. al., 2018 Deep end-to-end compression using hyperpriors to model the prior distribution of the latent variables.

$$R = \mathbb{E}_{\mathsf{x}\sim p(\mathsf{x})}[\underbrace{-\log p(\tilde{\mathsf{u}} \mid \tilde{\mathsf{z}})}_{\text{latent}}] + \mathbb{E}_{\mathsf{x}\sim p(\mathsf{x})}[\underbrace{-\log p(\tilde{\mathsf{z}})}_{\text{hyperprior}}]$$

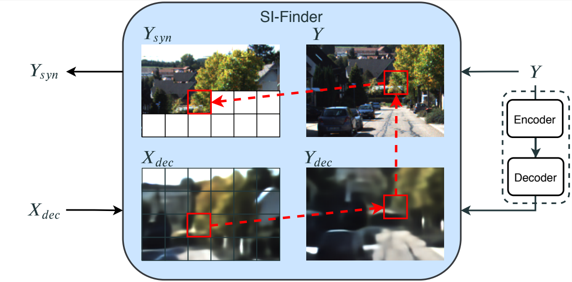
# Deep Image Compression with Side Information

# Deep Image Compression with Side Information

- Ayzik and Avidan, 2020 (DSIN) - Reconstruct an intermediate image, then find corresponding patches in the side information image, which they use to refine the reconstructed image.

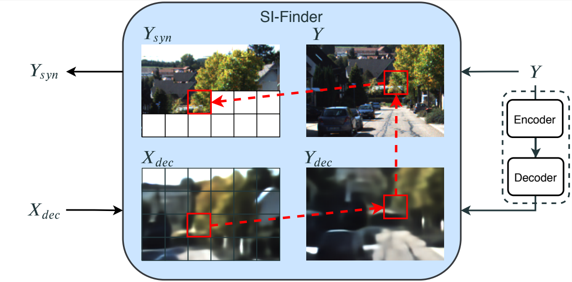
# Deep Image Compression with Side Information

- Ayzik and Avidan, 2020 (DSIN) - Reconstruct an intermediate image, then find corresponding patches in the side information image, which they use to refine the reconstructed image.



# Deep Image Compression with Side Information

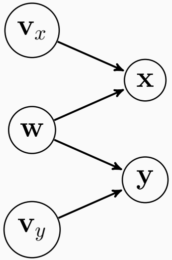
- Ayzik and Avidan, 2020 (DSIN) - Reconstruct an intermediate image, then find corresponding patches in the side information image, which they use to refine the reconstructed image.



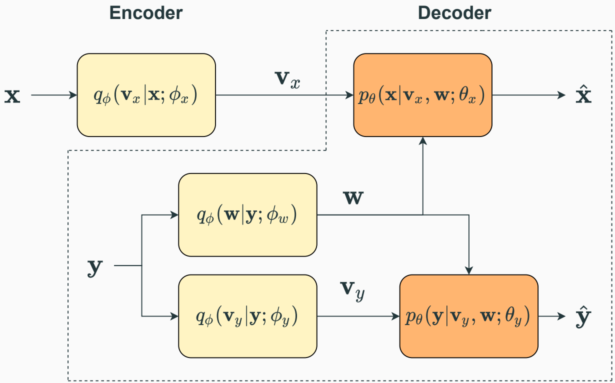

- Whang et. al., 2021 - Transform side information image to a latent space. Use it together with the received latent variable to jointly reconstruct the image.

# Proposed Solution

# Distributed Source Coding Architecture

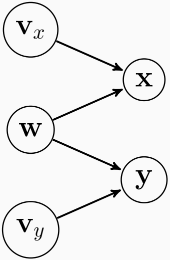



Graphical model. [Wang et al., 2017]

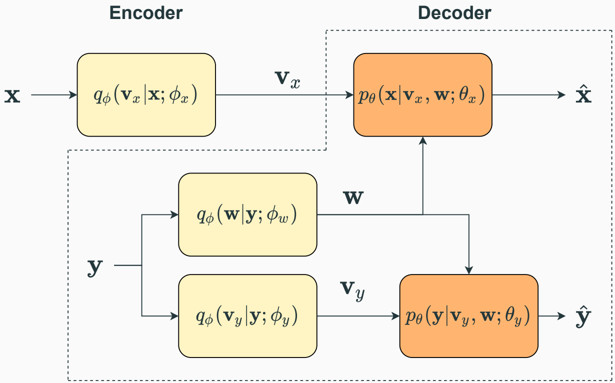



Distributed source coding architecture.

