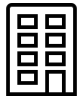
NeRF with Camera Pose Estimation



Sogang University
Vision & Display Systems Lab, Dept. of Electronic Engineering

Presented By
나송주

Outline

- Introduction
  - Background
    - Scenario of novel view synthesis
    - Volume rendering
  - NeRF
  - Camera parameters

- NeRF with Camera Pose Estimation
  - Nope-NeRF (CVPR 2023)
  - SPARF (CVPR 2023)

Introduction

- Background
  - Scenario of novel view synthesis

– N개의 시점에서 촬영된 2D image 학습 → 임의의 시점에서 촬영된 2D image 추론

  - Volume rendering

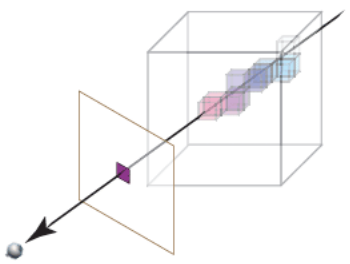
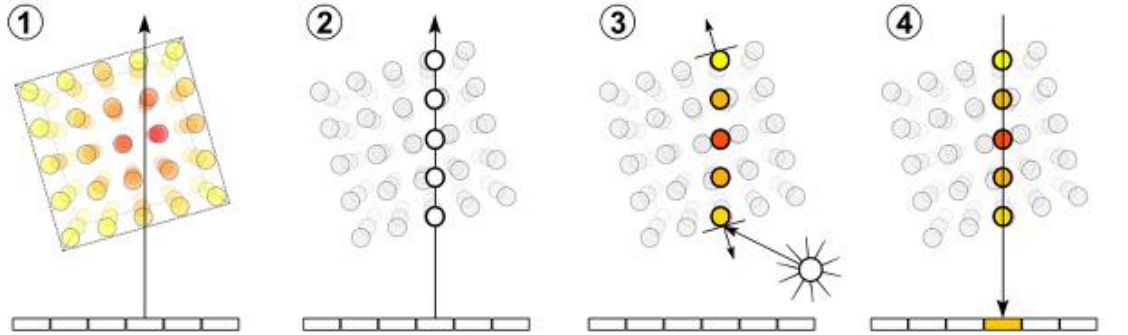


Figure 2: *The ray casting integral sums the color and opacity properties of each data voxel that intersects the ray.*

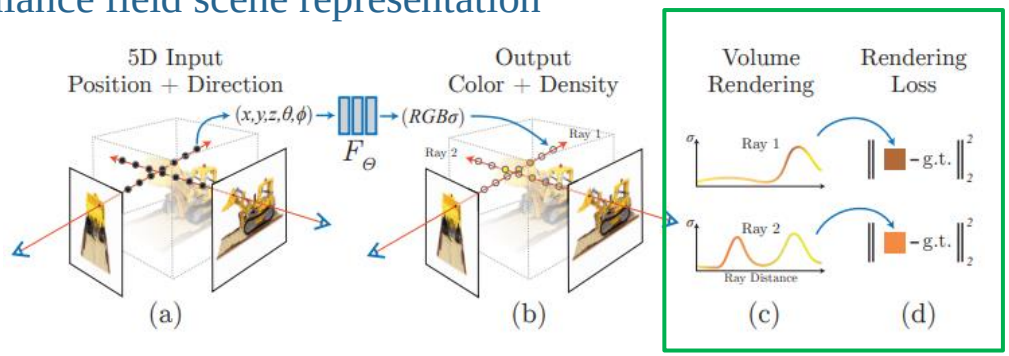


– 3차원 scholar field 형태의 discrete한 데이터를 2차원 image로 투영하는 기술

– 투영할 image plane의 각 픽셀에서 객체를 향해 ray를 발사하고, ray가 지나는 모든 voxel의 색상을 합쳐서 target pixel의 색상을 결정

Introduction

- NeRF[1)]
  - Neural radiance field scene representation



추정한 target pixel color와 GT color 사이의 L2 loss를 사용하여 학습됨

- 3D point (x, y, z)와 viewing direction (θ, ϕ)로 구성된 5D input을 입력 받아 해당 point의 density σ와 view-dependent RGB color c를 출력하는 MLP를 학습

투명도의 역수 개념으로,
그 위치의 particle의 density를 나타냄

  - MLP structure

**Positional encoding
5D input을 그대로 입력하면 high-frequency 성분을 잘 표현하지 못함
→higher dimension space로 mapping하여 network에 입력

$$\gamma(p) = \left(\sin(2^0\pi p), \cos(2^0\pi p), \cdots, \sin(2^{L-1}\pi p), \cos(2^{L-1}\pi p)\right)$$

- Density는 viewing direction에 상관없이 일정한 값을 가지므로 view들 간의 consistency를 유지하기 위해 position만을 사용하여 density를 추정함
- 이후 viewing direction을 입력 받아 view-dependent한 color를 추정

Introduction

- NeRF[1)]
  - Volume rendering
    - MLP가 추정한 각 point의 color를 density를 반영한 weighted sum으로 더해서 target pixel color를 계산
      - 1. Density가 클수록 weight가 커야 함
      - 2. 어떤 point를 가로막고 있는 point들의 density 합이 작을수록 그 point의 weight가 커야 함
    - 위의 2가지 특성을 반영하여 다음과 같은 식을 통해 rendering을 수행

카메라 위치 o와 viewing direction d이 정해지면 공간 상의 ray r과 point t는 다음과 같은 관계로 표현됨
$r(t) = o + td$

Density가 클수록 가중치가 커짐
→위의 1번 특성 반영

$$\hat{c}(r) = \int_0^{\infty} e^{-\int_0^t \sigma(r(\tau))d\tau} \sigma(r(t))c(r(t), d)dt$$

Transmittance, t보다 앞에 있는 point들의 density의 합의 역수 개념
→ 위의 2번 특성 반영

- 이를 계산하기 위해 ray의 시작점 t_n부터 끝점 t_f를 N 등분하고 각 segment에서 random point를 sample

$$t_i \sim U[t_n + \frac{i-1}{N}(t_f - t_n), t_n + \frac{i}{N}(t_f - t_n)]$$

$$\hat{c}(r) = \sum_{i=1}^{N} T_i(1 - \exp(-\sigma_i \delta_i))c_i, \text{where } T_i = \exp(-\sum_{j=1}^{i-1} \sigma_j \delta_j)$$ $\delta_i = \|t_{i+1} - t_i\|$, interval length를 나타냄

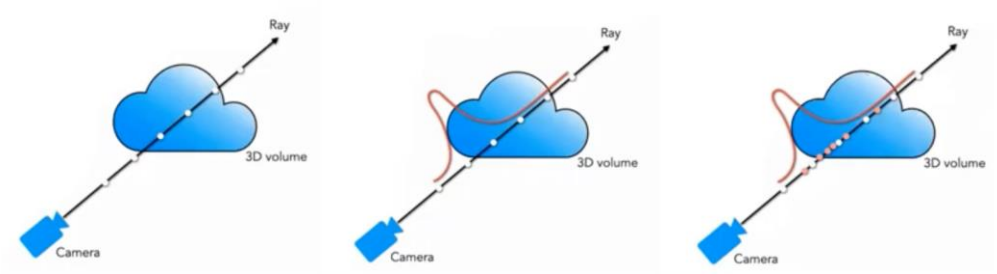
Color를 빼고 rendering하면, depth map을 얻을 수 있음

