Robotics and
Visual Intelligence Lab

MINE: Towards Continuous Depth MPI with NeRF for Novel View Synthesis

Jiaxin Li[1*], Zijian Feng[1*], Qi She[1], Henghui Ding[1], Changhu Wang[1], Gim Hee Lee[2]
[1]ByteDance, [2]National University of Singapore

ICCV 2021

Gyeongsu Cho @UNIST

2022.04.28 (Thu)

Introduction

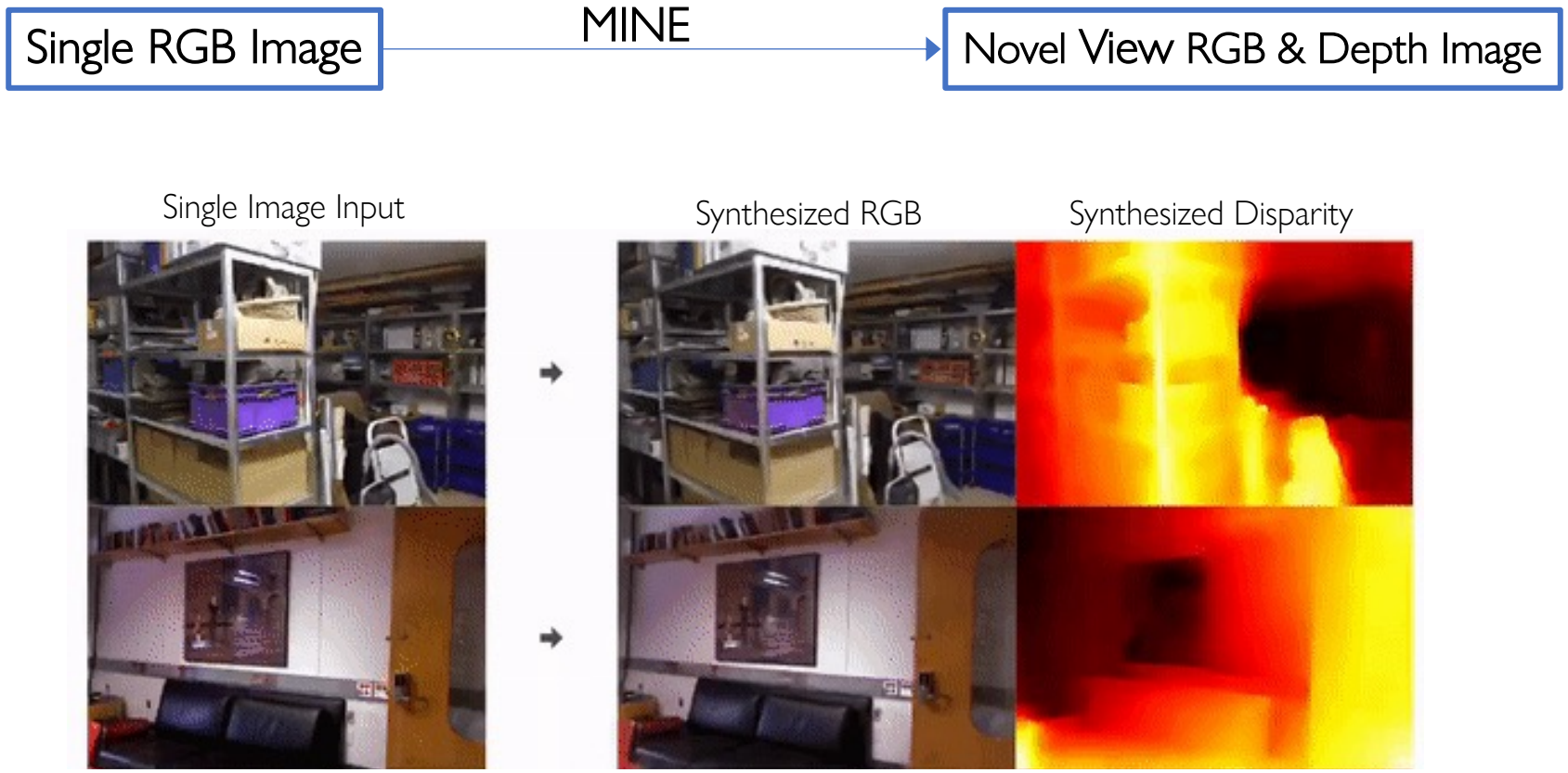
Method

Experiments

Conclusion

Introduction

- Single-View Novel View Synthesis



Introduction

NeRF [1]

[1] Ben Mildenhall, et al, Nerf: Representing scenes as neural radiance fields for view synthesis. In ECCV, 2020

Introduction

NeRF [1]

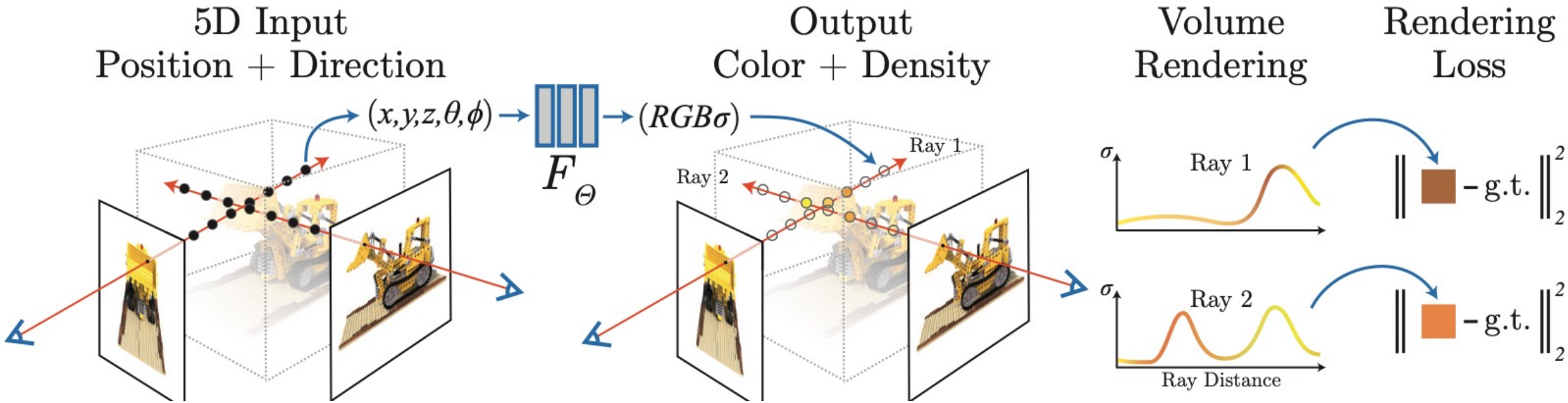
Inputs: sparsely sampled images of scene

Outputs: new views of same scene

[1] Ben Mildenhall, et al, Nerf: Representing scenes as neural radiance fields for view synthesis. In ECCV, 2020

Introduction

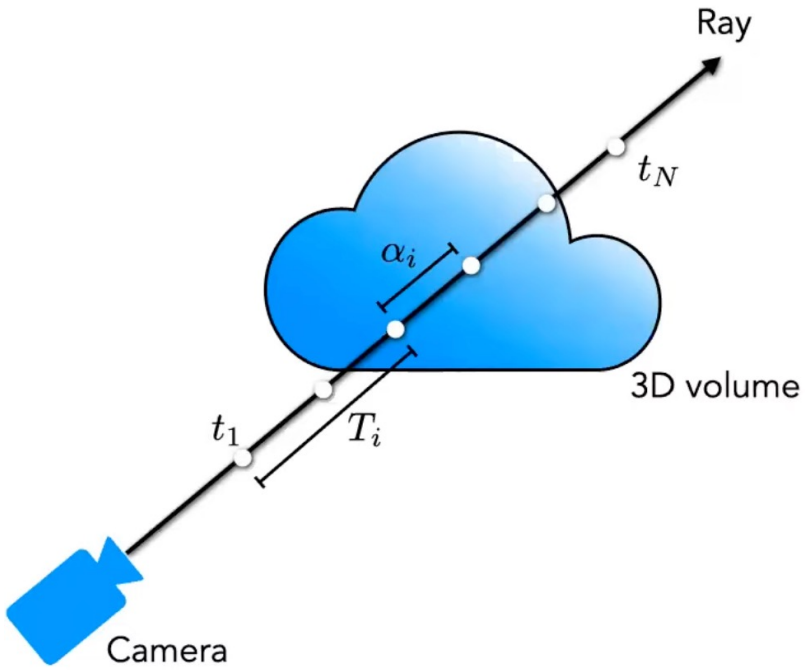
NeRF [1]



[1] Ben Mildenhall, et al, Nerf: Representing scenes as neural radiance fields for view synthesis. In ECCV, 2020

Introduction

Volume rendering in NeRF [1]



[1] Ben Mildenhall, et al, Nerf: Representing scenes as neural radiance fields for view synthesis. In ECCV, 2020

Introduction

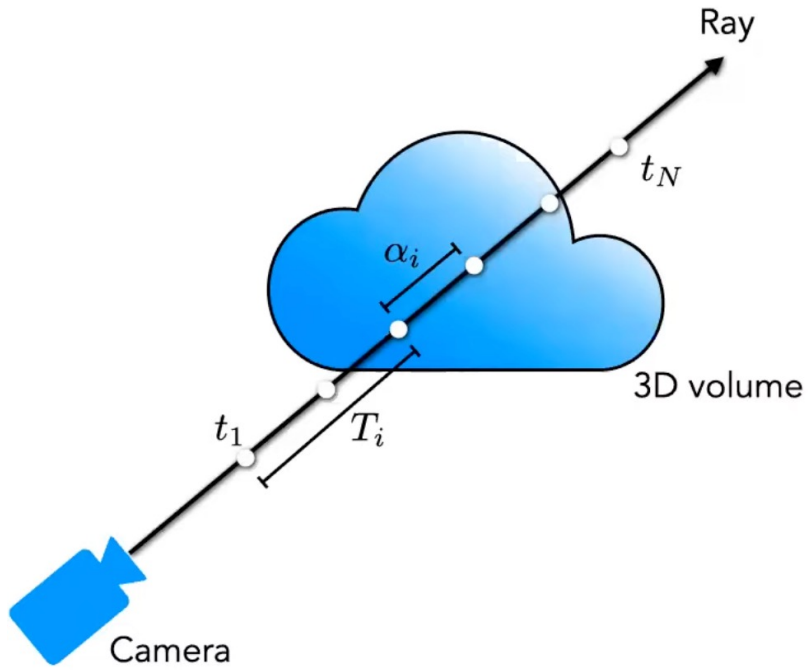
Volume rendering in NeRF [1]

Rendering model for ray r(t) = o + td:

$$C \approx \sum_{i=1}^{N} T_i \alpha_i c_i \quad [2]$$

weights

colors



[1] Ben Mildenhall, et al, Nerf: Representing scenes as neural radiance fields for view synthesis. In ECCV, 2020
[2] Max, N.: Optical models for direct volume rendering. In IEEE Transactions on Visualization and Computer Graphics, 1995