# Distributed Source Coding Architecture




Graphical model. [Wang et al., 2017]

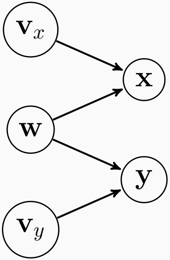



Distributed source coding architecture.

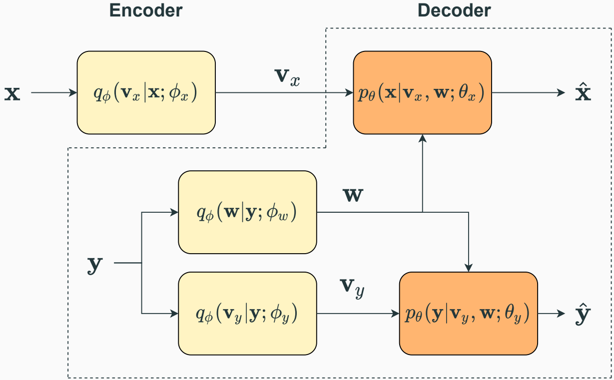
$\mathbf{v}_x$, $\mathbf{v}_y$ and $\mathbf{w}$ are independent latent variables:

- $\mathbf{w}$ - common information.
- $\mathbf{v}_x$ and $\mathbf{v}_y$ - independent information of $\mathbf{x}$ and $\mathbf{y}$.
- Generate $\mathbf{x}$ and $\mathbf{y}$ as above.

# Distributed Source Coding Architecture



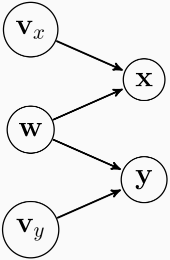

Graphical model. [Wang et al., 2017]



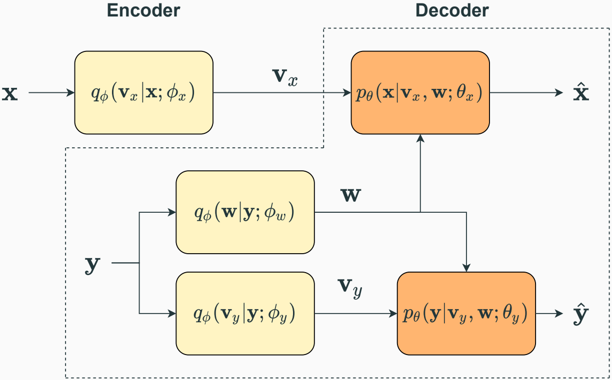

Distributed source coding architecture.

$$p(\mathsf{x},\mathsf{y},\mathsf{w},\mathsf{v}_x,\mathsf{v}_y) = p(\mathsf{w})p(\mathsf{v}_x)p(\mathsf{v}_y)p_{\boldsymbol{\theta}}(\mathsf{x} \mid \mathsf{w},\mathsf{v}_x;\boldsymbol{\theta}_x)p_{\boldsymbol{\theta}}(\mathsf{y} \mid \mathsf{w},\mathsf{v}_y;\boldsymbol{\theta}_y)$$

Factored joint prior distribution of the latent variables emerging from the graphical model.




Graphical model. [Wang et al., 2017]




Distributed source coding architecture.

$$q_\phi(\mathrm{w}, \mathrm{v}_x, \mathrm{v}_y \mid \mathrm{x}, \mathrm{y}) = q_\phi(\mathrm{v}_x \mid \mathrm{x}; \phi_x) q_\phi(\mathrm{w} \mid \mathrm{y}; \phi_w) q_\phi(\mathrm{v}_y \mid \mathrm{y}; \phi_y)$$

Factored variational approximation of the posterior distribution emerging from the system architecture.