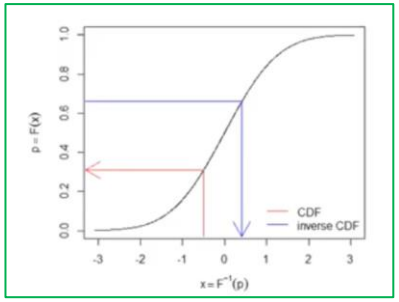
Introduction

- NeRF[1)]
  - Hierarchical volume sampling
    - Rendering 결과에 관여하지 않는 pixel에서 query하는 것을 방지하기 위해 coarse network와 fine network를 함께 optimize하는 학습 방식을 사용함
    - Coarse network는 ray의 시작점 t_n부터 끝점 t_f를 N_c 등분하고 각 segment에서 random point를 sample해서 rendering weight $w_i = T_i(1 - \exp(-\sigma_i \delta_i))$를 계산함
    - 여기서 w_i를 normalizing하면 ($\hat{w}_i = w_i / \sum_{j=1}^{N_c} w_j$) ray 상의 weight에 대한 PDF로 볼 수 있음
    - PDF를 통해 CDF를 계산하고, inverse sampling을 통해 N_f개의 point를 sample하고 fine network는 $N_c + N_f$개의 sample을 사용하여 학습됨 → w_i가 큰 부분에 가중치를 두고 sampling하게 됨
    - 최종적으로 다음과 같은 loss function을 사용하여 coarse / fine network를 학습시킴

$$L = \sum_{r \in R} [\| \hat{C}_c(r) - C(r) \|_2^2 + \| \hat{C}_f(r) - C(r) \|_2^2]$$

    - 여기서 R은 ray들의 집합이고, $C(r)$은 GT target pixel color임

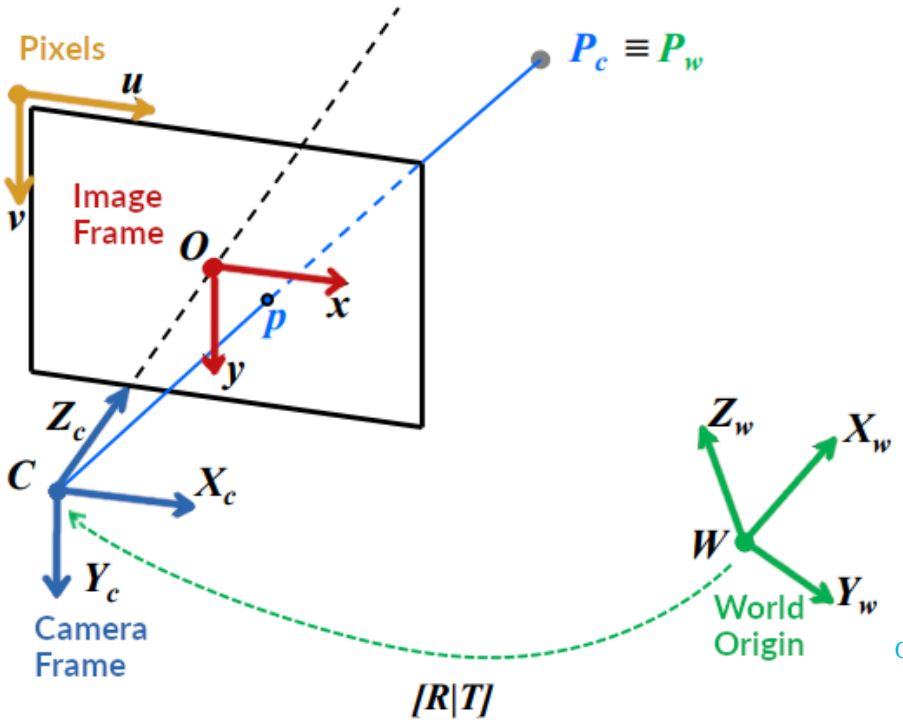




<Inverse sampling의 개념도>

Introduction

- Camera Parameters
  - Commonly used coordinate systems and transforms



$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = \begin{bmatrix} R & T \end{bmatrix} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

Extrinsic parameter

$$\begin{bmatrix} \lambda u \\ \lambda v \\ \lambda \end{bmatrix} = \begin{bmatrix} k_u f & 0 & u_0 \\ 0 & k_v f & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}$$

Intrinsic parameter
(=Calibration matrix),
Camera의 렌즈와 센서 위치에 의해 결정되는 값으로,
같은 카메라를 사용하면 같은 값을 가짐

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}$$

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K \begin{bmatrix} R|T \end{bmatrix} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

In general, the camera parameters Π include camera intrinsics, poses, and lens distortions. We only consider estimating camera poses in this work, *e.g.*, camera pose for frame I_i is a transformation $\mathbf{T}_i = [\mathbf{R}_i \mid \mathbf{t}_i]$ with a rotation $\mathbf{R}_i \in \mathrm{SO}(3)$ and a translation $\mathbf{t}_i \in \mathbb{R}^3$.

Camera pose estimation은 extrinsic parameter를 추정하는 것이 목표
(Intrinsic parameter는 알려져 있다고 가정)

NoPe-NeRF: Optimising Neural Radiance Field with No Pose Prior

NoPe-NeRF[1)]

- Abstract
  - NeRF와 camera pose를 함께 최적화함으로써 pose prior없이 3D scene을 표현하는 방법 제안
    - NeRF를 학습 시키기 위해선 train image들의 camera parameter 계산이 필요
      - COLMAP을 사용하여 계산되며, 긴 시간이 소요되어 실용성을 저하시키는 요인으로 작용
    - 기존 camera pose와 NeRF를 함께 최적화하는 방법은 camera motion이 큰 경우 성능이 저하됨
    - 본 논문은 monocular depth prior를 NeRF에 결합한 학습 방법을 사용하여 기존 방법의 문제점을 해결
      - 기존 방법은 camera pose를 독립적으로 학습하여 pose 추정의 정확도가 낮음
      - 본 논문은 frame간의 relative pose loss를 사용하여 global한 pose estimation 결과를 얻음

<기존 방법과 novel view synthesis 및 pose 추정 성능 비교 >

NoPe-NeRF[1)]

- Method Overview
  - Joint optimization of poses and NeRF
    - 일반적인 NeRF는 neural network를 사용하여 scene을 다음과 같은 mapping function으로 표현함

$$F_\theta : (x, d) \rightarrow (c, \sigma)$$

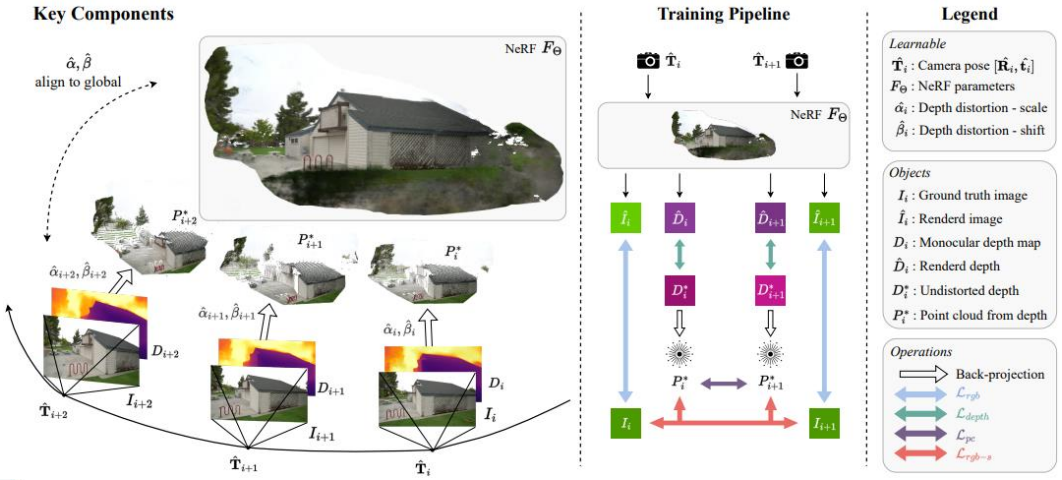
    - N개의 image $\mathcal{I} = \{I_i \mid i = 0 \dots N-1\}$와 그에 상응하는 camera pose $\Pi = \{\pi_i \mid i = 0 \dots N-1\}$에 대해 loss function $\mathcal{L}_{rgb} = \sum_i^N \left\| I_i - \hat{I}_i \right\|_2^2$를 최적화하여 prior parameter θ^*를 얻음

$$\theta^* = \arg_\theta \, min \, \mathcal{L}_{rgb}(\hat{\mathcal{I}} | \mathcal{I}, \Pi)$$

