『2026 동계 세미나』

Exploring 3D foundation models



Pusan National University
Dept. of Electronics Engineering
Computer Vision & Signal Processing Lab.



Presented By
진인환

3D Reconstruction

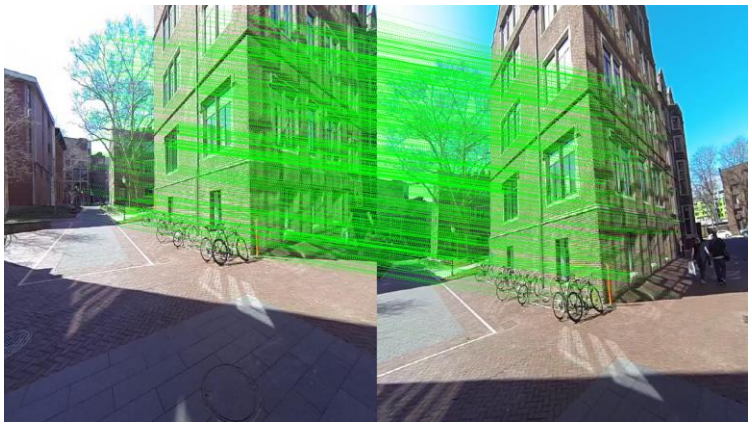
- Task
  - 3D Reconstruction
    - 다시점의 이미지로부터 가상의 3D space에 물체 또는 장면의 표현을 생성



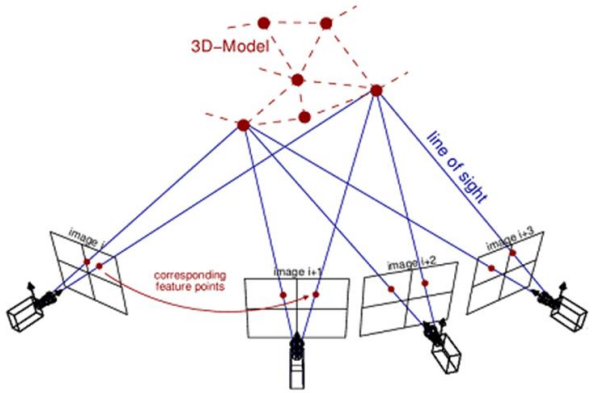
< 3D reconstructed output>

3D Reconstruction

- Task
  - Structure-from-Motion(SfM)
    - Spare 3D maps과 camera pose를 joint하게 추정
      - Camera pose(intrinsic, extrinsic parameter) 및 Sparse point cloud 추정
    - Keypoint matching을 통해 Pixel correspondences를 얻어 이미지 간의 geometric relationships를 얻음
      - bundle adjustment를 통해 카메라 파라미터와 3D 좌표를 joint optimize 수행



< keypoints matching >

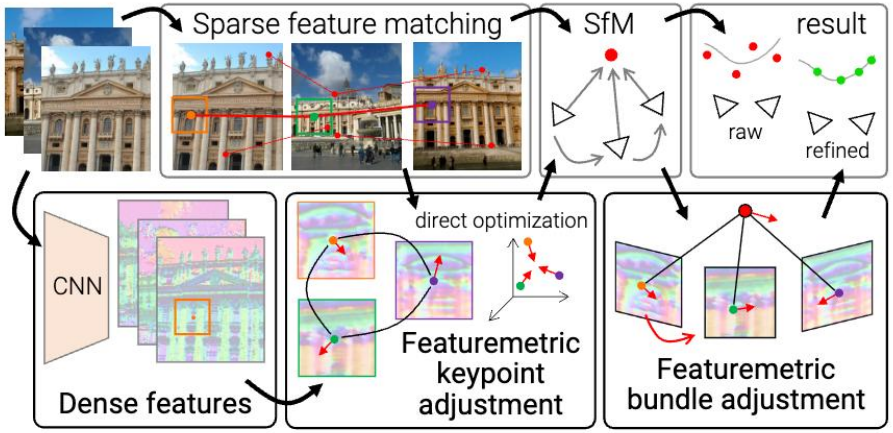


< Bundle adjustment >

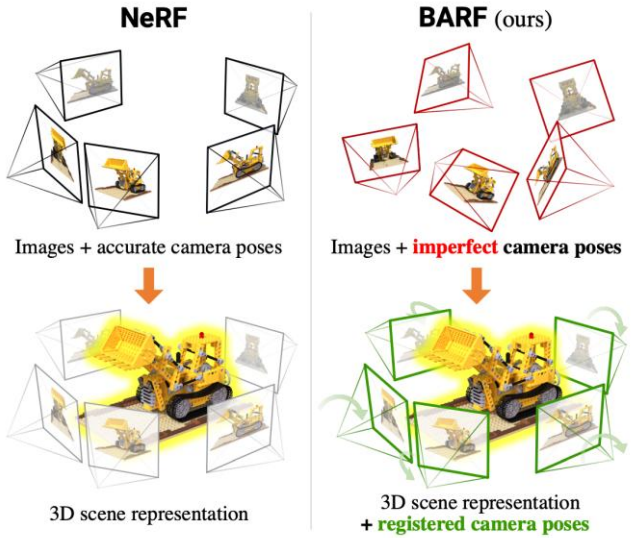
[1] Lindenberger, Philipp, et al. "Pixel-perfect structure-from-motion with featuremetric refinement." Proceedings of the IEEE/CVF international conference on computer vision. 2021.
[2] Lin, Chen-Hsuan, et al. "Barf: Bundle-adjusting neural radiance fields." Proceedings of the IEEE/CVF international conference on computer vision. 2021.