# Loss Function

$$\min_{\boldsymbol{\phi},\boldsymbol{\theta}} \mathbb{E}_{\mathsf{x},\mathsf{y}\sim p(\mathsf{x},\mathsf{y})} D_{\mathrm{KL}} \left[ q_{\boldsymbol{\phi}}(\tilde{\mathsf{v}}_x, \mathsf{v}_y, \mathsf{w} \mid \mathsf{x}, \mathsf{y}) \mid\mid p(\tilde{\mathsf{v}}_x, \mathsf{v}_y, \mathsf{w} \mid \mathsf{x}, \mathsf{y}) \right]$$

# Loss Function

$$
\begin{aligned}
&\min_{\boldsymbol{\phi},\boldsymbol{\theta}} \mathbb{E}_{\mathsf{x},\mathsf{y}\sim p(\mathsf{x},\mathsf{y})} D_{\mathrm{KL}} \left[ q_{\boldsymbol{\phi}}(\tilde{\mathsf{v}}_x, \mathsf{v}_y, \mathsf{w} \mid \mathsf{x}, \mathsf{y}) \mid\mid p(\tilde{\mathsf{v}}_x, \mathsf{v}_y, \mathsf{w} \mid \mathsf{x}, \mathsf{y}) \right] \\
&= \min_{\boldsymbol{\phi},\boldsymbol{\theta}} \mathbb{E}_{\mathsf{x},\mathsf{y}\sim p(\mathsf{x},\mathsf{y})} \mathbb{E}_{\tilde{\mathsf{v}}_x, \mathsf{v}_y, \mathsf{w}\sim q_{\boldsymbol{\phi}}} \Bigg( \Big( \log q_{\boldsymbol{\phi}}(\tilde{\mathsf{v}}_x \mid \mathsf{x}; \boldsymbol{\phi}_x) + \log q_{\boldsymbol{\phi}}(\mathsf{v}_y \mid \mathsf{y}; \boldsymbol{\phi}_y) + \log q_{\boldsymbol{\phi}}(\mathsf{w} \mid \mathsf{y}; \boldsymbol{\phi}_f) \Big) \\
&\qquad - \Big( \underbrace{\log p_{\boldsymbol{\theta}}(\mathsf{x} \mid \mathsf{w}, \tilde{\mathsf{v}}_x; \boldsymbol{\theta}_x)}_{D_x} + \underbrace{\log p_{\boldsymbol{\theta}}(\mathsf{y} \mid \mathsf{w}, \mathsf{v}_y; \boldsymbol{\theta}_y)}_{D_y} + \underbrace{\log p(\mathsf{w})}_{R_w} + \underbrace{\log p(\tilde{\mathsf{v}}_x)}_{R_x} + \underbrace{\log p(\mathsf{v}_y)}_{R_y} \Big) \Bigg) \\
&\qquad + \mathit{const.}
\end{aligned}
$$