Introduction

Volume rendering in NeRF [1]

Rendering model for ray r(t) = o + td:

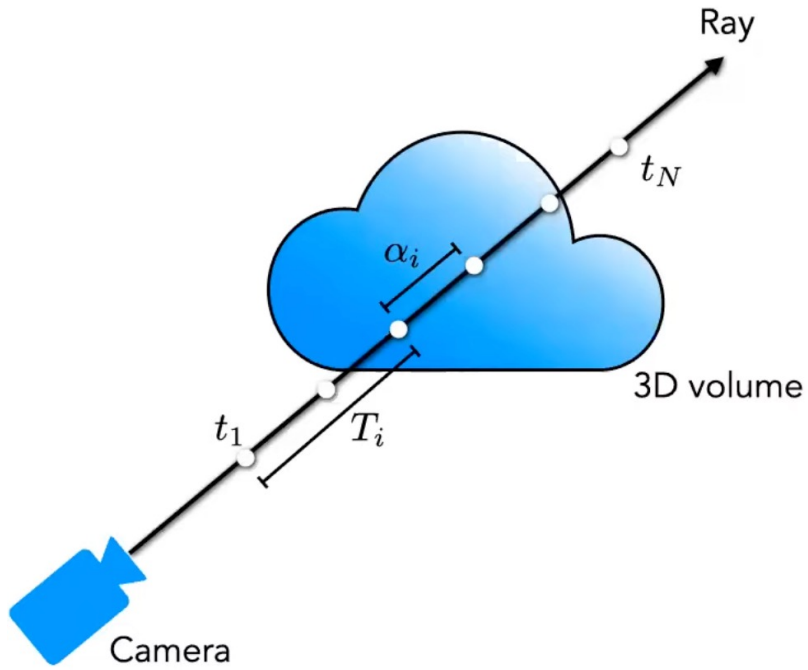
$$C \approx \sum_{i=1}^{N} T_i \alpha_i c_i \quad [2]$$

colors

weights

How much light is blocked earlier along ray:

$$T_i = \prod_{j=1}^{i-1} (1 - \alpha_j)$$



[1] Ben Mildenhall, et al, Nerf: Representing scenes as neural radiance fields for view synthesis. In ECCV, 2020
[2] Max, N.: Optical models for direct volume rendering. In IEEE Transactions on Visualization and Computer Graphics, 1995

Introduction

Volume rendering in NeRF [1]

Rendering model for ray r(t) = o + td:

$$C \approx \sum_{i=1}^{N} T_i \alpha_i c_i \quad [2]$$

colors

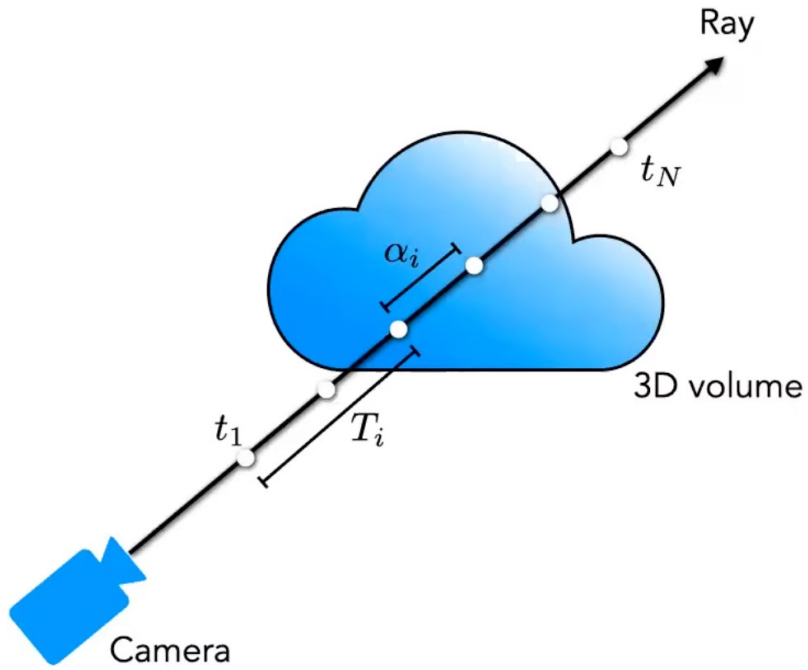
weights

How much light is blocked earlier along ray:

$$T_i = \prod_{j=1}^{i-1} (1 - \alpha_j)$$

How much light is contributed by ray segment i:

$$\alpha_i = 1 - e^{-\sigma_i \delta t_i}$$



[1] Ben Mildenhall, et al, Nerf: Representing scenes as neural radiance fields for view synthesis. In ECCV, 2020
[2] Max, N.: Optical models for direct volume rendering. In IEEE Transactions on Visualization and Computer Graphics, 1995

Introduction

Limitations of NeRF [1]

[1] Ben Mildenhall, et al, Nerf: Representing scenes as neural radiance fields for view synthesis. In ECCV, 2020

Introduction

Limitations of NeRF [1]

1. NeRF has to be optimized **per scene**
2. NeRF requires **millions** of network inferences

[1] Ben Mildenhall, et al, Nerf: Representing scenes as neural radiance fields for view synthesis. In ECCV, 2020

Introduction

Limitations of NeRF [1]

1. NeRF has to be optimized **per scene**
2. NeRF requires **millions** of network inferences

Advantages of MINE

1. MINE generalizes to **unseen scenes**
2. MINE requires lesser network inferences (e.g. 64)

[1] Ben Mildenhall, et al, Nerf: Representing scenes as neural radiance fields for view synthesis. In ECCV, 2020