3D Reconstruction

- Task
  - Structure-from-Motion(SfM)
    - 최근의 learning-based의 방법론이 제안되고 있음
      - SfM의 subprocess에 learning-based techniqes를 도입하여 noise에대한 robustness를 향상시킴
        - ✓Featuremetric refinement [1]: Neural network로 추출한 feature에서 keypoint를 refine
        - ✓Neural bundle adjustment [2]: NeRF의 학습과정에서 camera parameter를 함께 업데이트
    - 하지만 여전히 개별 구성요소들이 noise와 error에 취약함



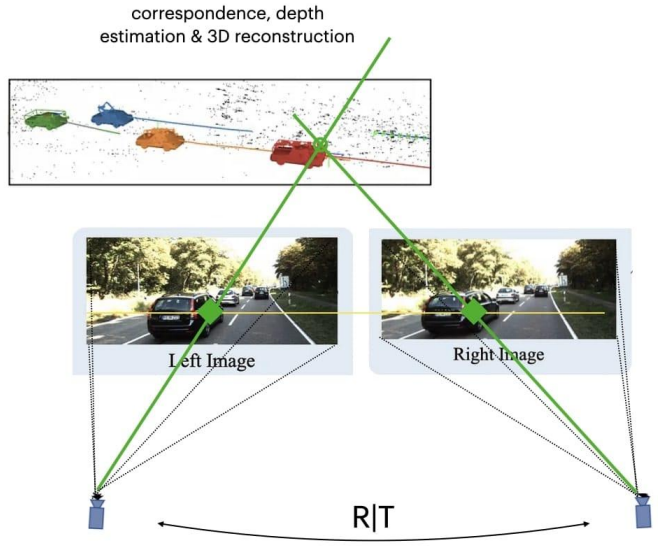
< Featuremetric refinement >



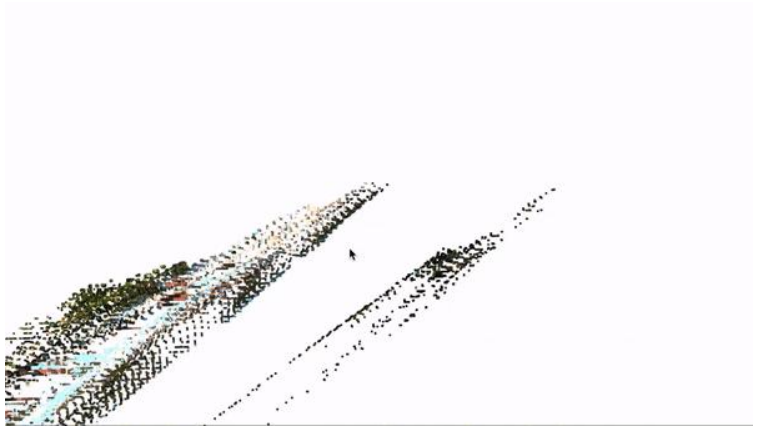
< Neural bundle adjustment >

3D Reconstruction

- Task
  - Multi-view stereo reconstruction(MVS)
    - 전체 viewpoint의 Camera parameter가 주어졌을 때 visible surface를 dense하게 reconstruction하는 task
      - 카메라 파라미터는 SfM 알고리즘으로 추정하거나, 측정된 데이터 활용
    - 다중 시점에서 triangulation을 통해 입력 카메라로부터 표면을 조밀하게 복원함
    - 입력 카메라 파라미터의 의존도가 매우 높음



< Multi-View Stereo >



< Two-view Reconstruction >

3D Reconstruction

- Task
  - SfM & MVS



< SfM results >

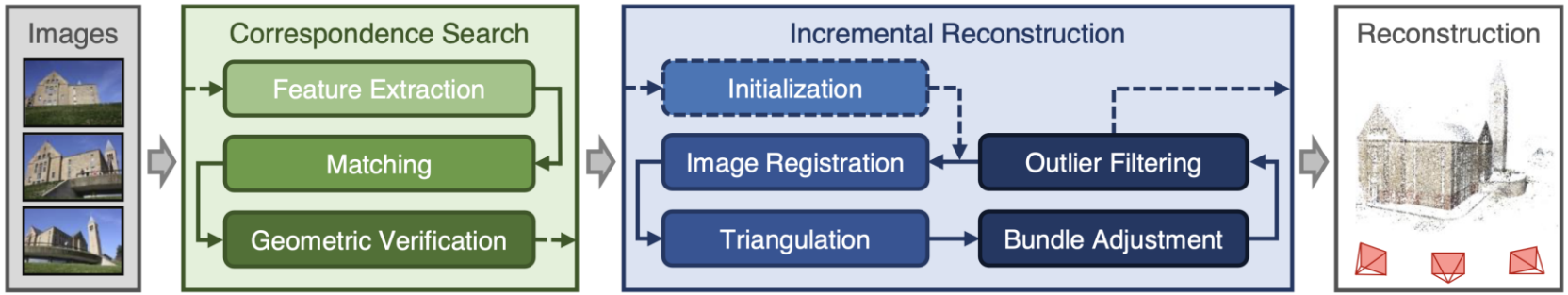


< MVS results >

DUSt3R: Geometric 3D Vision Made Easy [CVPR 24]

DUSt3R

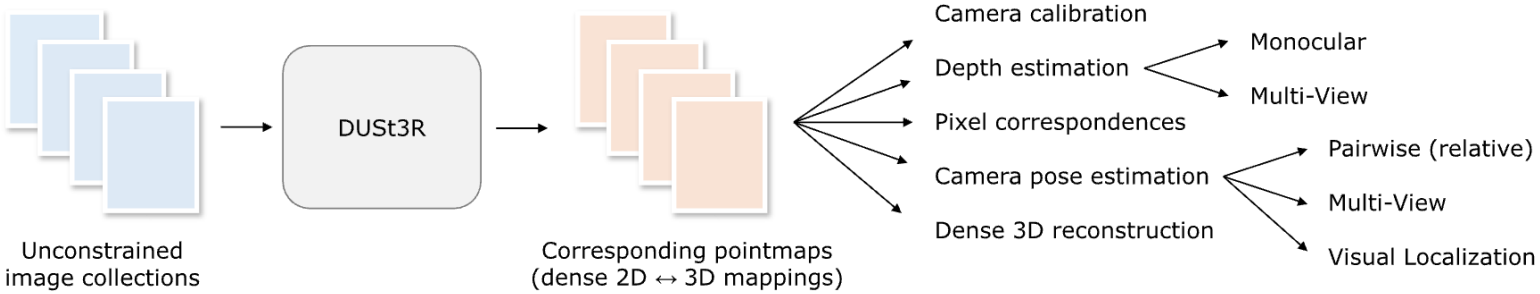
- Problem & Goal
  - Modern SfM and MVS pipelines
    - 해당 파이프라인은 3차원 복원을 위해 *minimal problem*들을 순차적으로 해결해 나감
      - Matching points, triangulating points, sparsely reconstruction and dense reconstruction
    - 이러한 복잡한 chain은 다음 단계로 거치면서 점점 노이즈가 추가되며, 복잡도를 높여 engineering effort가 많이 듦
    - 또한 단계가 단절로 인해 서로 도움을 주지 못하여 전체 과정이 실패할 수 있음
      - Dense 복원 시 sparse 복원을 고려하지 못함