    - 본 논문은 camera parameter를 learnable한 변수 형태로 변환하여 다음과 같은 최적화 문제를 해결함

Camerapose도 NeRF와 함께 최적화

$$\theta^*, \Pi^* = \arg_{\theta,\Pi} \, min \, \mathcal{L}_{rgb}(\hat{\mathcal{I}}, \hat{\Pi} | \mathcal{I})$$



<Overall training pipeline>

```
class LearnPose(nn.Module):
    def __init__(self, num_cams, learn_R, learn_t, cfg, init_c2w=None):
        """
        :param num_cams:
        :param learn_R:  True/False
        :param learn_t:  True/False
        :param cfg: config arguments options
        :param init_c2w: (N, 4, 4) torch tensor
        """
        super(LearnPose, self).__init__()
        self.num_cams = num_cams
        self.init_c2w = None
        if init_c2w is not None:
            self.init_c2w = nn.Parameter(init_c2w, requires_grad=False)
        self.r = nn.Parameter(torch.zeros(size=(num_cams, 3), dtype=torch.float32), requires_grad=learn_R)  # (N, 3)
        self.t = nn.Parameter(torch.zeros(size=(num_cams, 3), dtype=torch.float32), requires_grad=learn_t)  # (N, 3)

    def forward(self, cam_id):
        cam_id = int(cam_id)
        r = self.r[cam_id]  # (3, ) axis-angle
        t = self.t[cam_id]  # (3, )
        c2w = make_c2w(r, t)  # (4, 4)
        # learn a delta pose between init pose and target pose, if a init pose is provided
        if self.init_c2w is not None:
            c2w = c2w @ self.init_c2w[cam_id]
        return c2w
    def get_t(self):
        return self.t
```

torch.nn.Parameter 함수 사용하여 learnable한 변수로 선언

<Implementation>

NoPe-NeRF[1)]

- Training pipeline
  - Undistortion of monocular depth
    - Monocular depth estimation network인 DPT[2)]를 사용하여 input image $\mathcal{I}$에 대한 mono-depth map $\mathcal{D} = \{D_i | i = 0 \dots N-1\}$ 을 얻음 동일한 장면을 다른 시점에서 관찰할 때 rendering 결과가 일관되는 성질
    - Mono-depth map $\mathcal{D}$는 multi-view consistency가 없기 때문에 transform parameter $\psi = \{(\alpha_i, \beta_i) | i = 0 \dots N-1\}$을 사용하여 다음과 같이 view-consistent depth map $\mathcal{D}^*$을 모델링 함

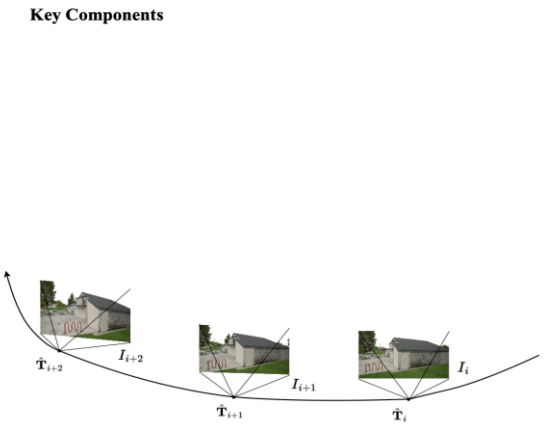
$$D_i^* = \alpha_i D_i + \beta_i$$

    - 다음과 같은 loss function $\mathcal{L}_{depth}$을 통해 $\mathcal{D}^*$ 가 view-consistent한 NeRF의 depth map $\hat{D}_i$에 가까워지도록 학습

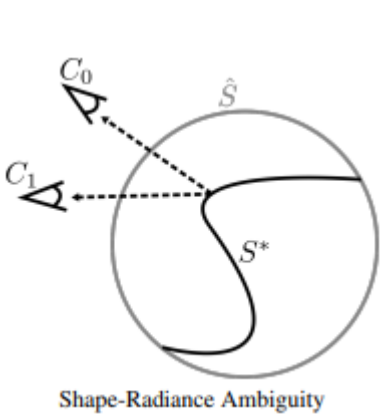
$$\mathcal{L}_{depth} = \sum_i^N \left\| D_i^* - \hat{D}_i \right\| \text{ where } \hat{D}_i(r) = \int_{h_n}^{h_f} T(h)\sigma\big(r(h)\big)dh$$

유사한 2D image를 생성할 수 있는 3D 장면이 여러 개 있을 때 발생
→View마다 다른 모양의 image 생성

    - Mono-depth 는 NeRF를 학습시키는 데에 geometry prior로 작용하여 shape-radiance ambiguity를 완화



<Undistorted depth map 생성 과정>



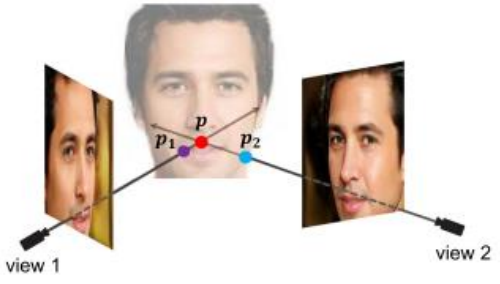


Fig. 3 Visualization of shape-radiance ambiguity. For illustration, we assume p (the red dot) is the location of correct geometry, p_1 (the violet dot) and p_2 (the blue dot) are incorrect geometries. In the absence of geometry constraints, the model can fit incorrect geometry p_1 in view 1 and p_2 in view 2 independently to simulate the effect of the correct geometry p (Color figure online)

<Illustration of shape-radiance ambiguity>

NoPe-NeRF[1)]

- Training pipeline
  - Relative pose constraint
    - 기존 방법들은 각 camera pose를 독립적으로 최적화하여 global pose 정보를 학습하지 못함
    - 이를 해결하기 위해 인접 frame간에 잘못된 pose 추정에 페널티를 줄 수 있는 loss function을 도입
    - Undistorted depth map $\mathcal{D}^*$을 camera intrinsic을 사용하여 back-projection해서 point cloud $\mathcal{P}^* = \{P_i^* | i = 0 \dots N-1\}$을 얻음
    - 다음과 같은 point cloud loss $\mathcal{L}_{pc}$를 통해 인접 frame이 3D 공간에서 matching되도록 하여 pose를 최적화 함

Depth map $\mathcal{D}^*$상의 pixel (i, j)과 3차원 point cloud 상의 좌표 (x, y, z)는 다음과 같은 관계를 가짐

$$\begin{cases} z = \mathcal{D}^*(i,j) \\ x = \dfrac{(j - c_x) \times z}{f_x} \\ y = \dfrac{(i - c_y) \times z}{f_y} \end{cases}$$

$$\mathcal{L}_{pc} = \sum_{(i,j)} l_{cd}(P_j^*, T_{ji} P_i^*)$$

$T_{ji} = T_j T_i^{-1}$
- P_i^*는 i번째 camera 좌표계 상에서 표현됨
- T_i^{-1}는 camera 좌표계를 world 좌표계로 변환
- T_j는 world 좌표계를 j번째 camera 좌표계로 전환

→ **결과적으로 P_i^*를 j번째 camera 좌표계로 변환**

Chamfer Distance loss, 2개의 Point cloud 각 점에 대해서 상대 point cloud의 가장 가까운 점의 L2 loss를 더하여 계산되며 **point cloud간의 유사도를 높이는 역할을 수행**

$$l_{cd}(P_i, P_j) = \sum_{p_i \in P_i} \min_{p_j \in P_j} \|p_i - p_j\|_2 + \sum_{p_j \in P_j} \min_{p_i \in P_i} \|p_i - p_j\|_2$$