Introduction

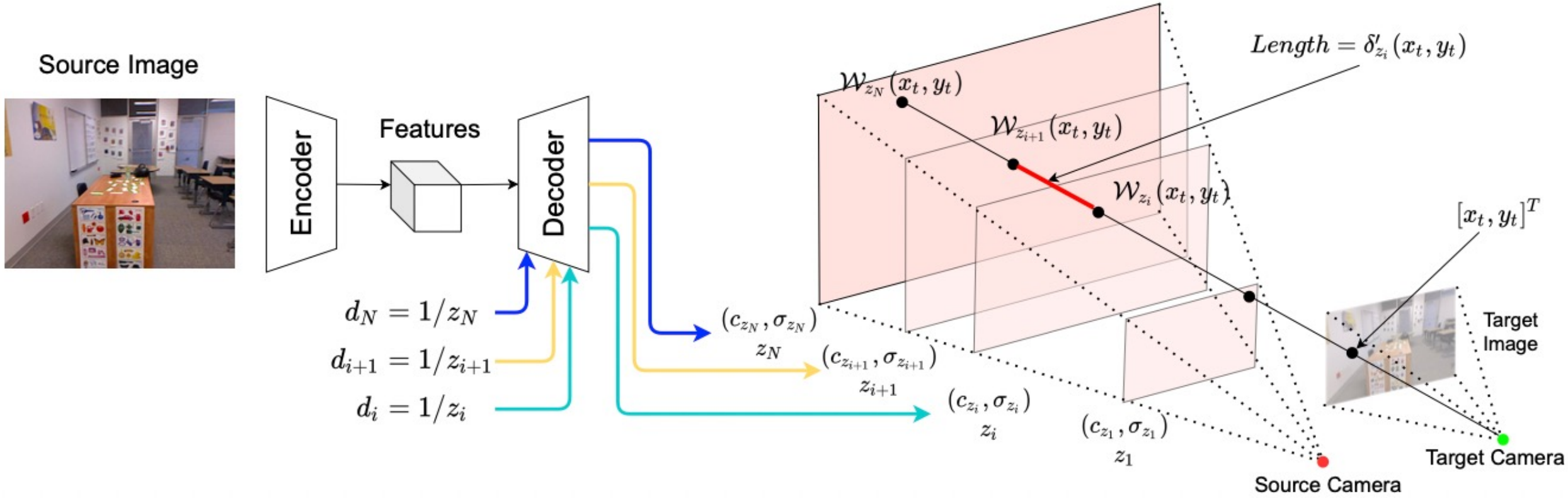
Method

Experiments

Conclusion

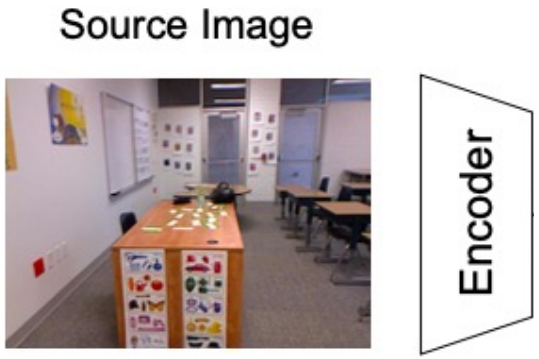
Method

- Network



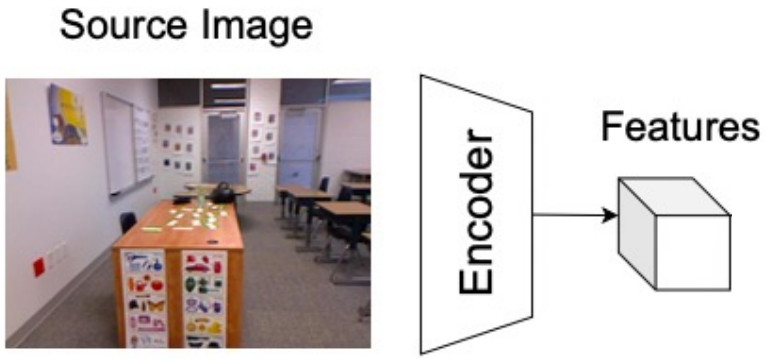
Method

- Network



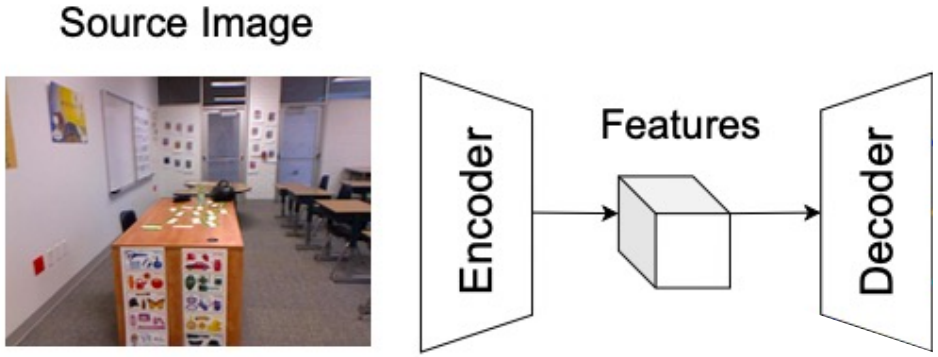
Method

- Network



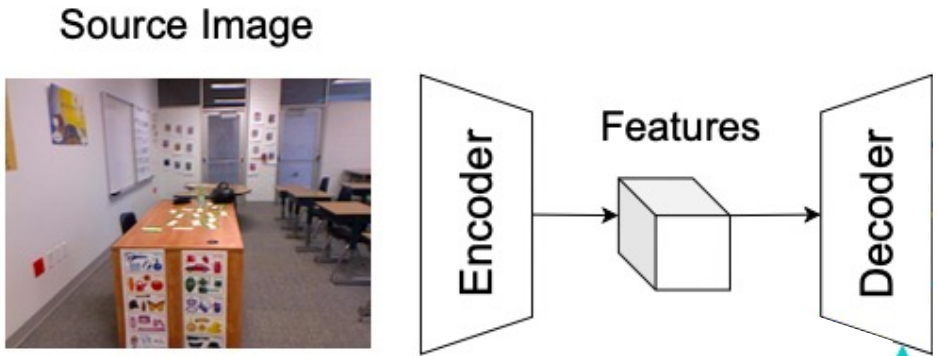
Method

- Network



Method

- Network

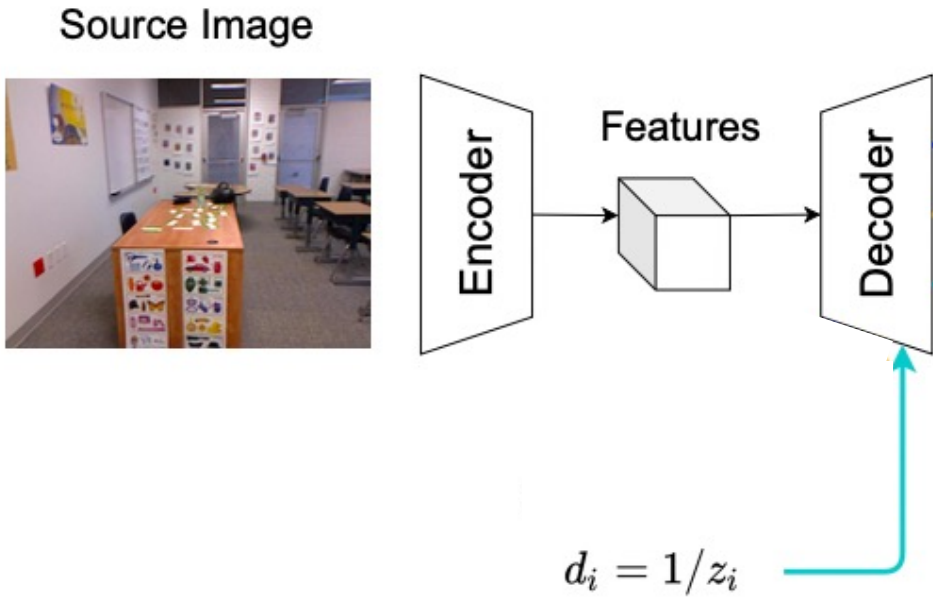


$$d_i = 1/z_i$$

$$d_i \sim \mathcal{U}\left[d_n + \frac{i}{N}(d_f - d_n), d_n + \frac{i-1}{N}(d_f - d_n)\right]$$

Method

- Network

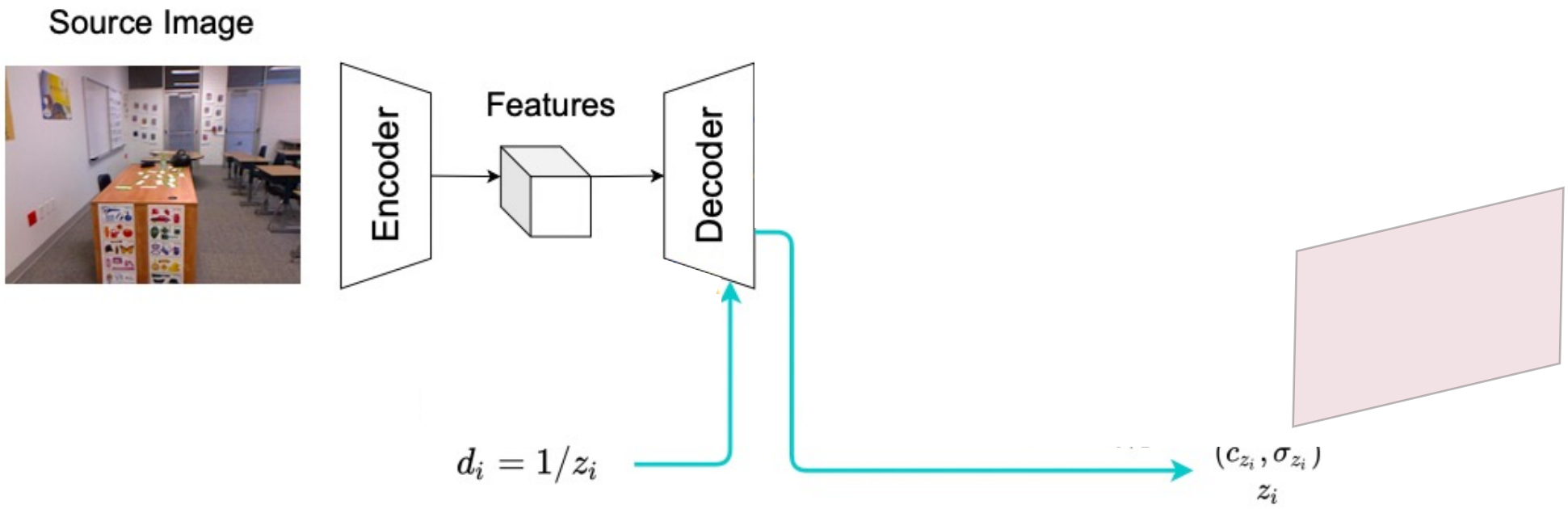


$$d_i \sim \mathcal{U}\left[d_n + \frac{i}{N}(d_f - d_n), d_n + \frac{i-1}{N}(d_f - d_n)\right]$$

$$\gamma(d_i) = [\sin(2^0 \pi d_i), \cos(2^0 \pi d_i), \cdots, \sin(2^{L-1} \pi d_i), \cos(2^{L-1} \pi d_i)]$$