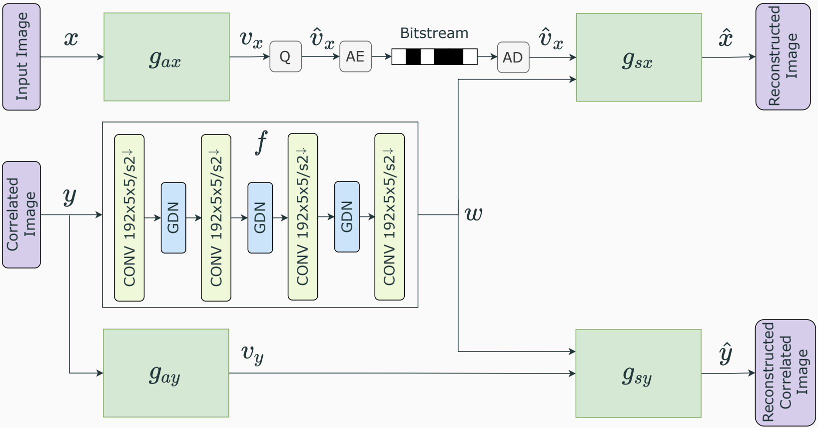
# Loss Function

$$
\begin{aligned}
&\min_{\boldsymbol{\phi},\boldsymbol{\theta}} \mathbb{E}_{\mathsf{x},\mathsf{y}\sim p(\mathsf{x},\mathsf{y})} D_{\mathrm{KL}}\left[q_{\boldsymbol{\phi}}(\tilde{\mathsf{v}}_x, \mathsf{v}_y, \mathsf{w} \mid \mathsf{x}, \mathsf{y}) \mid\mid p(\tilde{\mathsf{v}}_x, \mathsf{v}_y, \mathsf{w} \mid \mathsf{x}, \mathsf{y})\right] \\
&= \min_{\boldsymbol{\phi},\boldsymbol{\theta}} \mathbb{E}_{\mathsf{x},\mathsf{y}\sim p(\mathsf{x},\mathsf{y})} \mathbb{E}_{\tilde{\mathsf{v}}_x,\mathsf{v}_y,\mathsf{w}\sim q_{\boldsymbol{\phi}}} \Bigg( \Big( \log q_{\boldsymbol{\phi}}(\tilde{\mathsf{v}}_x \mid \mathsf{x}; \boldsymbol{\phi}_x) + \log q_{\boldsymbol{\phi}}(\mathsf{v}_y \mid \mathsf{y}; \boldsymbol{\phi}_y) + \log q_{\boldsymbol{\phi}}(\mathsf{w} \mid \mathsf{y}; \boldsymbol{\phi}_f) \Big) \\
&\qquad - \Big( \underbrace{\log p_{\boldsymbol{\theta}}(\mathsf{x} \mid \mathsf{w}, \tilde{\mathsf{v}}_x; \boldsymbol{\theta}_x)}_{D_x} + \underbrace{\log p_{\boldsymbol{\theta}}(\mathsf{y} \mid \mathsf{w}, \mathsf{v}_y; \boldsymbol{\theta}_y)}_{D_y} + \underbrace{\log p(\mathsf{w})}_{R_w} + \underbrace{\log p(\tilde{\mathsf{v}}_x)}_{R_x} + \underbrace{\log p(\mathsf{v}_y)}_{R_y} \Big) \Bigg) \\
&\qquad + \mathit{const.}
\end{aligned}
$$

Adding weights $\alpha$, $\beta$ and $\lambda$ to control the contribution of the terms, we write:

$$
L(\mathsf{g}_{ax}, \mathsf{g}_{sx}, \mathsf{g}_{ay}, \mathsf{g}_{sy}, \mathsf{f}) = (R_x + \lambda D_x) + \alpha\,(R_y + \lambda D_y) + \beta R_w,
$$

# Neural Network Architecture



# Experimental Setup and Results

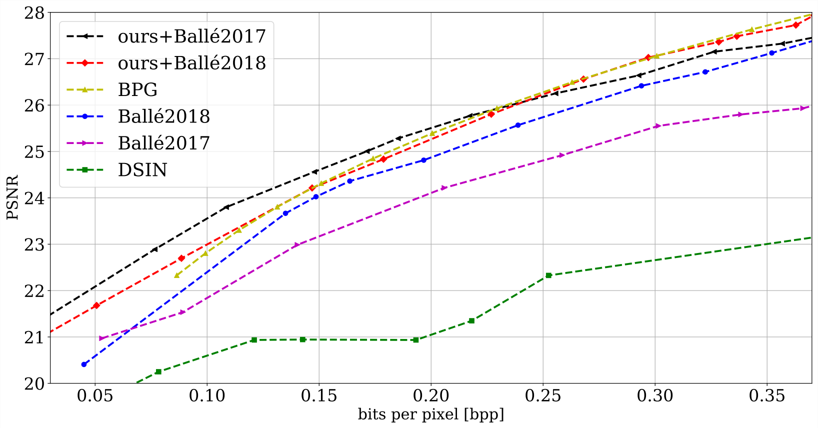
# Datasets

KITTI Stereo

Cityscape

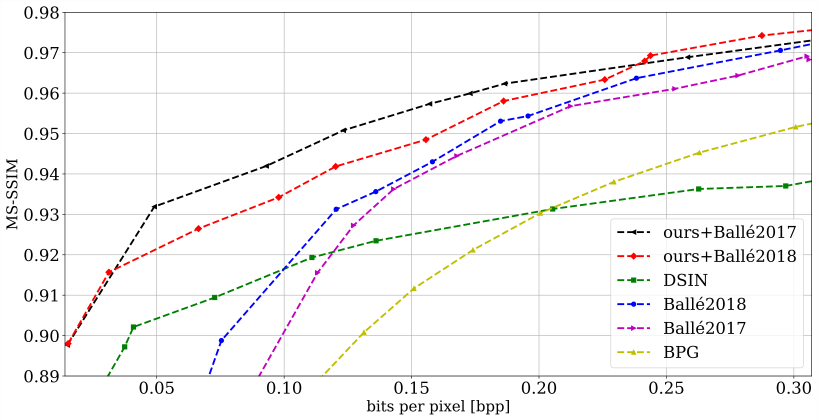
Example stereo image pairs from KITTI Stereo and Cityscape.