- 또한 point cloud 상의 point가 인접 frame에 projection된 값이 같아지도록 하는 surface-based photometric error $\mathcal{L}_{rgb-s}$를 도입하여 일정한 appearance를 가지도록 유도

$$\mathcal{L}_{rgb-s} = \sum_{(i,j)} \| I_i\langle K_i P_i^* \rangle - I_j\langle K_j T_j T_i^{-1} P_i^* \rangle \|$$

P_i^*를 i번째와 j번째 image plane에 project하였을 때 유사한 색을 가지도록 유도

→ **View들 간의 photometric consistency 향상**

NoPe-NeRF[1)]

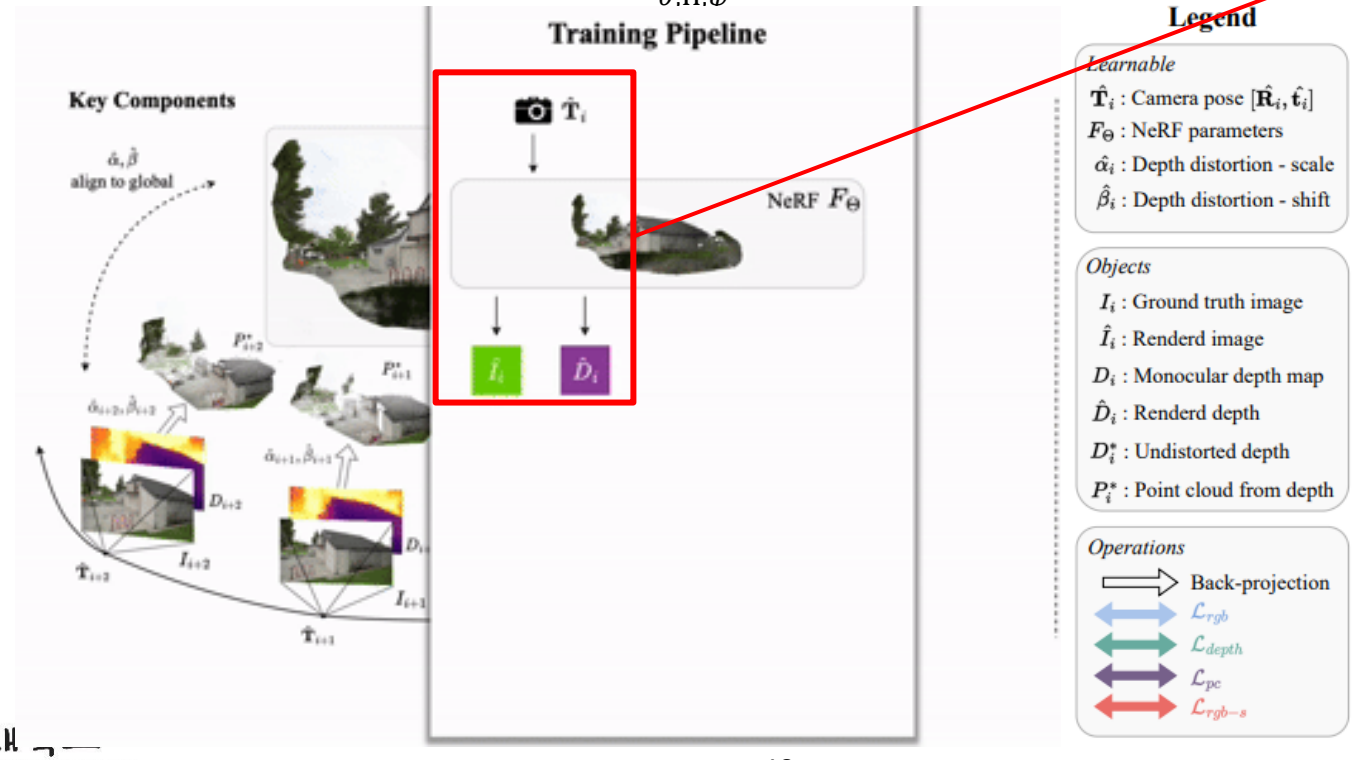
- Training pipeline
  - Total loss function
    - 최종적으로 loss function $\mathcal{L}$은 아래와 같이 정의됨

$$\mathcal{L} = \mathcal{L}_{rgb} + \lambda_1 \mathcal{L}_{depth} + \lambda_2 \mathcal{L}_{pc} + \lambda_3 \mathcal{L}_{rgb-s}$$

    - 또한 $\mathcal{L}$을 다음과 같이 NeRF parameter θ, camera pose Π, distortion parameter ψ를 최적화 함

$$\theta^*, \Pi^*, \psi^* = \arg\min_{\theta,\Pi,\psi} \mathcal{L}(\hat{\mathcal{I}}, \hat{\mathcal{D}}, \hat{\Pi}, \hat{\psi} | \mathcal{I}, \mathcal{D})$$

Learnable한 pose를 NeRF에 입력시켜 출력 image와 depth map을 얻은 후 제안한 loss function을 최적화



NoPe-NeRF[1)]

- Experimental Results
  - Comparisons with pose-unknown methods
    - View synthesis quality

Rendered depth map $\hat{D}_i(r) = \int_{h_n}^{h_f} T(h)\sigma(r(h))dh$

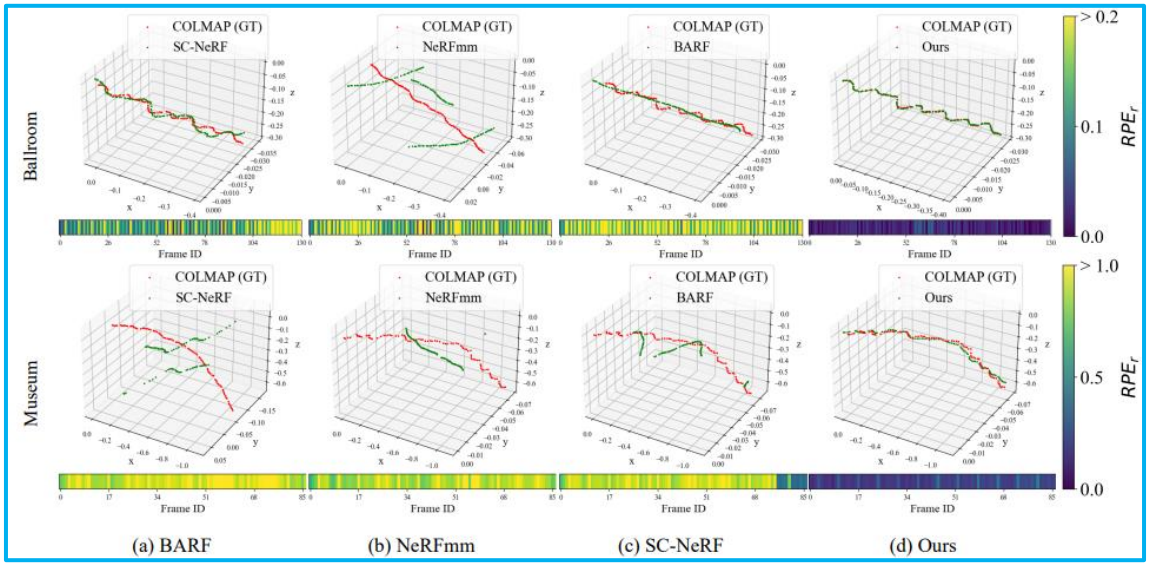
(a) BARF (b) NeRFmm (c) SC-NeRF (d) Ours (e) Ground Truth