< Structure from Motion of COLMAP >

DUSt3R

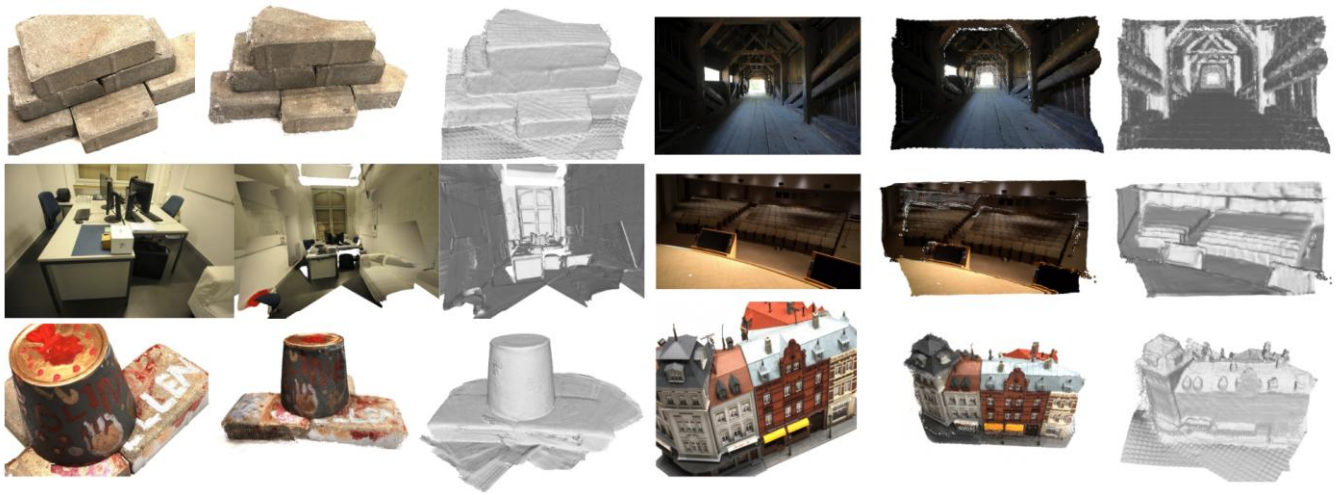
- Problem & Goal
  - Approach
    - *Pair* of images로부터 dense scene representation을 한번에 추정함
      - Point maps regression 문제로 변경하여, 카메라 파라미터 없이 직접적으로 포인트를 추정
    - 하나의 output만으로 모든 scene parameter를 직접적으로 추출할 수 있음
      - 이는 network가 input images와 결과 3D pointmaps를 joint하게 처리하여
      - 2D Structure의 3D shapes와 여러 minimal problem을 동시에 다뤄 'collaboration'이 가능함



< Overview of DUST3R >

DUSt3R

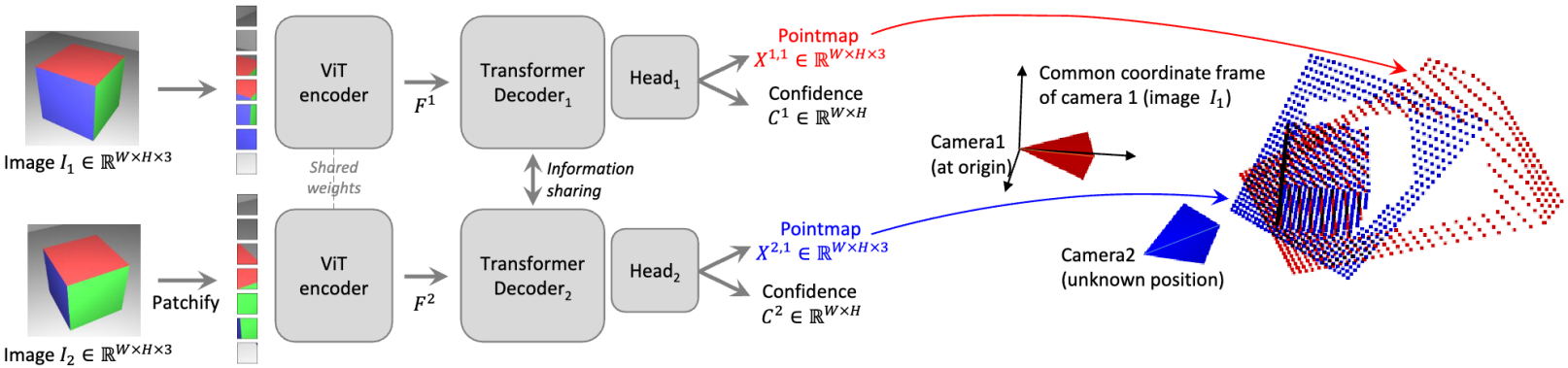
- Method
  - TLDR;
    - DUST3R는 fully-supervised 방식의 simple regression manner로 학습함
      - SfM softwares로 복원된 Large public datasets를 사용
    - Fully data-driven strategy를 기반으로 transformer architecture를 사용하고, geometric constraints 없는 end-to-end pipeline
    - 학습된 Network는 strong geometric and shape prior를 형성함