# Results with KITTI Stereo



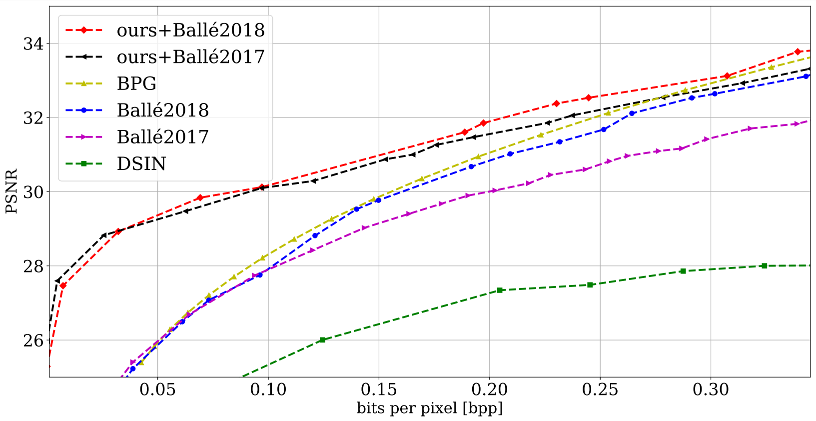

Comparison of different models in terms of MSE.

# Results with KITTI Stereo



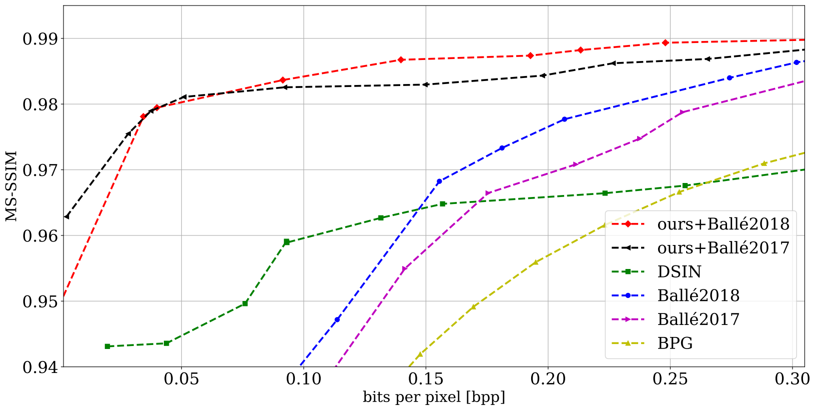

Comparison of different models in terms of MS-SSIM.

# Results with Cityscape dataset



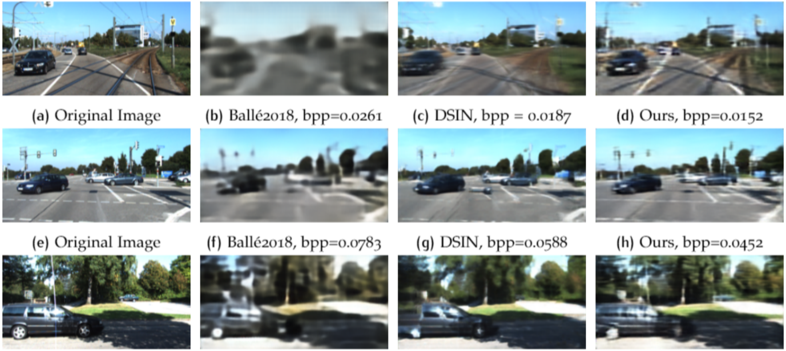

Comparison of different models in terms of MSE.

# Results with Cityscape dataset




Comparison of different models in terms of MS-SSIM.