	scenes	Ours			BARF			NeRFmm			SC-NeRF		
		PSNR ↑	SSIM ↑	LPIPS ↓	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
ScanNet	0079_00	**32.47**	**0.84**	**0.41**	32.31	0.83	0.43	30.59	0.81	0.49	31.33	0.82	0.46
	0418_00	**31.33**	**0.79**	**0.34**	31.24	**0.79**	0.35	30.00	0.77	0.40	29.05	0.75	0.43
	0301_00	**29.83**	**0.77**	0.36	29.31	0.76	0.38	27.84	0.72	0.45	29.45	**0.77**	**0.35**
	0431_00	**33.83**	**0.91**	**0.39**	32.77	0.90	0.41	31.44	0.88	0.45	32.57	0.90	0.40
	mean	**31.86**	**0.83**	**0.38**	31.41	0.82	0.39	29.97	0.80	0.45	30.60	0.81	0.41
Tanks and Temples	Church	**25.17**	**0.73**	**0.39**	23.17	0.62	0.52	21.64	0.58	0.54	21.96	0.60	0.53
	Barn	**26.35**	**0.69**	**0.44**	25.28	0.64	0.48	23.21	0.61	0.53	23.26	0.62	0.51
	Museum	**26.77**	**0.76**	**0.35**	23.58	0.61	0.55	22.37	0.61	0.53	24.94	0.69	0.45
	Family	**26.01**	**0.74**	**0.41**	23.04	0.61	0.56	23.04	0.58	0.56	22.60	0.63	0.51
	Horse	**27.64**	**0.84**	**0.26**	24.09	0.72	0.41	23.12	0.70	0.43	25.23	0.76	0.37
	Ballroom	**25.33**	**0.72**	**0.38**	20.66	0.50	0.60	20.03	0.48	0.57	22.64	0.61	0.48
	Francis	**29.48**	**0.80**	**0.38**	25.85	0.69	0.57	25.40	00.69	0.52	26.46	0.73	0.49
	Ignatius	**23.96**	**0.61**	**0.47**	21.78	0.47	0.60	21.16	0.45	0.60	23.00	0.55	0.53
	mean	**26.34**	**0.74**	**0.39**	23.42	0.61	0.54	22.50	0.59	0.54	23.76	0.65	0.48

NoPe-NeRF[1)]

<Metrics>
- RPE-t : GT pose와 prediction 각각에 대해 인접 view와 position 변화량의 변화량을 평균내서 계산
- RPE-r : GT pose와 prediction 각각에 대해 인접 view와 rotation 변화량의 변화량을 평균내서 계산
- ATE : GT pose와 predicted pose사이의 MSE

- Experimental Results
  - Comparisons with pose-unknown methods
    - Camera pose



Camera trajectory 시각화

	scenes	Ours			BARF			NeRFmm			SC-NeRF		
		$RPE_t \downarrow$	$RPE_r \downarrow$	ATE↓	RPE_t	RPE_r	ATE	RPE_t	RPE_r	ATE	RPE_t	RPE_r	ATE
ScanNet	0079_00	**0.752**	**0.204**	**0.023**	1.110	0.480	0.062	1.706	0.636	0.100	2.064	0.664	0.115
	0418_00	**0.455**	**0.119**	**0.015**	1.398	0.538	0.020	1.402	0.460	0.013	1.528	0.502	0.016
	0301_00	**0.399**	**0.123**	**0.013**	1.316	0.777	0.219	3.097	0.894	0.288	1.133	0.422	0.056
	0431_00	**1.625**	**0.274**	**0.069**	6.024	0.754	0.168	6.799	0.624	0.496	4.110	0.499	0.205
	mean	**0.808**	**0.180**	**0.030**	2.462	0.637	0.117	3.251	0.654	0.224	2.209	0.522	0.098
Tanks and Temples	Church	**0.034**	**0.008**	**0.008**	0.114	0.038	0.052	0.626	0.127	0.065	0.836	0.187	0.108
	Barn	**0.046**	**0.032**	**0.004**	0.314	0.265	0.050	1.629	0.494	0.159	1.317	0.429	0.157
	Museum	**0.207**	**0.202**	**0.020**	3.442	1.128	0.263	4.134	1.051	0.346	8.339	1.491	0.316
	Family	**0.047**	**0.015**	**0.001**	1.371	0.591	0.115	2.743	0.537	0.120	1.171	0.499	0.142
	Horse	**0.179**	**0.017**	**0.003**	1.333	0.394	0.014	1.349	0.434	0.018	1.366	0.438	0.019
	Ballroom	**0.041**	**0.018**	**0.002**	0.531	0.228	0.018	0.449	0.177	0.031	0.328	0.146	0.012
	Francis	**0.057**	**0.009**	**0.005**	1.321	0.558	0.082	1.647	0.618	0.207	1.233	0.483	0.192
	Ignatius	**0.026**	**0.005**	**0.002**	0.736	0.324	0.029	1.302	0.379	0.041	0.533	0.240	0.085
	mean	**0.080**	**0.038**	**0.006**	1.046	0.441	0.078	1.735	0.477	0.123	1.890	0.489	0.129

NoPe-NeRF[1)]

- Experimental Results
  - Comparisons with COLMAP assisted NeRF
    - Comparison of pose accuracy with COLMAP on ScanNet

ScanNet데이터셋의 GT pose를 사용하여 제안 방법과 COLMAP의 pose 추정 정확도 측정
→ COLMAP과 비슷한 수준의 정확도 달성

scenes	Ours			COLMAP		
	$RPE_t \downarrow$	$RPE_r \downarrow$	ATE↓	RPE_t	RPE_r	ATE
0079_00	0.752	**0.204**	0.023	**0.655**	0.221	**0.012**
0418_00	**0.455**	**0.119**	**0.015**	0.491	0.124	0.016
0301_00	**0.399**	**0.123**	0.013	0.414	0.136	**0.009**
0431_00	1.625	0.274	0.069	**1.292**	**0.249**	**0.051**
mean	0.808	**0.180**	0.030	**0.713**	0.182	**0.022**

NeRF는 2stage 학습 방식을 사용하므로, 공정한 비교를 위해 이와 유사한 2stage 학습 방법을 적용한 모델.
먼저, 제안 방법으로 NeRF와 pose를 학습한 후 pose를 고정한채 NeRF를 scratch부터 학습

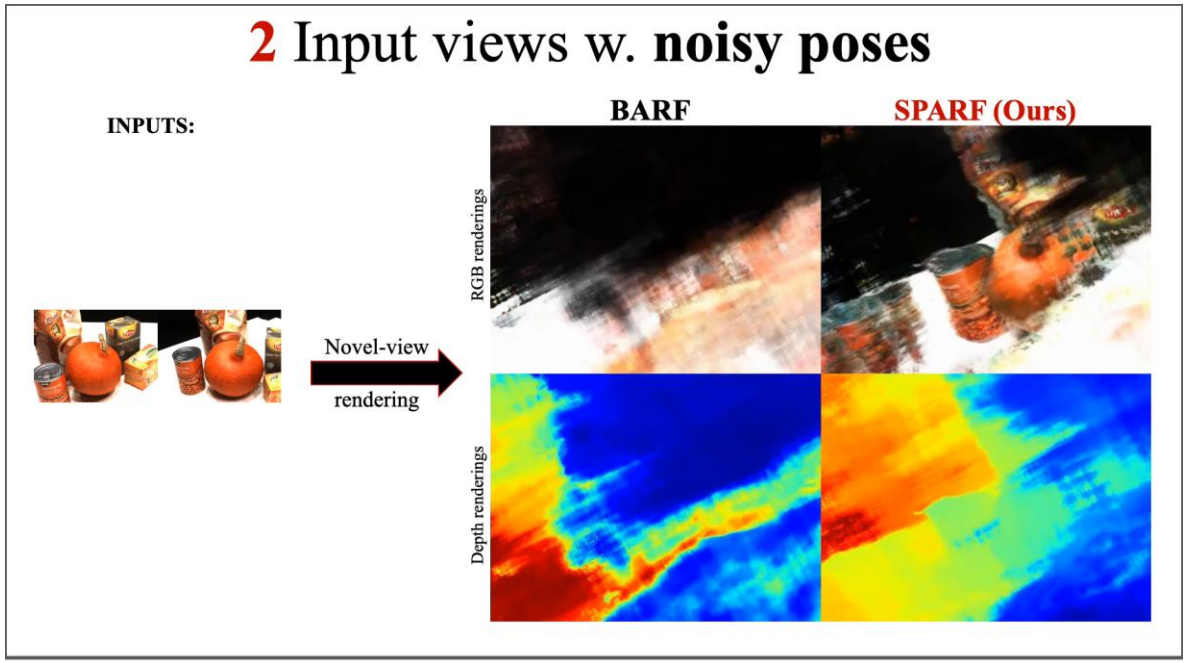
    - Comparison to NeRF with COLMAP poses