< Output of DUST3R >

DUSt3R

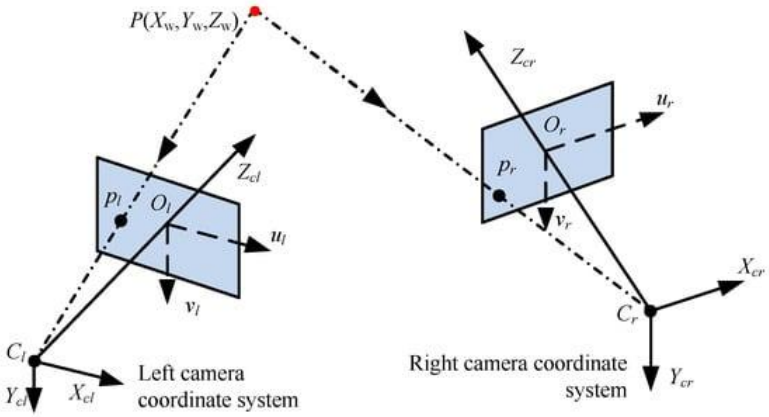
- Method
  - Essential Concept of pointmaps
    - Pointmap
      - 2D 이미지 내의 모든 픽셀을 3D 공간 상의 점 $X \in \mathbb{R}^{W \times H \times 3}$으로 표현함
      - RGB image I를 one-to-one mapping을 수행한 결과
        - ✓모든 픽셀 좌표계에 대해 $I_{i,j} \leftrightarrow X_{i,j}$ 매핑
      - 이를 위해 모든 카메라 ray는 하나의 표면만 만난다고 가정함



< Architecture of the network >

DUSt3R

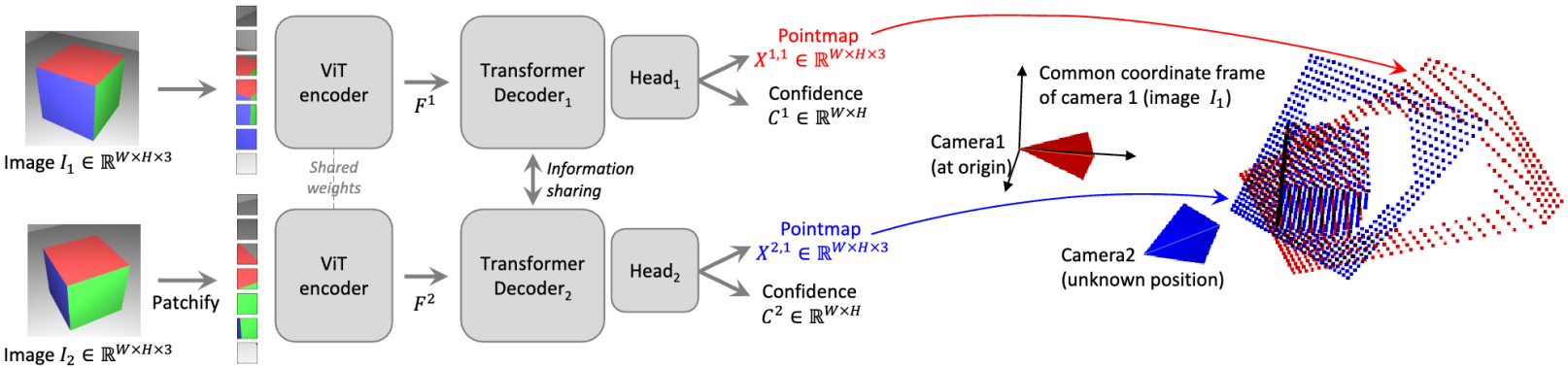
- Method
  - Essential Concept of pointmaps
    - Cameras and scene
      - Camera intrinsics $K \in \mathbb{R}^{3\times3}$이 주어지면, ground-truth depthmap $D \in \mathbb{R}^{W\times H}$로 pointmap X로 변환할 수 있음
        - ✓$X_{i,j} = K^{-1}[iD_{i,j}, jD_{i,j}, D_{i,j}]$
      - X는 camera coordinate로, camera n의 X^n을 camera m의 좌표계에서 $X^{n,m}$으로 변환할 수 있음
        - ✓$X^{n,m} = P_m P_n^{-1} h(X^n)$
        - ✓$P_m, P_n \in \mathbb{R}^{3\times4}$: image n과 m의 world-to-camera pose



< Camera coordinate system >

DUSt3R

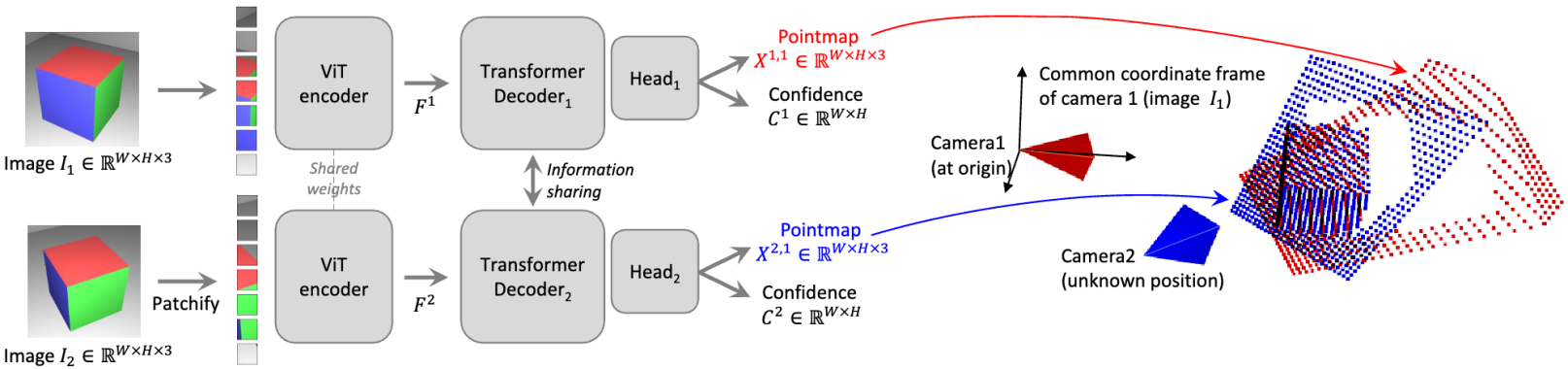
- Method
  - Overview
    - Direct regression을 통해 일반화된 3D reconstruction network를 만드는 것을 목표로 함
    - Network $\mathcal{F}$는 2 RGB images $I^1, I^2 \in \mathbb{R}^{W\times H\times 3}$을 입력으로 받아 2개의 대응하는 pointmaps $X^{1,1}, X^{2,1} \in \mathbb{R}^{W\times H\times 3}$을 출력함
      - 이때 pointmaps는 동일한 I^1의 camera coordinate에서 표현됨