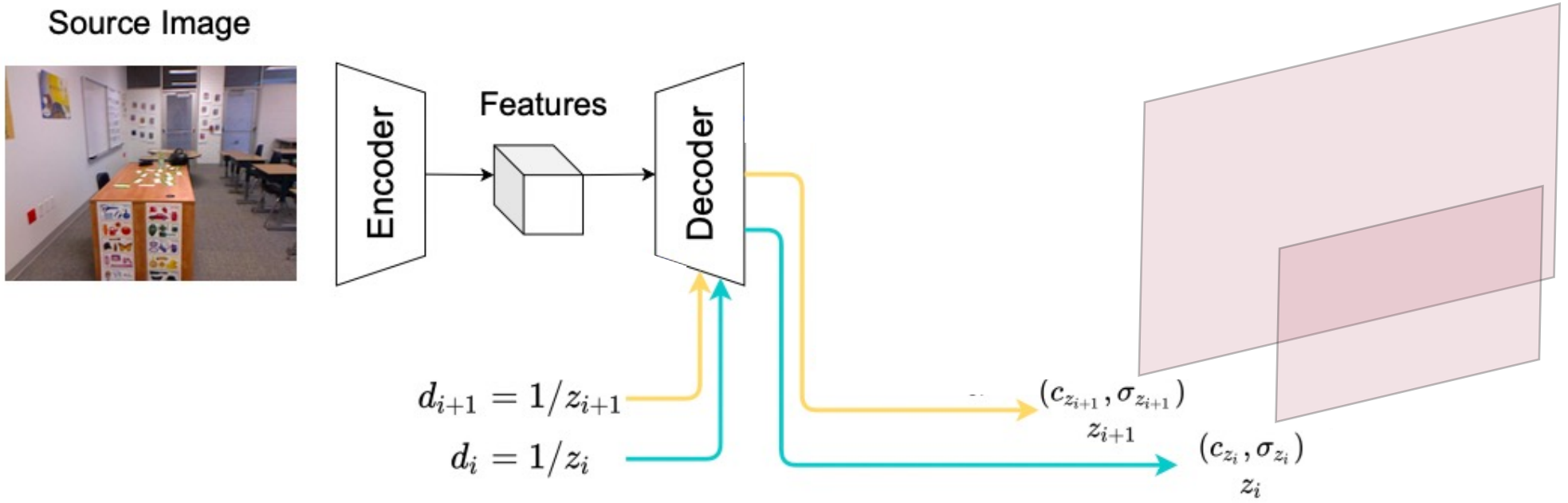
Method

- Network



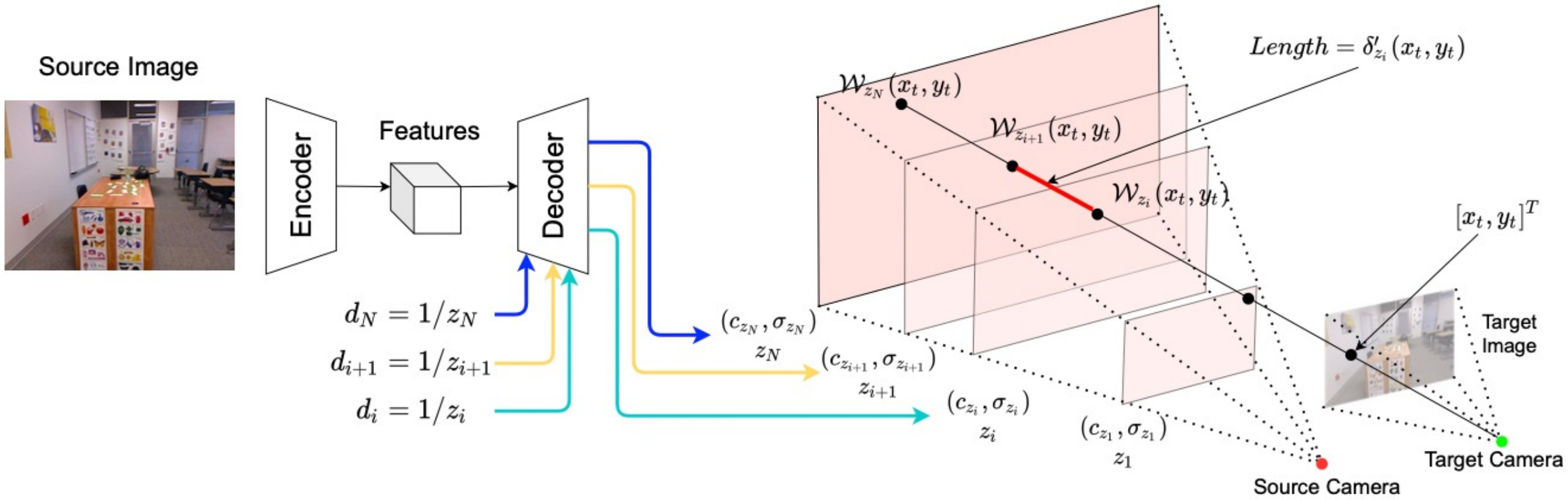
Method

- Network



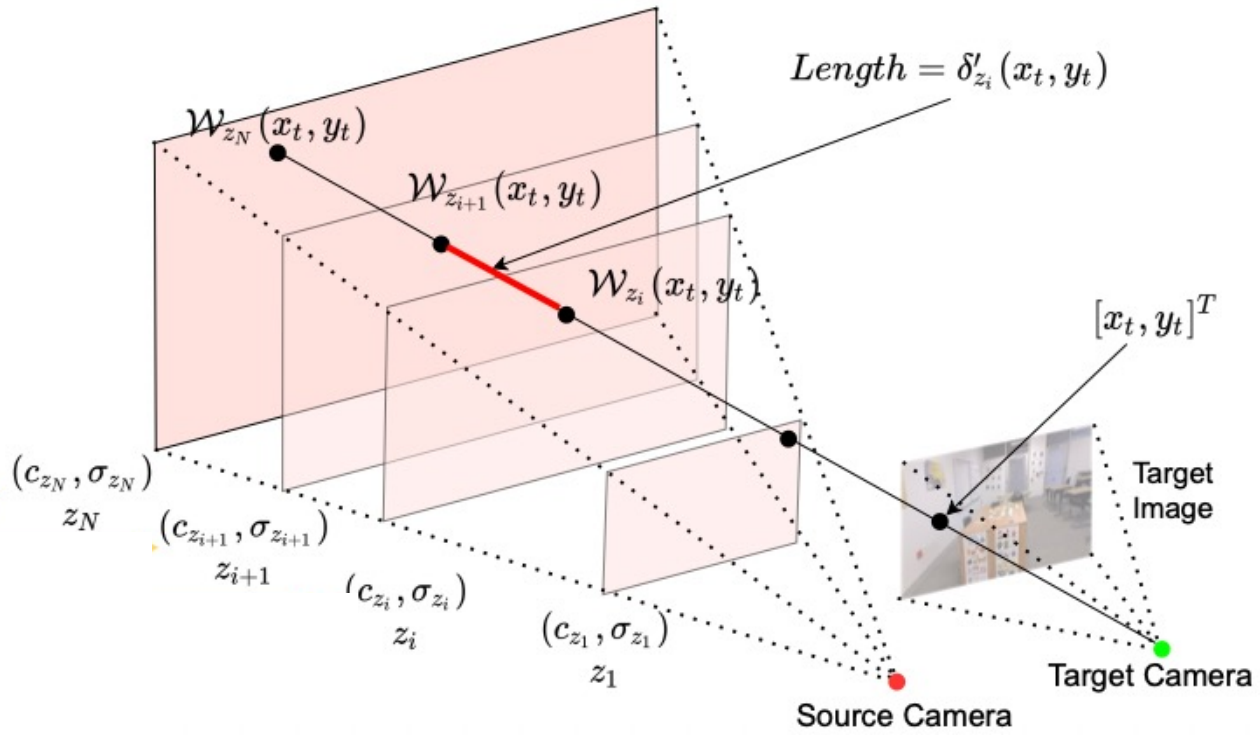
Method

- Network



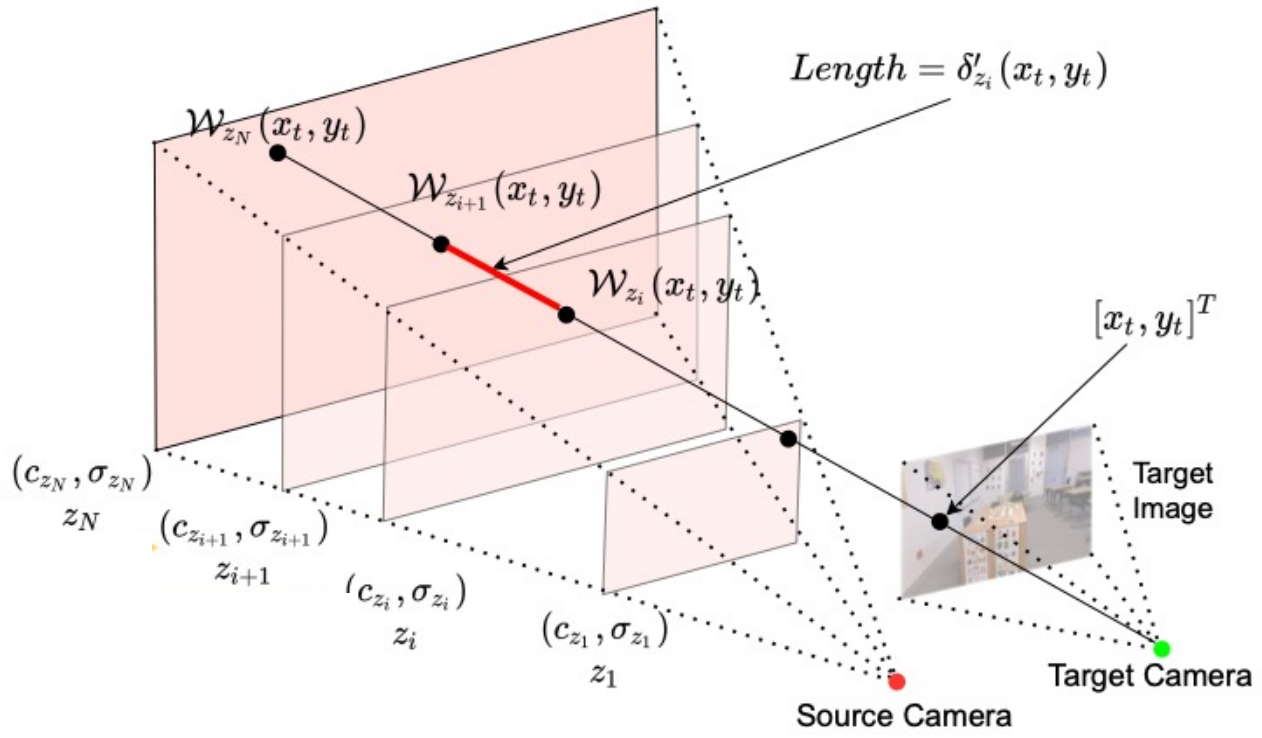
Method

- Rendering



Method

- Rendering

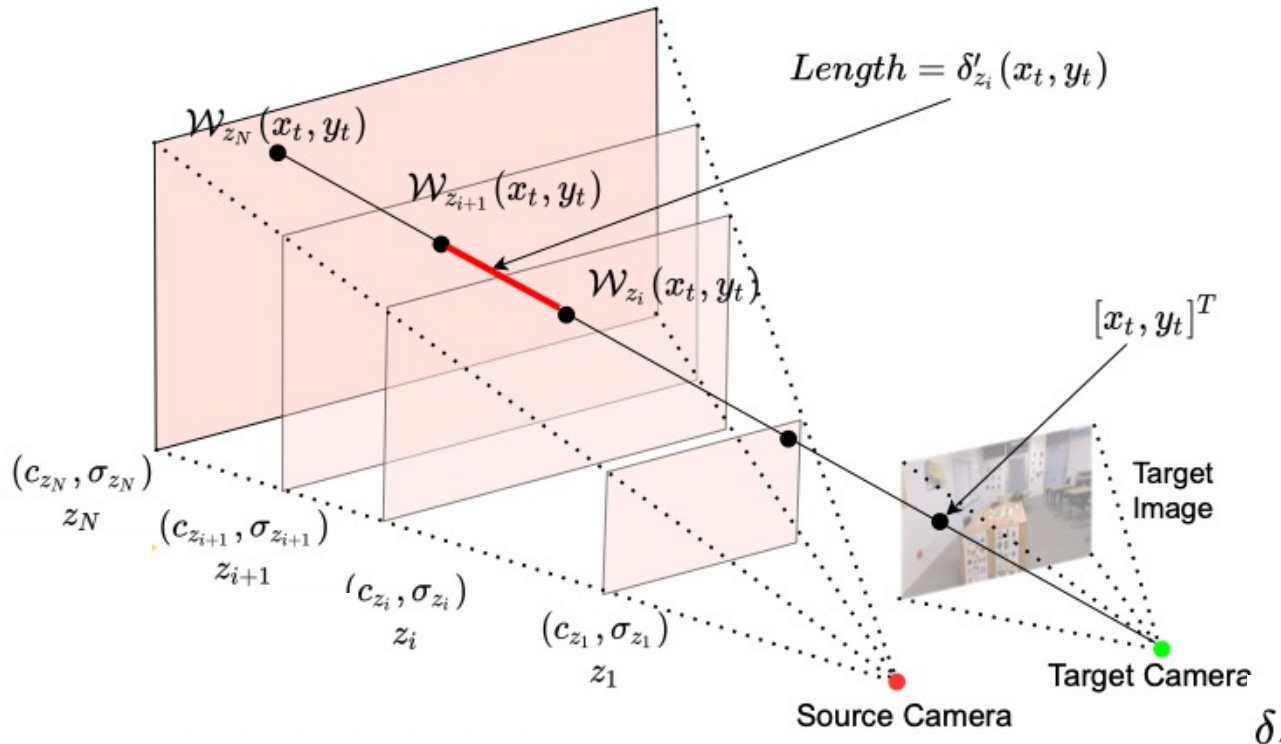


$$\hat{\mathrm{I}} = \sum_{i=1}^{N} T_i \big(1 - \exp(-\sigma_{z_i}\delta_{z_i})\big) c_{z_i}, \tag{1}$$

$$T_i = \exp\Big(- \sum_{j=1}^{i-1} \sigma_{z_j}\delta_{z_j} \Big) : \mathbb{R}^2 \to \mathbb{R}^+ \tag{2}$$

Method

- Rendering



$$\hat{\mathrm{I}} = \sum_{i=1}^{N} T_i \big(1 - \exp(-\sigma_{z_i}\delta_{z_i})\big) c_{z_i}, \tag{1}$$