	scenes	Ours			Ours-r			COLMAP+NeRF		
		PSNR ↑	SSIM ↑	LPIPS ↓	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
ScanNet	0079_00	32.47	0.84	0.41	**33.12**	**0.85**	**0.40**	31.98	0.83	0.43
	0418_00	**31.33**	**0.79**	**0.34**	30.49	0.77	0.40	30.60	0.78	0.40
	0301_00	29.83	0.77	0.36	**30.05**	**0.78**	**0.34**	30.01	**0.78**	0.36
	0431_00	33.83	0.91	0.39	**33.86**	**0.91**	**0.39**	33.54	**0.91**	**0.39**
	mean	31.86	**0.83**	**0.38**	**31.88**	**0.83**	**0.38**	31.53	0.82	0.40
Tanks and Temples	Church	25.17	0.73	0.39	**26.74**	**0.78**	**0.32**	25.72	0.75	0.37
	Barn	26.35	0.69	0.44	26.58	**0.71**	**0.42**	**26.72**	**0.71**	**0.42**
	Museum	26.77	0.76	0.35	26.98	0.77	0.36	**27.21**	**0.78**	**0.34**
	Family	26.01	0.74	0.41	26.21	0.75	0.40	**26.61**	**0.77**	**0.39**
	Horse	27.64	**0.84**	**0.26**	**28.06**	**0.84**	**0.26**	27.02	0.82	0.29
	Ballroom	25.33	0.72	**0.38**	**25.53**	**0.73**	**0.38**	25.47	**0.73**	**0.38**
	Francis	29.48	0.80	**0.38**	29.73	**0.81**	**0.38**	30.05	**0.81**	**0.38**
	Ignatius	23.96	0.61	0.47	23.98	**0.62**	**0.46**	24.08	0.61	0.47
	mean	26.34	0.74	0.39	**26.73**	**0.75**	**0.37**	26.61	**0.75**	0.38

SPARF: Neural Radiance Fields from Sparse and Noisy Poses

SPARF[1)]

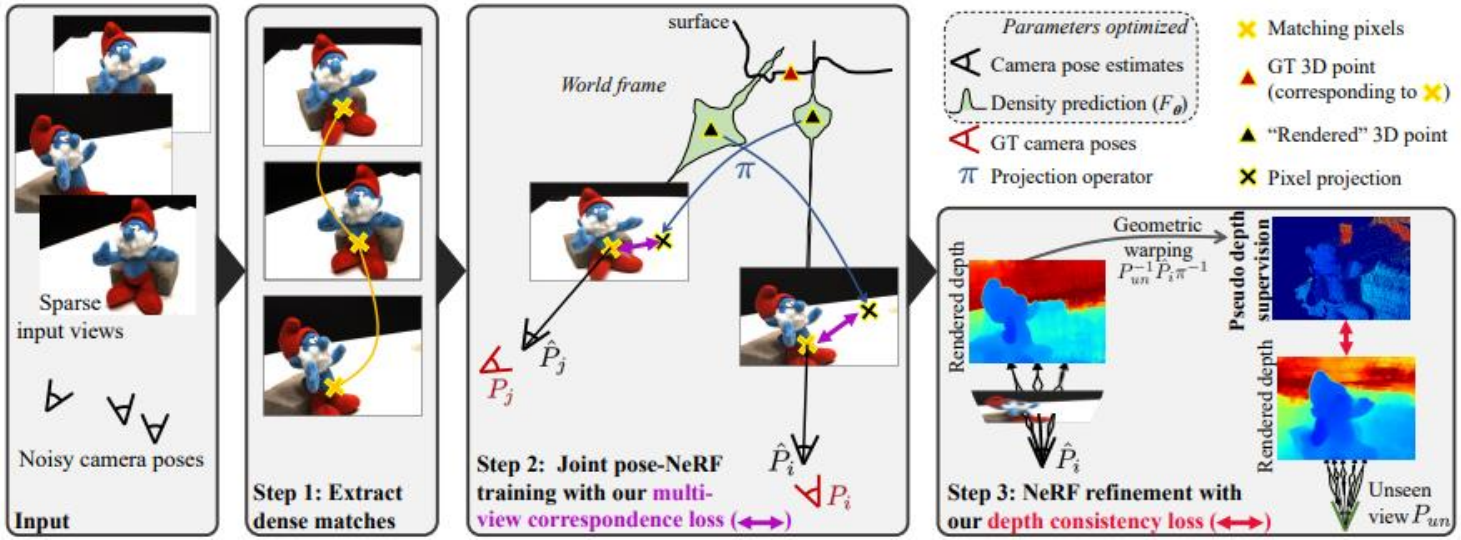
- Abstract
  - Few-shot 조건에서 noisy camera pose를 사용하여 3D scene을 표현하는 방법 제안
    - 기존 few-shot 조건을 다룬 NeRF 방법론들은 정확한 camera pose를 필요로 함
    - Camera pose 추정을 위해 자주 사용되는 COLMAP은 sparse view에 대해 성능이 크게 저하됨
    - 본 논문은 multi-view geometry에 기반한 explicit한 constraint를 적용하여 few-shot 조건에서 NeRF와 camera pose를 함께 최적화하는 방법을 제안함
    - 또한 depth consistency loss를 제안하여 모든 view에서 일관된 rendering 결과를 가지도록 유도함



<Fewshot 조건에서 BARF와 성능 비교>

SPARF[1)]

- Method Overview
  - Training NeRF with few-shot images and noisy camera poses
    - 기존에 pose와 NeRF를 함께 최적화하는 방법은 image마다 독립적으로 pose를 최적화 함
    - 이는 spare-view scenario에서 scene을 under-constraint하므로 명시적으로 training image간의 geometry를 이용할 필요가 있음
    - 본 논문은 multi-view geometry에 기반한 loss function을 도입하여 기존 방법의 문제점을 해결
    - 또한 training 중에 학습한 depth map을 사용하여 novel view point에서 rendering된 image에 일관성을 부여하는 depth consistency loss를 제안



SPARF[1)]

- Method
  - Multi-view correspondence loss
    - 인접 image pair (I_i, I_j)에 대해 correspondence $\mathbf{p} \in I_i$와 $\mathbf{q} \in I_j$를 얻음
    - 이후 **p**와 **q**에 대해 NeRF를 사용하여 다음과 같이 depth $\hat{z}$를 추정함

$$\hat{z}_{i,p} = \hat{z}(\mathbf{p}; \theta, \hat{P}_i) = \sum_{m=1}^{M} \alpha_m t_m \qquad \alpha_m = T_m(1 - \exp(-\sigma_m \delta_m))$$

Noisy pose

3D상에서 이 두 점이 가까워 지도록 loss를 설정하는 것도 생각할 수 있으나, 이는 scene scale과 initial camera pose에 따라 편차가 커져서 많은 tuning이 필요함