< Architecture of the network >

DUSt3R

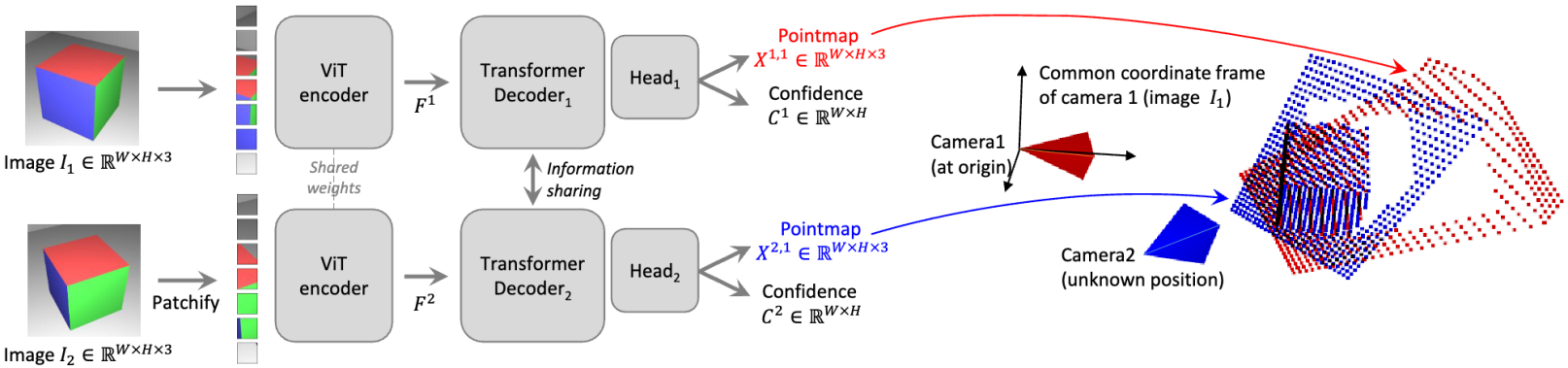
- Method
  - Network architecture
    - $\mathcal{F}$는 CroCo [2]의 pretrained model을 사용하고, image encoder와 decoder 그리고 regression head로 구성됨
    - 두 입력 이미지는 동일한 weight를 공유하는 ViT encoder로 token F_1, F_2로 각각 변환함
      - $F_1 = Encoder(I^1), F_2 = Encoder(I^2)$
      - 이후에 두 토큰을 decoder에서 jointly reasoning함



< Architecture of the network >

DUSt3R

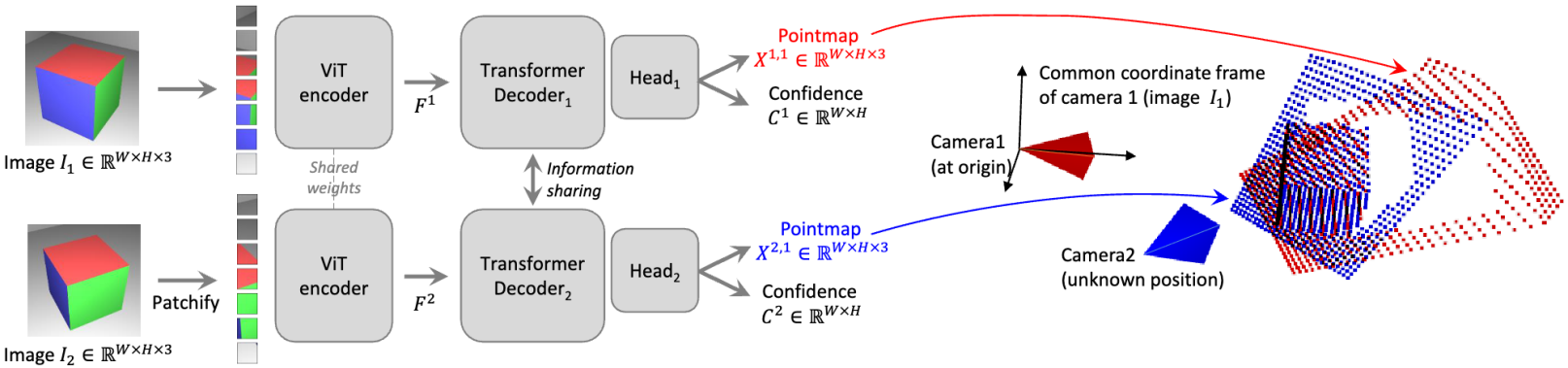
- Method
  - Network architecture
    - CroCo [2]와 같이 decoder는 cross attention이 포함된 generic transformer network로 구성됨
    - 개별 decoder block은 순차적으로 self-attention을 수행하고 cross attention을 거쳐 최종 MLP의 입력됨
      - Self-attention: 개별 view의 token끼리만 attention을 수행하여 이미지 내부의 정보 추출
      - Cross-attention: 두 이미지 사이를 attention을 통해 대응 관계를 학습함



< Architecture of the network >

DUSt3R

- Method
  - Network architecture
    - Decoder pass에서 두 branch의 정보를 일관되게 공유하도록 설계함
      - 이는 최종 출력의 pointmaps align에 영향을 미침
    - 개별 decoder block은 다른 branch의 token을 참조함
      - $G_i^1 = DecoderBlock_i^1(G_{i-1}^1, G_{i-1}^2)$,
      - $G_i^2 = DecoderBlock_i^2(G_{i-1}^2, G_{i-1}^1)$,
        - ✓$i = 1, \ldots, B$의 B개 block이며, G_0^1, G_0^2는 F_1, F_2로 초기화 됨
        - ✓$DecoderBlock_i^v(G^1, G^2)$는 branch $v \in \{1,2\}$의 i번째 Decoder Block을 나타냄