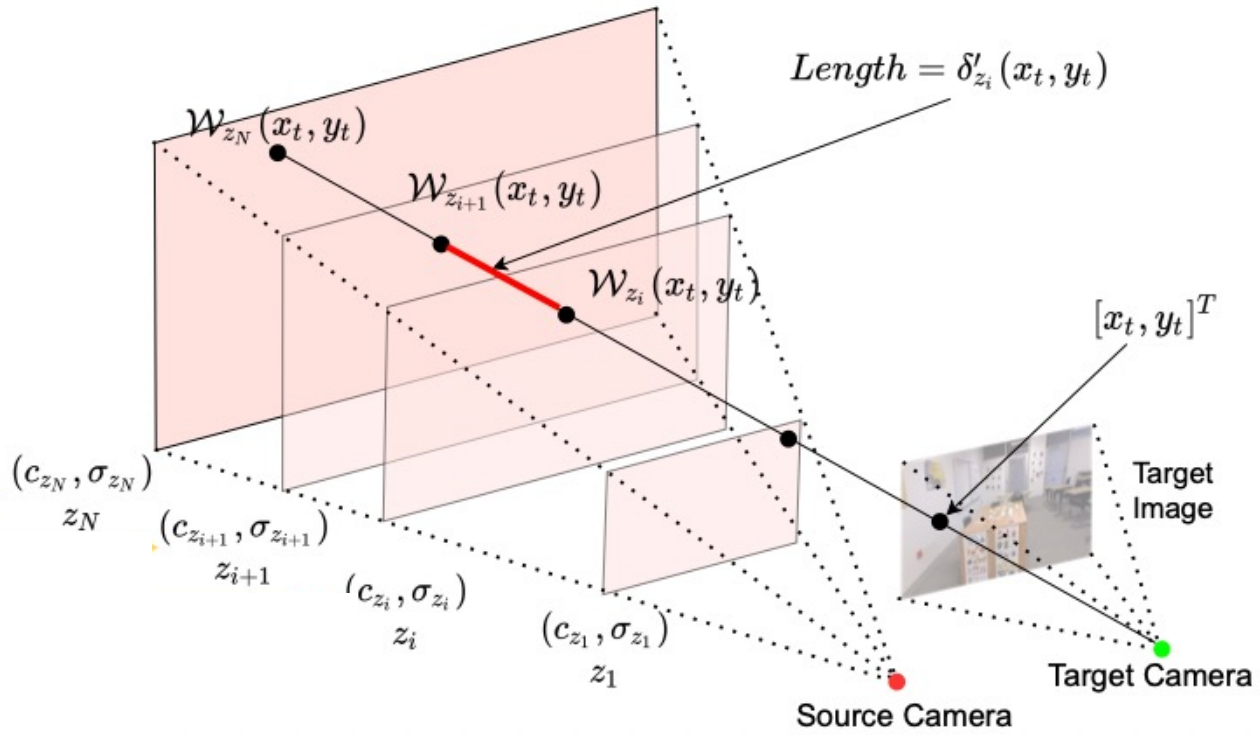
$$T_i = \exp\Big(- \sum_{j=1}^{i-1} \sigma_{z_j}\delta_{z_j} \Big) : \mathbb{R}^2 \to \mathbb{R}^+ \tag{2}$$

$$\mathfrak{C}\left(\begin{bmatrix} x \\ y \\ z \end{bmatrix}\right) = \mathrm{K}^{-1} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} z = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} zx \\ zy \\ z \end{bmatrix} \tag{3}$$

$$\delta_{z_i}(x, y) = \|\mathfrak{C}([x, y, z_{i+1}]^\top) - \mathfrak{C}([x, y, z_i]^\top)\|_2 : \mathbb{R}^2 \to \mathbb{R}^+ \tag{4}$$