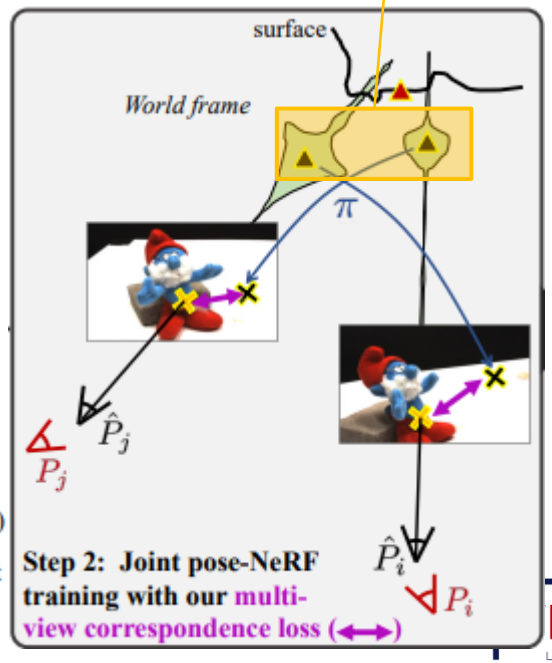
- Correspondence는 같은 점으로 back projection돼야 하는 것에 착안하여 다음과 같은 multi-view correspondence loss function $\mathcal{L}_{\text{MVCorr}}$도입

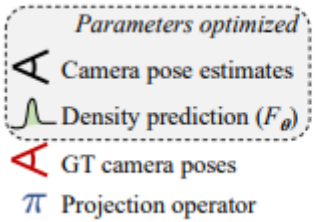
$$\mathcal{L}_{\text{MVCorr}}(\theta, \hat{\mathcal{P}}) = \sum_{\text{p} \in \mathcal{V}} \mathcal{L}_{\text{p}},$$

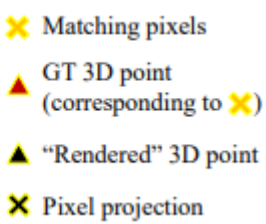
$$\mathcal{L}_{\text{p}} = w_{\text{p}} \rho \left(\mathbf{q} - \pi \left(\hat{P}_j^{-1} \hat{P}_i \pi^{-1} \left(\mathbf{p}, \hat{z}(\mathbf{p}; \theta, \hat{P}_i) \right) \right) \right).$$

- w_p: Correspondence의 confidence, ($w_p \in [0,1]$)
- $\mathcal{V} = \{\text{p}: \text{w}_\text{p} \geq k\}$
- π : camera projection matrix (camera coordinate → pixel coordinate)
- ρ: Huber loss function,

$$\rho(a) = \begin{cases} \frac{1}{2}a^2 & \text{for } |a| < \delta, \\ \delta(|a| - \frac{1}{2}\delta), & \text{othetwise.} \end{cases}$$







<Multi-view correspondence loss>

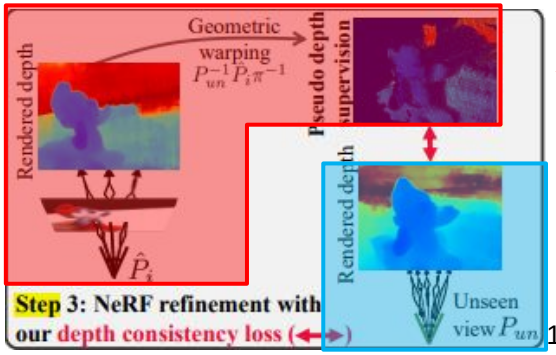
SPARF[1)]

- Method
  - Improving geometry at unobserved views
    - View consistency를 향상 시키기 위해 training view point를 pseudo-depth map으로 사용하는 depth consistency loss $\mathcal{L}_{\text{DCons}}$를 도입

$$\mathcal{L}_{\text{DCons}}(\theta) = \sum_{\text{p}} \gamma_y \rho(z_y - \hat{z}(\mathbf{y}; \theta, P_{un}))$$

    - 여기서 pseudo-depth z_y는 error를 포함하기 때문에 $\hat{P}_i$에는 back propagation을 수행하지 않아야 성능이 향상되었음을 서술
    - 또한 visibility mask γ_y는 다음과 같은 기준으로 결정됨
      - 1. Pixel projection $\mathbf{y}$가 virtual view의 image plane 밖으로 나가면 $\gamma_y = 0$
      - 2. Occluded인 경우를 제외하기 위해 $\gamma_y = T_{un,z_y}$

      →Transmittance가 크면 camera center와 $r_{un,y}(z_y)$ 사이에 density가 큰 point가 거의 없는 것이고 따라서 depth에 큰 영향을 주는 표면상의 점이므로 γ_y도 큰 값으로 설정



$\mathbf{r}_{\text{p}}^{un} = P_{un}^{-1}\hat{P}_i\pi^{-1}\left(\mathbf{p}, \hat{z}(\mathbf{p}; \theta, \hat{P}_i)\right)$

$z_y = \left[r_p^{un}\right]_3$

→z_y는 training view 상의 point $\mathbf{p}$를 unseen view 방향으로 warping한 후 projection한 depth

$\mathbf{y} = \pi(\mathbf{r}_{\text{p}}^{un})$는 $\mathbf{p}$를 unseen view에 projection한 pixel이며, $\hat{z}(\mathbf{y}; \theta, P_{un})$는 y에서 NeRF를 사용하여 추정한 depth

SPARF[1)]

- Method
  - Training Framework
    - 최종 loss function은 다음과 같이 계산됨

$$\mathcal{L}(\theta, \hat{\mathcal{P}}) = \mathcal{L}_{\text{photo}}(\theta, \hat{\mathcal{P}}) + \lambda_c \mathcal{L}_{\text{MVCorr}}(\theta, \hat{\mathcal{P}}) + \lambda_d \mathcal{L}_{\text{DCons}}(\theta)$$

$$\mathcal{L}_{\text{photo}}(\theta, \hat{\mathcal{P}}) = \frac{1}{n} \sum_{i=1}^{n} \sum_{\text{p}} \left\| I_i(\mathbf{p}) - \hat{I}(\mathbf{p}; \theta, \hat{\mathcal{P}}_i) \right\|_2^2$$

    - 먼저 coarse network F_{θ_c}와 pose를 함께 최적한 후, 학습된 pose를 freeze하고 coarse network F_{θ_c}, fine network F_{θ_f}를 함께 최적화 함

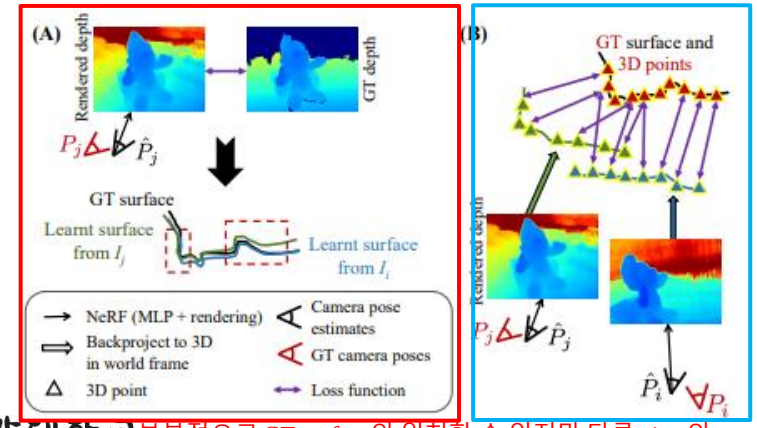
$$\mathcal{L}_{\text{total}} = \sum_{r \in R} \mathcal{L}(\theta_c, \hat{\mathcal{P}}) + \mathcal{L}(\theta_f, \hat{\mathcal{P}})$$

    - 이 과정에서 fine-network는 pre-trained된 coarse network를 통해 sharp한 geometry를 학습하게 됨
    - 또한 학습이 진행됨에 따라 점진적으로 positional encoding의 high-frequency component를 사용하는 방식을 사용함
      - Few-shot setting에서 이와 같은 방식을 사용하는 논문이 다수 존재

SPARF[1)]