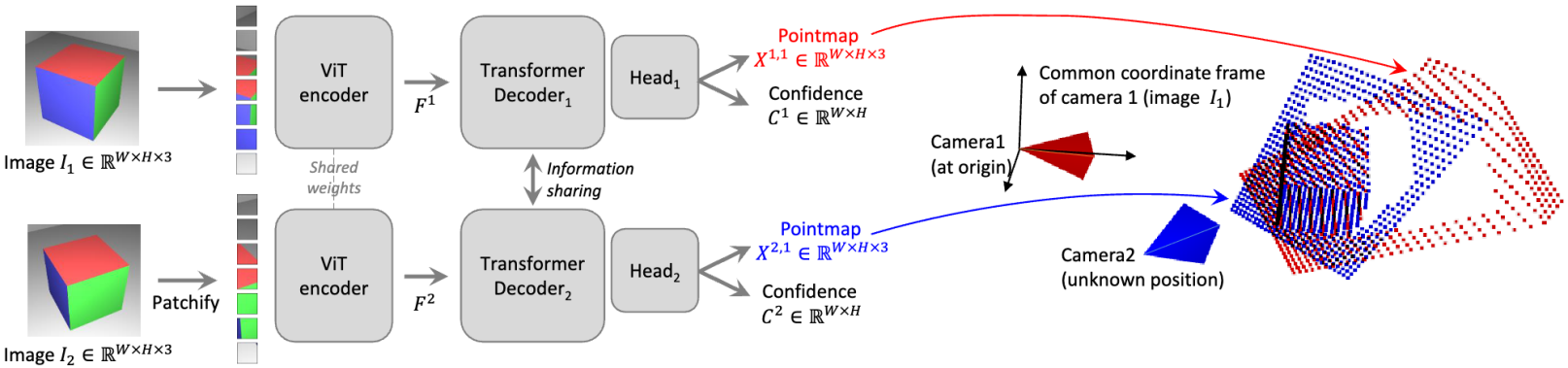
< Architecture of the network >

DUSt3R

- Method
  - Network architecture
    - 최종적으로 개별 branch는 각각의 regression head에 decoder tokens를 전달하여 pointmap과 confidence map을 출력함

$$X^{1,1}, C^{1,1} = Head^1\left(G_0^1, \ldots, G_B^1\right),$$

$$X^{2,1}, C^{2,1} = Head^2\left(G_0^2, \ldots, G_B^2\right),$$

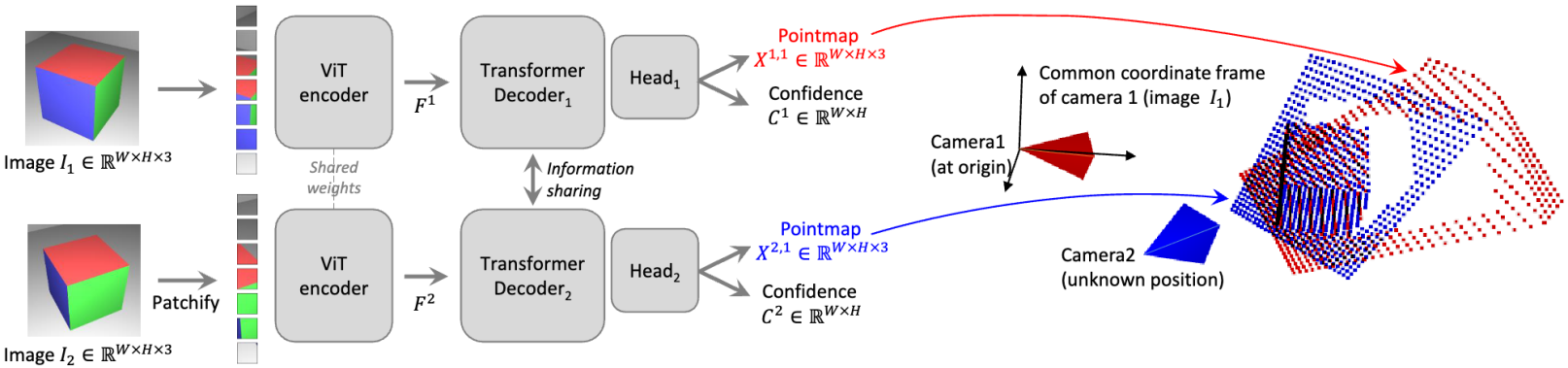


< Architecture of the network >

DUSt3R

- Method
  - Training Objective
    - 3D Regression loss.
      - 단 하나의 training objective는 3D space에서 regression을 기반으로 함
      - Ground-truth pointmaps $\bar{X}^{1,1}, \bar{X}^{2,1}$가 주어졌을 때, valid pixel $\mathcal{D}^1, \mathcal{D}^2 \subseteq \{1 \dots W\} \times \{1 \dots H\}$에 대한 regression loss를 Euclidean distance로 정의함

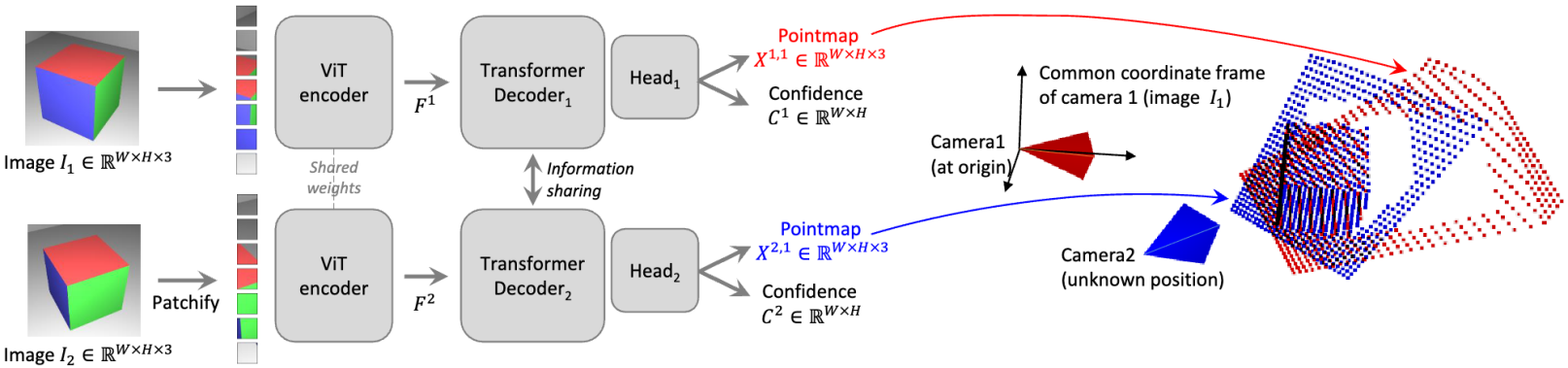
$$\checkmark \ell_{regr}(v,i) = \left| \left| \frac{1}{z} X_i^{v,1} - \frac{1}{z} \bar{X}_i^{v,1} \right| \right|$$