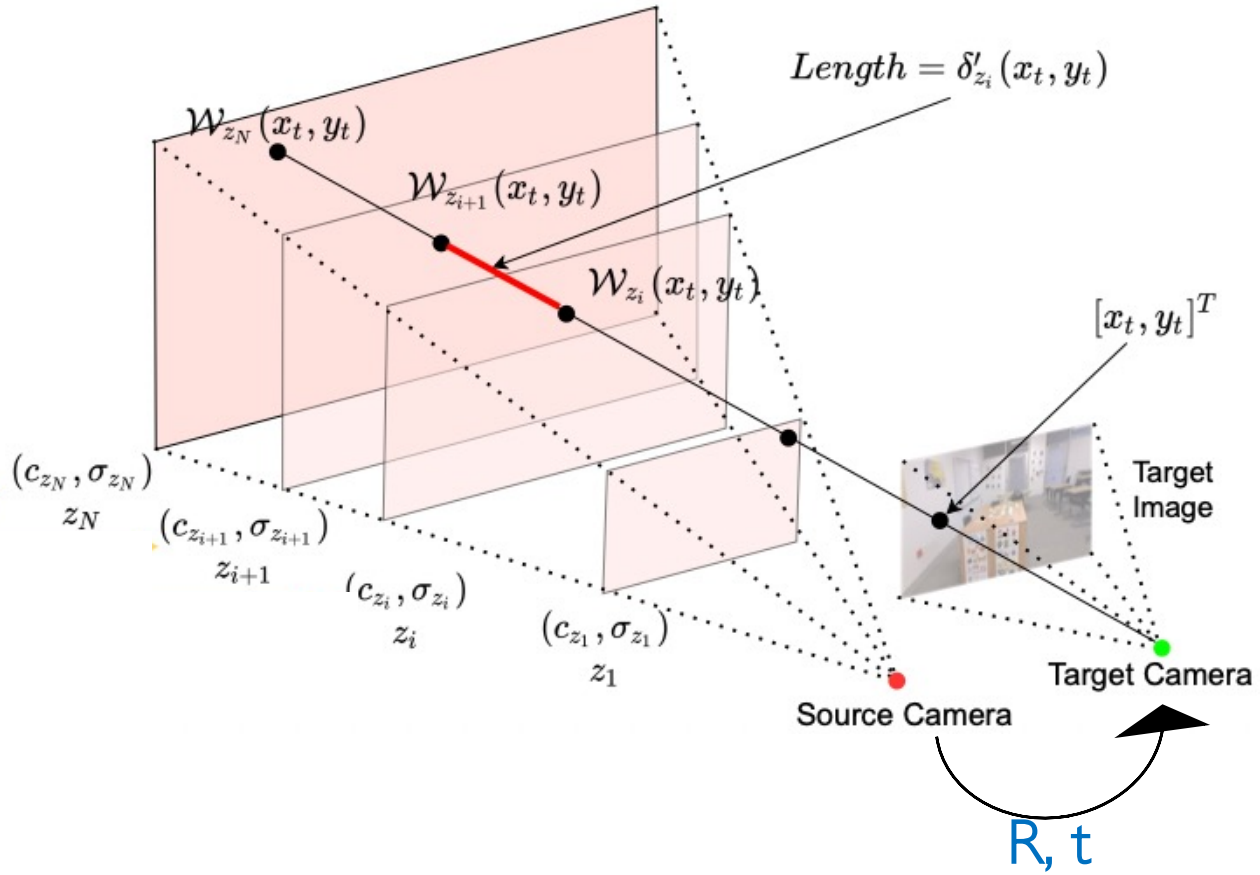
Method

- Rendering



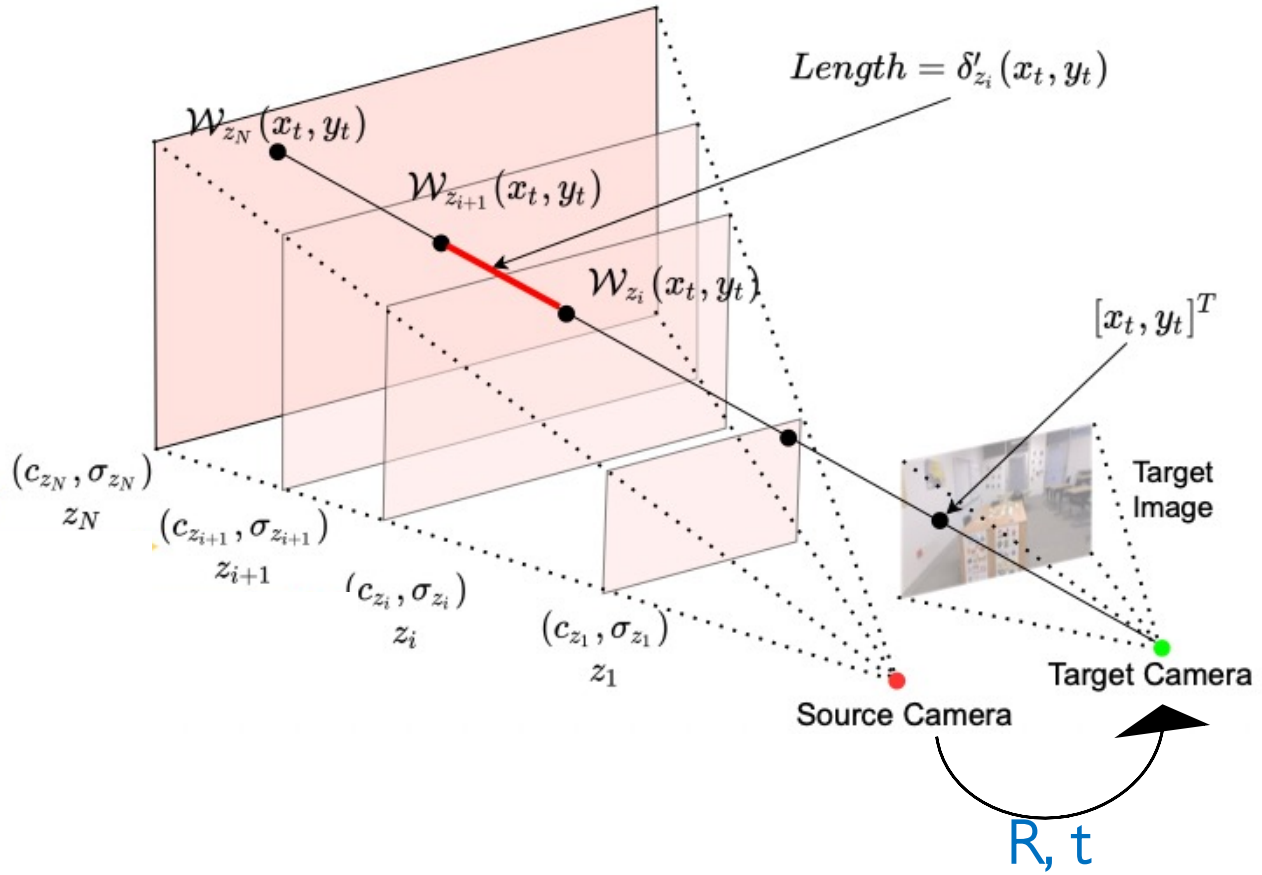
Method

- Rendering



Method

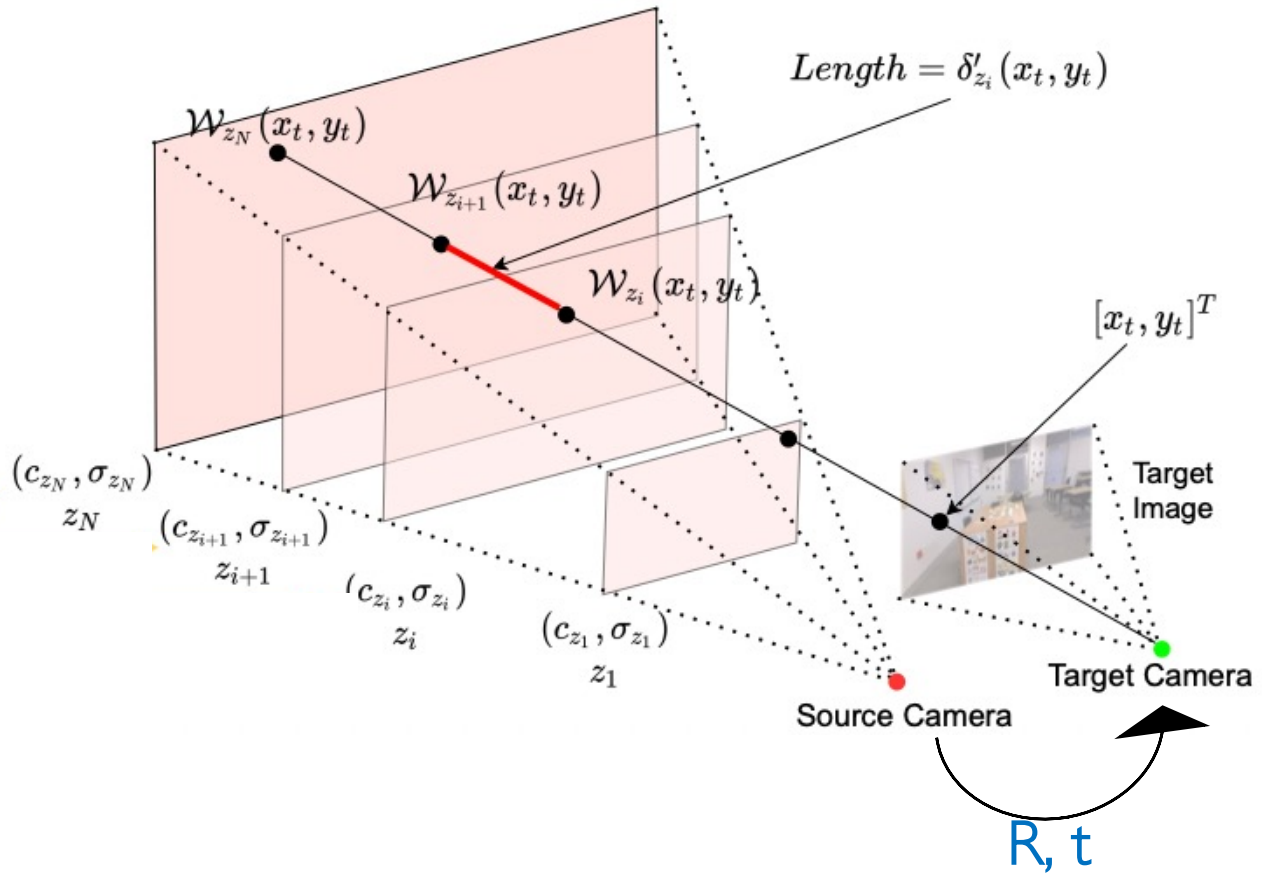
- Rendering



$$\left[x_s, y_s, 1\right]^\top \sim \mathbf{K}\left(\mathbf{R} - \frac{\mathbf{t}\mathbf{n}^\top}{z_i}\right)\mathbf{K}^{-1}\left[x_t, y_t, 1\right]^\top \quad (5)$$

Method

- Rendering

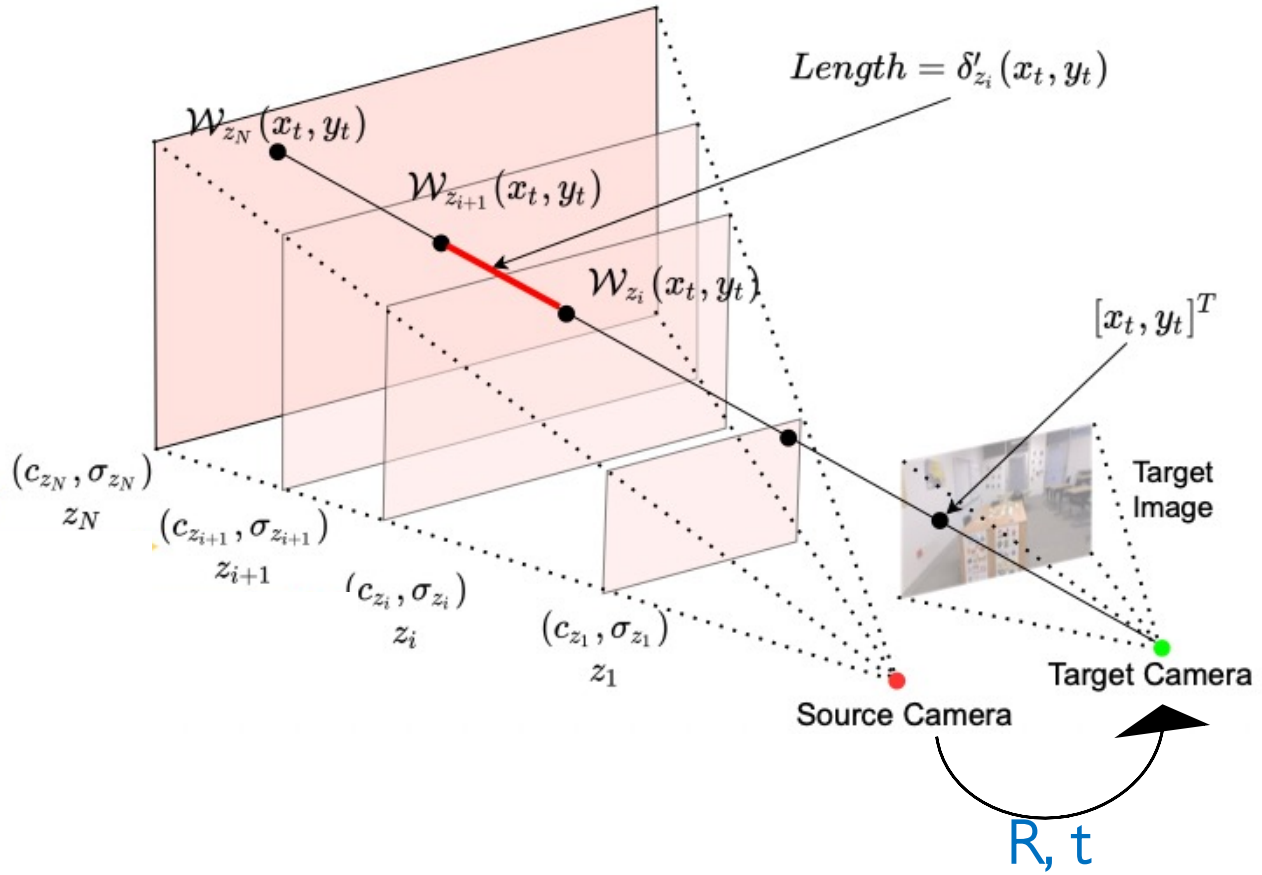


$$[x_s, y_s, 1]^\top \sim \mathbf{K}\left(\mathbf{R} - \frac{\mathbf{t}\mathbf{n}^\top}{z_i}\right)\mathbf{K}^{-1}[x_t, y_t, 1]^\top \quad (5)$$

$$\delta'_{z_i}(x_t, y_t) = \|\mathfrak{C}([\mathcal{W}_{z_{i+1}}(x_t, y_t), z_{i+1}]^\top) - \mathfrak{C}([\mathcal{W}_{z_i}(x_t, y_t), z_i]^\top)\|_2 \quad (6)$$

Method

- Rendering



$$[x_s, y_s, 1]^\top \sim \mathbf{K}\left(\mathbf{R} - \frac{\mathbf{t}\mathbf{n}^\top}{z_i}\right)\mathbf{K}^{-1}[x_t, y_t, 1]^\top \quad (5)$$

$$\delta'_{z_i}(x_t, y_t) = \|\mathfrak{C}([\mathcal{W}_{z_{i+1}}(x_t, y_t), z_{i+1}]^\top) - \mathfrak{C}([\mathcal{W}_{z_i}(x_t, y_t), z_i]^\top)\|_2 \quad (6)$$

$$\hat{\mathbf{I}} = \sum_{i=1}^{N} T_i\left(1 - \exp(-\sigma_{z_i}\delta_{z_i})\right)c_{z_i}, \quad (1)$$

Method

- Loss function

$$\mathcal{L} = \lambda_{\text{L1}}\mathcal{L}_{\text{L1}} + \lambda_{\text{ssim}}\mathcal{L}_{\text{ssim}} + \lambda_{\text{smooth}}\mathcal{L}_{\text{smooth}} + \lambda_{d}\mathcal{L}_{d}$$ (optional)

Method

- Loss function

$$\mathcal{L} = \lambda_{\text{L1}}\mathcal{L}_{\text{L1}} + \lambda_{\text{ssim}}\mathcal{L}_{\text{ssim}} + \lambda_{\text{smooth}}\mathcal{L}_{\text{smooth}} + \lambda_{d}\mathcal{L}_{d} \text{ (optional)}$$

$$\mathcal{L}_{\text{L1}} = \frac{1}{3HW}\sum |\hat{\mathrm{I}}_{tgt} - \mathrm{I}_{tgt}|, \quad \mathcal{L}_{\text{ssim}} = 1 - \mathrm{SSIM}(\hat{\mathrm{I}}_{tgt}, \mathrm{I}_{tgt})$$

Method

- Loss function

$$\mathcal{L} = \lambda_{\text{L1}}\mathcal{L}_{\text{L1}} + \lambda_{\text{ssim}}\mathcal{L}_{\text{ssim}} + \lambda_{\text{smooth}}\mathcal{L}_{\text{smooth}} + \lambda_d\mathcal{L}_d \text{ (optional)}$$

$$\mathcal{L}_{\text{L1}} = \frac{1}{3HW}\sum |\hat{\mathrm{I}}_{tgt} - \mathrm{I}_{tgt}|, \quad \mathcal{L}_{\text{ssim}} = 1 - \text{SSIM}(\hat{\mathrm{I}}_{tgt}, \mathrm{I}_{tgt})$$

$$\mathcal{L}_{smooth} = |\partial_x \hat{\mathcal{D}}^*| \exp^{-|\partial_x \mathrm{I}|} + |\partial_y \hat{\mathcal{D}}^*| \exp^{-|\partial_y \mathrm{I}|}$$