- Experimental Results
  - Intuition on pose-NeRF training losses
    - Rendered depth와 GT depth 사이에 L1 loss를 사용하여 학습하면 성능이 크게 저하됨
    - 반면에 rendered depth를 back projection하여 GT 3D point와 거리를 loss function으로 사용하면 성능이 향상됨

    → Sparse-view setting에서 training image간의 geometry를 명시적으로 이용하는 것이 매우 중요
  - Comparison of losses for pose-NeRF training
    - 일반적인 photometric loss와 제안한 multi-view correspondence loss를 사용 했을 때 pose추정 및 NeRF 성능을 비교하는 실험 진행
    - Photometric loss는 단일 image에 대해 pose가 최적화 되어 global한 정보를 학습하지 못함
    - 반면에 제안한 loss function을 사용하면 인접 frame간의 pose에 일관성을 부여하여 성능이 크게 상승



Losses	Rot. ↓	Trans. ↓	PSNR ↑	SSIM ↑	LPIPS ↓	DE↓
I Photo. + L1 **GT** depth	7.3	28.9	13.8 (14.0)	0.54 (0.70)	0.46 (0.27)	0.17
II Photo. + L1 **GT** 3D points	**0.4**	**1.5**	**18.8 (20.3)**	**0.75 (0.84)**	**0.21 (0.11)**	**0.07**
III Photo. (7)	10.3	51.5	10.7 (9.8)	0.43 (0.62)	0.59 (0.36)	1.9
IV Photo. + mask loss [24, 60]	13.2	57.7	-	-	-	-
V MVCorr (8)	1.98	6.6	-	-	-	0.19
VI **Photo. + MVCorr** ((7)-(8))	**1.85**	**5.5**	**16.0 (17.8)**	**0.68 (0.81)**	**0.28 (0.14)**	**0.13**

부분적으로 GT surface와 일치할 수 있지만 다른 view와 불일치로 인해 global solution을 얻지 못함

SPARF[1)]

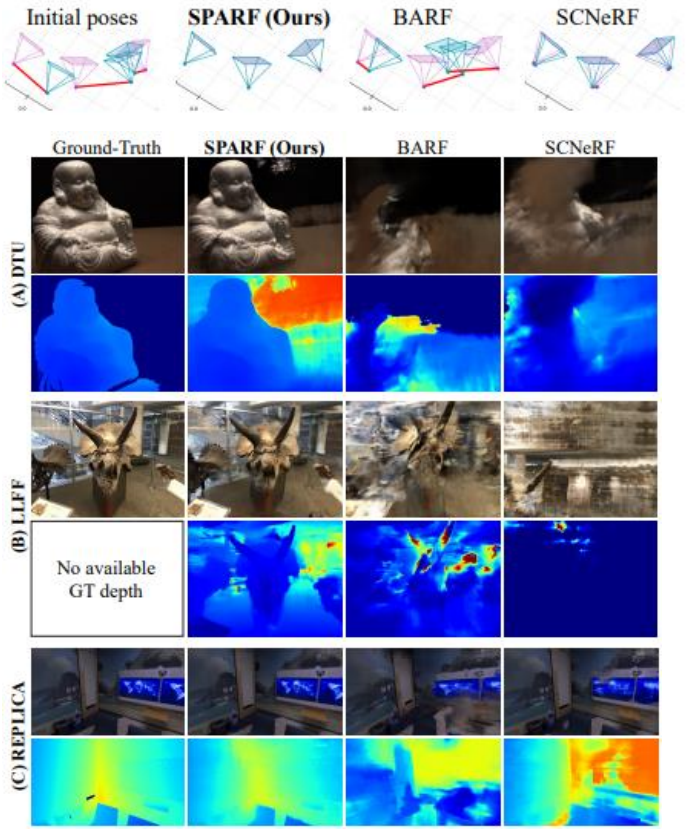
- Experimental Results
  - Comparison to SOTA with noisy poses

Method	Rot. ↓	Trans. ↓	PSNR ↑	SSIM ↑	LPIPS ↓	DE ↓
BARF [25]	10.33	51.5	10.71 (9.76)	0.43 (0.62)	0.59 (0.36)	1.90
RegBARF [25, 33]	11.20	52.8	10.38 (9.20)	0.45 (0.62)	0.61 (0.38)	2.33
DistBARF [4, 25]	11.69	55.7	9.50 (9.15)	0.34 (0.76)	0.67 (0.36)	1.90
SCNeRF [21]	3.44	16.4	12.04 (11.71)	0.45 (0.66)	0.52 (0.30)	0.85
SPARF (Ours)	**1.81**	**5.0**	**17.74 (18.92)**	**0.71 (0.83)**	**0.26 (0.13)**	**0.12**

	Rot. (°) ↓	Trans. (×100) ↓	PSNR ↑	SSIM ↑	LPIPS ↓
BARF [25]	2.04	11.6	17.47	0.48	0.37
RegBARF [25, 33]	1.52	5.0	18.57	0.52	0.36
DistBARF [4, 25]	5.59	26.5	14.69	0.34	0.49
SCNeRF [21]	1.93	11.4	17.10	0.45	0.40
SPARF (Ours)	**0.53**	**2.8**	**19.58**	**0.61**	**0.31**

	Method	Rot (°) ↓	Trans (×100) ↓	PSNR ↑	SSIM ↑	LPIPS ↓	DE ↓
G	**SPARF (Ours)**	Fixed GT poses		*26.43*	**0.88**	**0.13**	*0.39*
F	NeRF [31]	Fixed poses obtained		20.99	0.73	0.32	1.33
	DS-NeRF [11]	from COLMAP (run w.		23.52	0.81	0.20	0.99
	SPARF (Ours)	PDC-Net [45] matches)		25.03	*0.84*	*0.15*	0.66
R	BARF [25]	3.35	16.96	20.73	0.72	0.30	0.84
	RegBARF [25, 33]	3.66	20.87	20.00	0.70	0.32	1.00
	DistBARF [4, 25]	2.36	7.73	22.46	0.77	0.23	0.47
	SCNeRF [21]	*0.65*	*4.12*	22.54	0.79	0.24	0.73
	DS-NeRF [11]	1.30	5.04	24.75	0.83	0.20	0.69
	SPARF (Ours)	**0.15**	**0.76**	**26.98**	**0.88**	**0.13**	**0.36**

Noisy한 initial pose로 학습을 시작해도, 학습이 완료되면 GT pose와 같은 pose를 추정함



  - Comparison to SOTA with ground-truth poses

Method	DTU PSNR ↑	DTU SSIM ↑	DTU LPIPS ↓	LLFF PSNR ↑	LLFF SSIM ↑	LLFF LPIPS ↓
PixelNeRF [57]	**19.36** (18.00)	*0.70* (*0.77*)	*0.32* (0.23)	7.93	0.27	0.68
PixelNeRF-ft [57]	-	-	-	16.17	0.44	0.51
MipNeRF [3]	7.64 (8.68)	0.23 (0.57)	0.66 (0.35)	14.62	0.35	0.50
NeRF [31]	8.41 (9.34)	0.31 (0.63)	0.71 (0.36)	13.61	0.28	0.56
DietNeRF [19]	10.01 (11.85)	0.35 (0.63)	0.57 (0.31)	14.94	0.37	0.5
InfoNeRF [22]	11.23 (-)	0.44 (-)	0.54 (-)	-	-	-
RegNeRF [33]	15.33 (*18.89*)	0.62 (0.75)	0.34 (*0.19*)	*19.08*	*0.59*	0.34
DS-NeRF [11]	16.52 (-)	0.54 (-)	0.48 (-)	18.00	0.55	*0.27*
SPARF (Ours)	*18.30* (**21.01**)	**0.78** (**0.87**)	**0.21** (**0.10**)	**20.20**	**0.63**	**0.24**

(·)은 background를 제거한 성능, DTU 데이터셋은 큰 검은색 배경을 가져서 잘못된 추정을 해도 성능이 높을 수 있음