< Architecture of the network >

DUSt3R

- Method
  - Training Objective
    - Confidence-aware loss.
      - 이전의 가정과 달리, 실제 세계는 하늘이나 투명한 물체의 ill-defined 3D points가 존재함
        - ✓특정 이미지의 영역 별로 예측 난이도가 다름
      - 따라서 network는 개별 픽셀에 대한 confidence를 출력하도록 함께 학습됨
      - 최종 training objective는 confidence-weighted regression loss로 구성됨
        - ✓$\mathcal{L}_{conf} = \sum_{v\in\{1,2\}}\sum_{i\in D_v} C_i^{v,1}\ell_{regr}(v,i) - \alpha log C_i^{v,1}$
          - $C_i^{v,1}$: confidence score for pixel i, α: hyper parameter



< Architecture of the network >

DUSt3R

- Method
  - Global Alignment
    - Pair 이상의 이미지를 복원하기 위해 post-processing optimization으로 joint 3D space에 alignment함
    - Pairwise graph.
      - 특정 장면에 대한 Image set $\{I^1, I^2, \ldots, I^N\}$이 주어졌을 때, connectivity graph $\mathcal{G}(v, \varepsilon)$를 추정함
        - ✓N개의 image에 대한 vertices v와 edge $e = (n, m) \in \varepsilon$ 으로 구성됨
      - 기존의 retrieval methods나 $\mathcal{F}$를 사용하여 low-confidence pair를 필터링하여 추정함



< Reconstruction Examples >

DUSt3R

- Method
  - Global Alignment
    - Global optimization.
      - Connectivity graph $\mathcal{G}$를 기반으로, 전체 N개의 카메라에 대한 pointmaps $\{X^n \in \mathbb{R}^{W\times H\times 3}\}$을 공통된 world 좌표계로 정렬함
      - 모든 image pair $e = (n, m) \in \varepsilon$에서, world 좌표계로 변환하는 pairwise pose $P_e \in \mathbb{R}^{3\times 4}$와 scailing $\sigma_e > 0$를 도입하고 이를 전역적으로 최적화함
        - ✓$X^* = argmin_{X,P,\sigma} \sum_{e\in\varepsilon} \sum_{v\in e} \sum_{i=1}^{HW} C_i^{v,e} \left\| X_i^v - \sigma_e P_e X_i^{v,e} \right\|$
        - ✓동일한 이미지가 포함된 pair는 동일한 world 좌표계를 갖도록 최적화함



< Reconstruction Examples >

DUSt3R

- Experiments
  - Implementation Details
    - Training data.
      - 8개의 datasets를 섞어 network를 학습함
        - ✓Habitat, MegaDepth, ARKitScenes, MegaDepth, Static Scenes 3D, Blended MVS, ScanNet++, CO3D-v2 and Waymo
        - ✓Indoor, outdoor, synthetic, real-world, object-centric 등 다양한 scene type으로 구성됨
        - ✓8.5M개의 pair 데이터로 학습함



< Habitat Examples >

DUSt3R

- Experiments
  - Quantitative Results
    - Multi-view Pose Estimation

	Methods	7Scenes (Indoor) [114]							Cambridge (Outdoor) [49]				
		Chess	Fire	Heads	Office	Pumpkin	Kitchen	Stairs	S. Facade	O. Hospital	K. College	St.Mary's	G. Court
FM	AS [103]	4/1.96	3/1.53	2/1.45	9/3.61	8/3.10	7/3.37	**3**/2.22	**4**/0.21	20/0.36	13/0.22	8/0.25	24/**0.13**
FM	HLoc [101]	**2/0.79**	**2/0.87**	**2/0.92**	**3/0.91**	**5/1.12**	**4/1.25**	6/**1.62**	**4/0.2**	**15/0.3**	**12/0.20**	**7/0.21**	**11**/0.16
E2E	DSAC* [11]	**2**/1.10	**2**/1.24	**1**/1.82	**3**/1.15	**4**/1.34	4/1.68	**3**/1.16	**5**/0.3	**15/0.3**	15/0.3	13/0.4	49/0.3
E2E	HSCNet [55]	**2/0.7**	**2**/0.9	**1**/0.9	**3/0.8**	**4/1.0**	4/**1.2**	**3/0.8**	6/0.3	19/**0.3**	18/0.3	9/0.3	28/0.2
E2E	PixLoc [102]	**2**/0/80	**2/0.73**	**1/0.82**	**3**/0.82	**4**/1.21	**3/1.20**	5/1.30	**5/0.23**	16/0.32	14/0.24	10/0.34	30/0.14
E2E	SC-wLS [152]	3/0.76	5/1.09	3/1.92	6/0.86	8/1.27	9/1.43	12/2.80	11/0.7	42/1.7	14/0.6	39/1.3	164/0.9
E2E	NeuMaps [125]	**2**/0.81	3/1.11	2/1.17	**3**/0.98	**4**/1.11	4/1.33	4/1.12	6/0.25	19/0.36	14/0.19	17/0.53	**6/ 0.10**
	DUSt3R 224-NoCroCo	5/1.76	6/2.02	3/1.75	5/1.54	9/2.35	6/1.82	34/7.81	24/1.33	79/1.17	69/1.15	46/1.51	143/1.32
	DUSt3R 224	3/0.96	3/1.02	**1**/1.00	4/1.04	5/1.26	4/1.36	21/4.08	9/0.38	26/0.46	20/0.32	11/0.38	36/0.24
	DUSt3R 512	3/0.97	3/0.95	2/1.37	**3**/1.01	**4**/1.14	4/1.34	11/2.84	6/0.26	17/0.33	**11/0.20**	**7/0.24**	38/0.16