# Visual Comparisons



(a) Original Image (b) Ballé2018, bpp=0.0261 (c) DSIN, bpp = 0.0187 (d) Ours, bpp=0.0152

(e) Original Image (f) Ballé2018, bpp=0.0783 (g) DSIN, bpp=0.0588 (h) Ours, bpp=0.0452

(i) Original Image (j) Ballé2018, bpp = 0.0827 (k) DSIN, bpp = 0.0741 (l) Ours, bpp = 0.0521

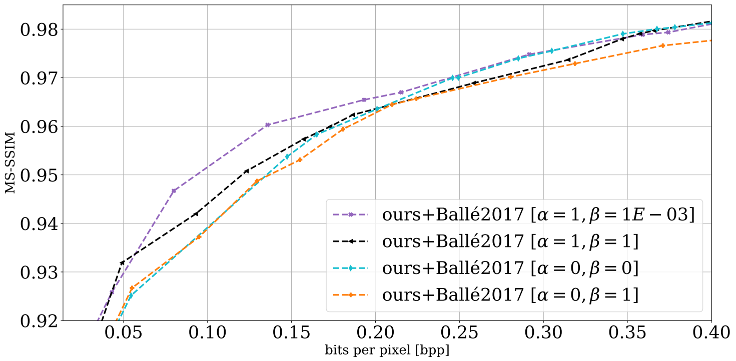
"Ours" refers to "Ours + Ballé2017" model.

# Visual Comparisons

(a) DSIN, bpp=0.0449

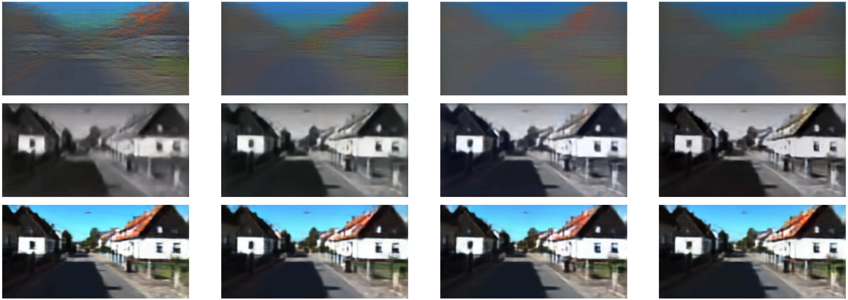
(b) Ours, bpp=0.0431

# Effect of hyperparameters $\alpha$ and $\beta$




$$(R_x + \lambda D_x) + \alpha (R_y + \lambda D_y) + \beta R_w \quad (1)$$

# Effect of hyperparameters $\alpha$ and $\beta$



(a) $\alpha = 1, \beta = 10^{-3}$ bpp $= 0.1281$

(b) $\alpha = 0, \beta = 0$ bpp $= 0.1572$

(c) $\alpha = 1, \beta = 1$ bpp $= 0.1658$

(d) $\alpha = 0, \beta = 1$ bpp $= 0.1731$

Common information (1st row), private information (2nd row) decomposition, reconstructed images with similar reconstruction quality (3rd row).

# Conclusion

- Practical implementation of Wyner-Ziv lossy source compression with decoder-only side information.

# Conclusion

- Practical implementation of Wyner-Ziv lossy source compression with decoder-only side information.
- First result that can truly benefit from side information.

# Conclusion

- Practical implementation of Wyner-Ziv lossy source compression with decoder-only side information.
- First result that can truly benefit from side information.
- Neural networks learn side information statistics despite not having access to its realization.

# Conclusion

- Practical implementation of Wyner-Ziv lossy source compression with decoder-only side information.
- First result that can truly benefit from side information.
- Neural networks learn side information statistics despite not having access to its realization.
- A novel approach by disentangling common and private latent generating variables.

# Conclusion

- Practical implementation of Wyner-Ziv lossy source compression with decoder-only side information.
- First result that can truly benefit from side information.
- Neural networks learn side information statistics despite not having access to its realization.
- A novel approach by disentangling common and private latent generating variables.
- Demonstrated significant reductions in compression rates by using the common information extracted from side information at the decoder.

# Conclusion

- Practical implementation of Wyner-Ziv lossy source compression with decoder-only side information.
- First result that can truly benefit from side information.
- Neural networks learn side information statistics despite not having access to its realization.
- A novel approach by disentangling common and private latent generating variables.
- Demonstrated significant reductions in compression rates by using the common information extracted from side information at the decoder.
- Common information consists of global texture and color information. The quantity of each can be controlled using hyperparameters.

Code publicly available at: *`https://github.com/ipc-lab/NDIC`*