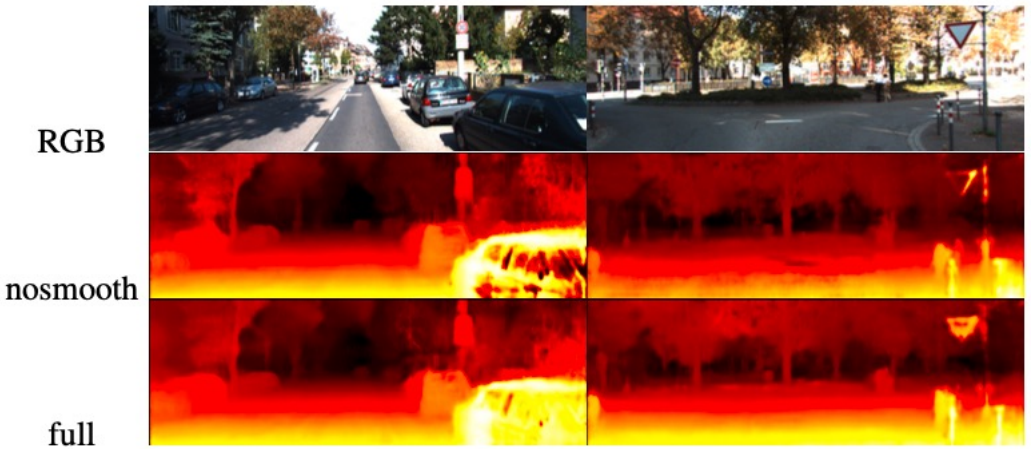
Method

- Loss function

$$\mathcal{L} = \lambda_{\text{L1}}\mathcal{L}_{\text{L1}} + \lambda_{\text{ssim}}\mathcal{L}_{\text{ssim}} + \lambda_{\text{smooth}}\mathcal{L}_{\text{smooth}} + \lambda_{d}\mathcal{L}_{d} \text{ (optional)}$$

$$\mathcal{L}_{\text{L1}} = \frac{1}{3HW}\sum |\hat{\text{I}}_{tgt} - \text{I}_{tgt}|, \quad \mathcal{L}_{\text{ssim}} = 1 - \text{SSIM}(\hat{\text{I}}_{tgt}, \text{I}_{tgt})$$

$$\mathcal{L}_{smooth} = |\partial_x \hat{\mathcal{D}}^*| \exp^{-|\partial_x \text{I}|} + |\partial_y \hat{\mathcal{D}}^*| \exp^{-|\partial_y \text{I}|}$$



Method

- Loss function

$$\mathcal{L} = \lambda_{\text{L1}}\mathcal{L}_{\text{L1}} + \lambda_{\text{ssim}}\mathcal{L}_{\text{ssim}} + \lambda_{\text{smooth}}\mathcal{L}_{\text{smooth}} + \lambda_d\mathcal{L}_d \text{ (optional)}$$

$$\mathcal{L}_{\text{L1}} = \frac{1}{3HW}\sum |\hat{\mathrm{I}}_{tgt} - \mathrm{I}_{tgt}|, \quad \mathcal{L}_{\text{ssim}} = 1 - \text{SSIM}(\hat{\mathrm{I}}_{tgt}, \mathrm{I}_{tgt})$$

$$\mathcal{L}_{smooth} = |\partial_x \hat{\mathcal{D}}^*| \exp^{-|\partial_x \mathrm{I}|} + |\partial_y \hat{\mathcal{D}}^*| \exp^{-|\partial_y \mathrm{I}|}$$

$$\mathcal{L}_d = 0.5\mathcal{L}_d^{src} + 0.5\mathcal{L}_d^{tgt}, \quad \text{where}$$

$$\mathcal{L}_d^{src} = \frac{1}{|\mathbf{P_s}|}\sum_{(x,y,z)\in\mathbf{P_s}} \left(\ln \frac{\hat{\mathcal{D}}_{src}(x,y)}{s} - \ln \frac{1}{z} \right),$$

$$\mathcal{L}_d^{tgt} = \frac{1}{|\mathbf{P_t}|}\sum_{(x,y,z)\in\mathbf{P_t}} \left(\ln \frac{\hat{\mathcal{D}}_{tgt}(x,y)}{s} - \ln \frac{1}{z} \right).$$

Method

- Loss function

$$\mathcal{L} = \lambda_{\text{L1}}\mathcal{L}_{\text{L1}} + \lambda_{\text{ssim}}\mathcal{L}_{\text{ssim}} + \lambda_{\text{smooth}}\mathcal{L}_{\text{smooth}} + \lambda_d\mathcal{L}_d \text{ (optional)}$$

$$\mathcal{L}_{\text{L1}} = \frac{1}{3HW}\sum |\hat{\mathrm{I}}_{tgt} - \mathrm{I}_{tgt}|, \quad \mathcal{L}_{\text{ssim}} = 1 - \text{SSIM}(\hat{\mathrm{I}}_{tgt}, \mathrm{I}_{tgt})$$

$$\mathcal{L}_{smooth} = |\partial_x \hat{\mathcal{D}}^*| \exp^{-|\partial_x \mathrm{I}|} + |\partial_y \hat{\mathcal{D}}^*| \exp^{-|\partial_y \mathrm{I}|}$$

$$\mathcal{L}_d = 0.5\mathcal{L}_d^{src} + 0.5\mathcal{L}_d^{tgt}, \quad \text{where}$$

$$\mathcal{L}_d^{src} = \frac{1}{|\mathbf{P_s}|}\sum_{(x,y,z)\in\mathbf{P_s}} \left(\ln \frac{\hat{\mathcal{D}}_{src}(x,y)}{s} - \ln \frac{1}{z} \right), \qquad s = \exp\left[\frac{1}{|\mathbf{P_s}|}\sum_{(x,y,z)\in\mathbf{P_s}} \left(\ln(\hat{\mathcal{Z}}_{src}(x,y) - \ln z) \right] \right.$$

$$\mathcal{L}_d^{tgt} = \frac{1}{|\mathbf{P_t}|}\sum_{(x,y,z)\in\mathbf{P_t}} \left(\ln \frac{\hat{\mathcal{D}}_{tgt}(x,y)}{s} - \ln \frac{1}{z} \right).$$

Introduction

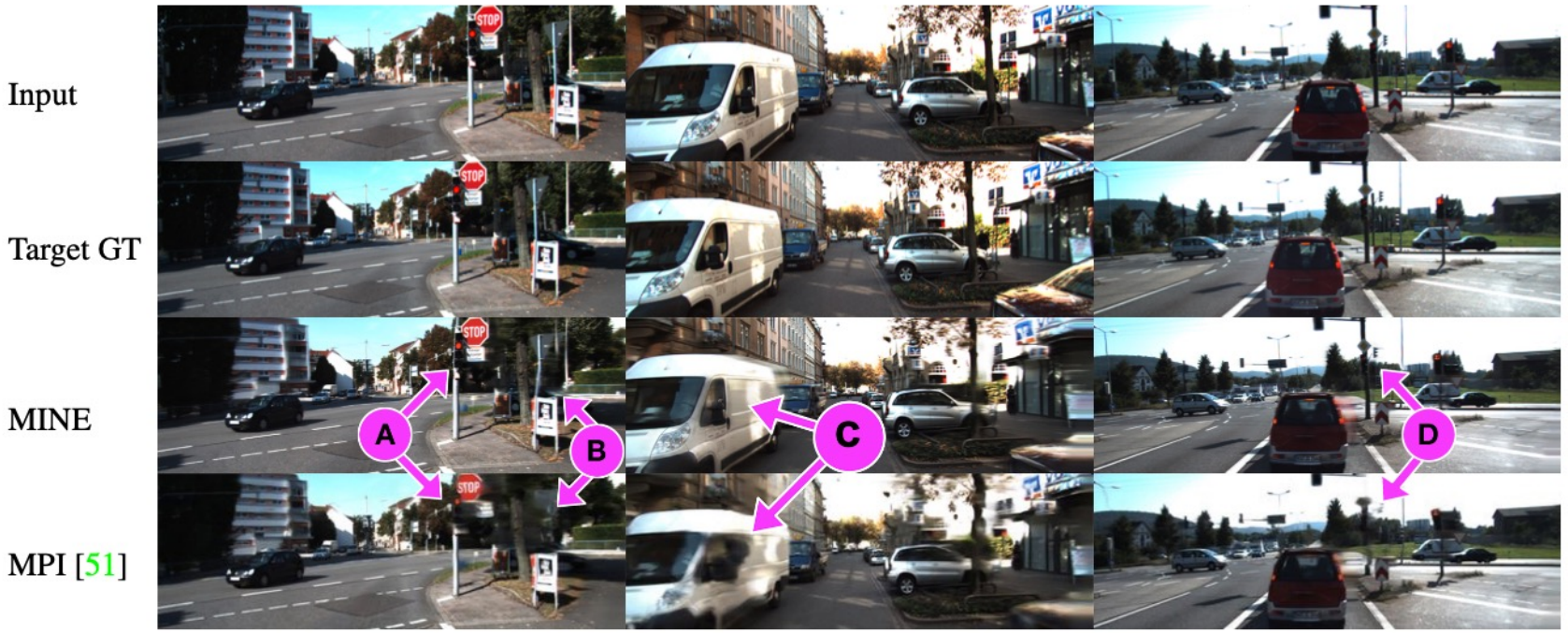
Method

Experiments

Conclusion

Experiments

Experiments on KITTI



	Train Res.	N	Pre-trained	Depth Smoothess	LPIPS↓	SSIM↑	PSNR↑
MINE	384x128	32	N	Y	0.129	0.812	21.4
MINE	384x128	32	Y	N	0.123	0.816	21.6
MINE	384x128	32	Y	Y	0.122	0.815	21.6
MINE	384x128	64	Y	Y	0.117	0.818	21.6
MINE	384x128	256	Y	Y	**0.112**	**0.828**	**21.9**
Tulsiani et. al. [52]	768x256	NA	NA	NA	-	0.572	16.5
MPI [51]	768x256	32	NA	NA	-	0.733	19.5
MINE	768x256	32	Y	Y	0.112	**0.822**	21.4
MINE	768x256	64	Y	Y	**0.108**	0.820	21.3

Experiments

Experiments on RealEstate10K

Method	LPIPS↓			SSIM↑			PSNR↑		
	$n = 5$	$n = 10$	$n = random$	$n = 5$	$n = 10$	$n = random$	$n = 5$	$n = 10$	$n = random$
SynSin [56]	-	-	-	-	-	0.74	-	-	22.31
MPI [51]	0.0967	0.1420	0.1761	0.8699	0.8124	0.7851	27.05	24.43	23.52
MINE ($N = 32$)	0.0934	0.1346	0.1674	0.8970	0.8464	0.8172	**28.51**	**25.73**	**24.56**
MINE ($N = 64$)	**0.0896**	**0.1280**	**0.1562**	**0.8974**	**0.8500**	**0.8219**	28.39	25.71	24.50

Experiments

Experiments on Depth Estimation

Method	Supervision	Dataset	NYU-Depth V2 [30]						iBims-1 [21]					
			rel↓	log10↓	RMS↓	$\sigma 1$↑	$\sigma 2$↑	$\sigma 3$↑	rel↓	log10↓	RMS↓	$\sigma 1$↑	$\sigma 2$↑	$\sigma 3$↑
DIW [2]	Depth	DIW	0.25	0.1	0.76	0.62	0.88	0.96	0.25	0.1	1	0.61	0.86	0.95
DIW [2]	Depth	DIW+NYU	0.19	0.08	0.6	0.73	0.93	0.98	0.19	0.08	0.8	0.72	0.91	0.97
MegaDepth [24]	Depth	Mega	0.24	0.09	0.72	0.63	0.88	0.96	0.23	0.09	0.83	0.67	0.89	0.96
MegaDepth [24]	Depth	Mega+DIW	0.21	0.08	0.65	0.68	0.91	0.97	0.2	0.08	0.78	0.7	0.91	0.97
3DKenBurns [32]	Depth	Mega+NYU+3DKenBurn	**0.08**	**0.03**	**0.3**	**0.94**	**0.99**	**1**	**0.1**	**0.04**	**0.47**	**0.9**	**0.97**	**0.99**
MiDaS v2.1 [37]	Depth	MiDaS 10 datasets	0.16	0.06	0.50	0.80	0.95	0.99	0.14	0.06	0.57	0.84	**0.97**	**0.99**
MPI [51]	RGB	RealEstate10K	0.15	0.06	0.49	0.81	0.96	0.99	0.21	0.08	0.85	0.7	0.91	0.97
MINE ($N = 64$)	RGB	RealEstate10K	0.11	0.05	0.40	0.88	0.98	0.99	0.11	0.05	0.53	0.87	**0.97**	**0.99**

Experiments

Experiments on Depth Estimation

Method	Supervision	Dataset	NYU-Depth V2 [30]						iBims-1 [21]					
			rel↓	log10↓	RMS↓	$\sigma 1$↑	$\sigma 2$↑	$\sigma 3$↑	rel↓	log10↓	RMS↓	$\sigma 1$↑	$\sigma 2$↑	$\sigma 3$↑
DIW [2]	Depth	DIW	0.25	0.1	0.76	0.62	0.88	0.96	0.25	0.1	1	0.61	0.86	0.95
DIW [2]	Depth	DIW+NYU	0.19	0.08	0.6	0.73	0.93	0.98	0.19	0.08	0.8	0.72	0.91	0.97
MegaDepth [24]	Depth	Mega	0.24	0.09	0.72	0.63	0.88	0.96	0.23	0.09	0.83	0.67	0.89	0.96
MegaDepth [24]	Depth	Mega+DIW	0.21	0.08	0.65	0.68	0.91	0.97	0.2	0.08	0.78	0.7	0.91	0.97
3DKenBurns [32]	Depth	Mega+NYU+3DKenBurn	**0.08**	**0.03**	**0.3**	**0.94**	**0.99**	**1**	**0.1**	**0.04**	**0.47**	**0.9**	**0.97**	**0.99**
MiDaS v2.1 [37]	Depth	MiDaS 10 datasets	0.16	0.06	0.50	0.80	0.95	0.99	0.14	0.06	0.57	0.84	**0.97**	**0.99**
MPI [51]	RGB	RealEstate10K	0.15	0.06	0.49	0.81	0.96	0.99	0.21	0.08	0.85	0.7	0.91	0.97
MINE ($N = 64$)	RGB	RealEstate10K	0.11	0.05	0.40	0.88	0.98	0.99	0.11	0.05	0.53	0.87	**0.97**	**0.99**

- **RMSE** $= \sqrt{\frac{1}{|N|}\sum_{i \in N} \| d_i - d_i^* \|^2}$,
- **RMSE log** $= \sqrt{\frac{1}{|N|}\sum_{i \in N} \| \log(d_i) - \log(d_i^*) \|^2}$,
- **Abs Rel** $= \frac{1}{|N|}\sum_{i \in N} \frac{|d_i - d_i^*|}{d_i^*}$,
- **Sq Rel** $= \frac{1}{|N|}\sum_{i \in N} \frac{\|d_i - d_i^*\|^2}{d_i^*}$,
- **Accuracies:** % of d_i s.t. $\max(\frac{d_i}{d_i^*}, \frac{d_i^*}{d_i}) = \delta < thr$,

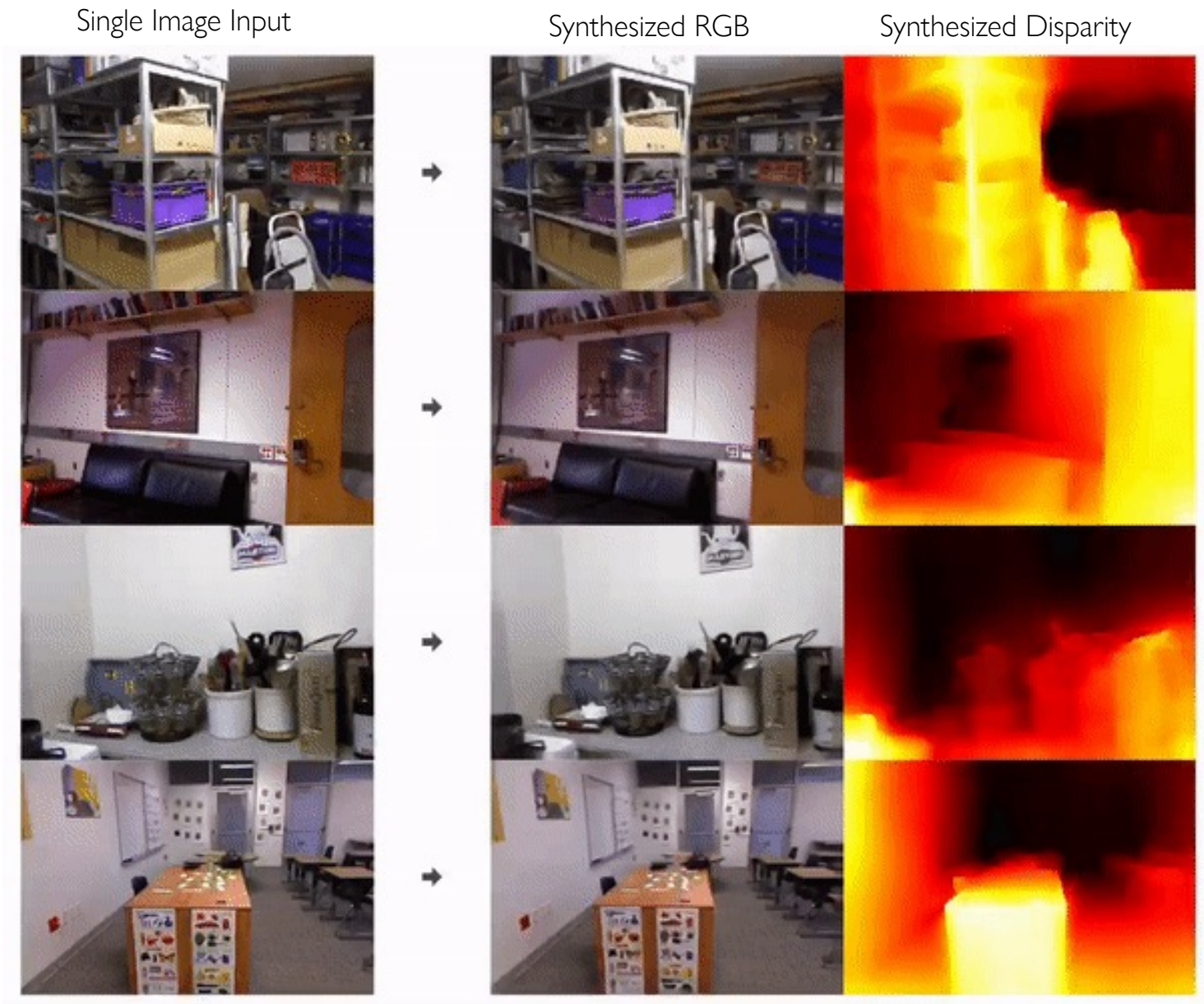
Introduction

Method

Experiments

Conclusion

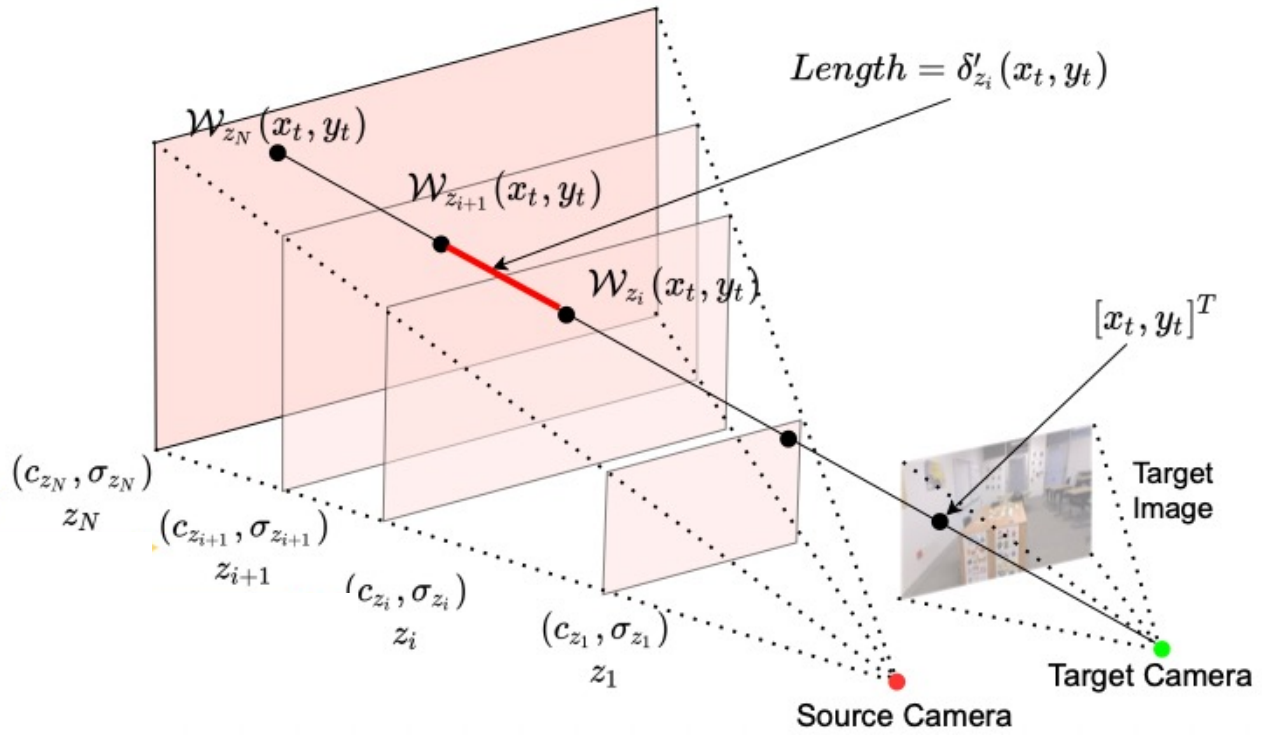
Conclusion


Single Image Input
Synthesized RGB
Synthesized Disparity

Q & A

Method

- Rendering



$$\hat{\mathrm{I}} = \sum_{i=1}^{N} T_i \big(1 - \exp(-\sigma_{z_i}\delta_{z_i})\big) c_{z_i}, \tag{1}$$

$$T_i = \exp\Big(- \sum_{j=1}^{i-1} \sigma_{z_j}\delta_{z_j} \Big) : \mathbb{R}^2 \to \mathbb{R}^+ \tag{2}$$

$$\hat{\mathcal{Z}} = \sum_{i=1}^{N} T_i \big(1 - \exp(-\sigma_{z_i}\delta_{z_i})\big) z_i \tag{7}$$