< Camera pose on 7scenes >

DUSt3R

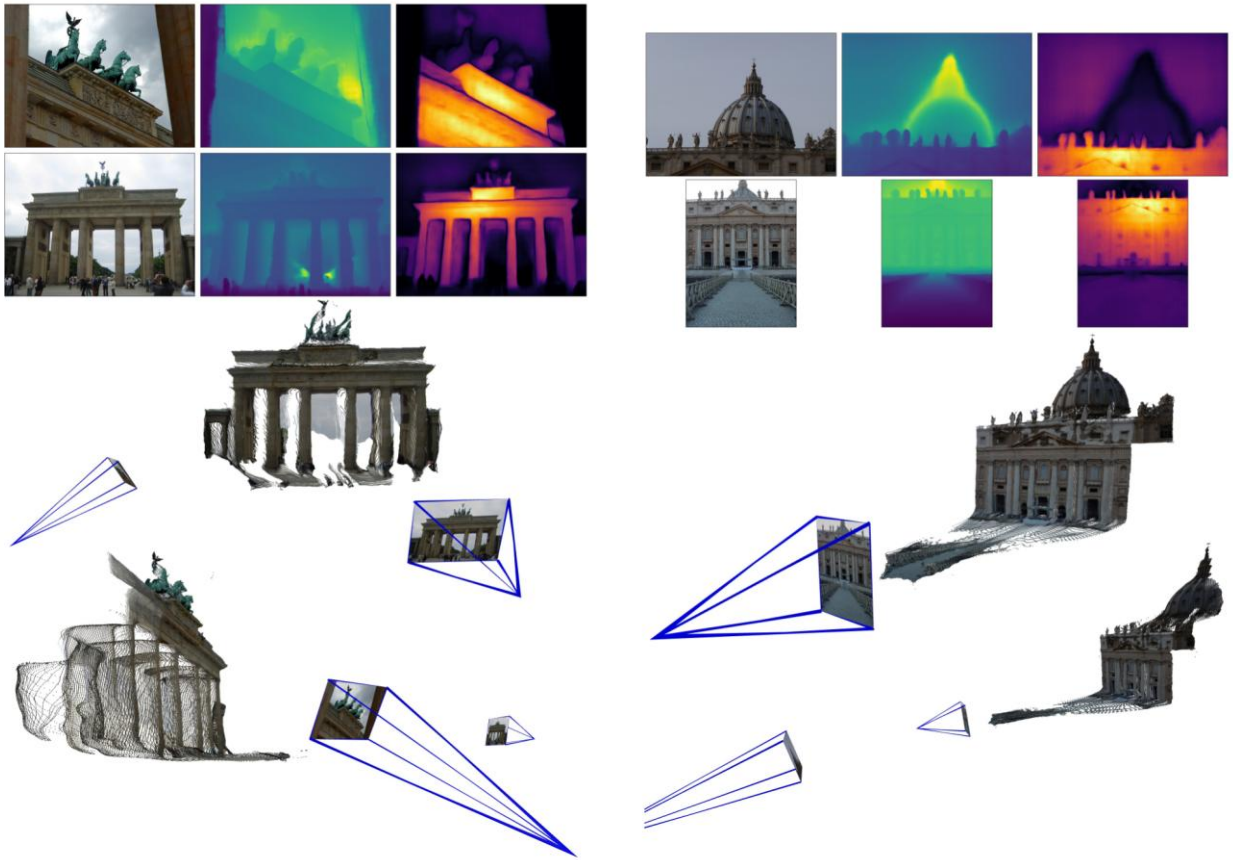
- Experiments
  - Quantitative Results
    - Multiview Depth

	Methods	GT Pose	GT Range	GT Intrinsics	Align	KITTI rel ↓	KITTI τ ↑	ScanNet rel ↓	ScanNet τ ↑	ETH3D rel ↓	ETH3D τ ↑	DTU rel ↓	DTU τ ↑	T&T rel↓	T&T τ ↑	Average rel↓	Average τ ↑	Average time (s)↓
(a)	COLMAP [106, 107]	✓	×	✓	×	**12.0**	**58.2**	**14.6**	**34.2**	**16.4**	**55.1**	**0.7**	**96.5**	**2.7**	95.0	**9.3**	**67.8**	≈ 3 min
	COLMAP Dense [106, 107]	✓	×	✓	×	26.9	52.7	38.0	22.5	89.8	23.2	20.8	69.3	25.7	76.4	40.2	48.8	≈ 3 min
(b)	MVSNet [161]	✓	✓	✓	×	22.7	36.1	24.6	20.4	35.4	31.4	(1.8)	(86.0)	8.3	73.0	18.6	49.4	0.07
	MVSNet Inv. Depth [161]	✓	✓	✓	×	18.6	30.7	22.7	20.9	21.6	35.6	(1.8)	(86.7)	6.5	74.6	14.2	49.7	0.32
	Vis-MVSSNet [176]	✓	✓	✓	×	**9.5**	**55.4**	8.9	33.5	**10.8**	**43.3**	**(1.8)**	**(87.4)**	**4.1**	**87.2**	**7.0**	**61.4**	0.70
	MVS2D ScanNet [160]	✓	✓	✓	×	21.2	8.7	(27.2)	(5.3)	27.4	4.8	17.2	9.8	29.2	4.4	24.4	6.6	**0.04**
	MVS2D DTU [160]	✓	✓	✓	×	226.6	0.7	32.3	11.1	99.0	11.6	(3.6)	(64.2)	25.8	28.0	77.5	23.1	0.05
(c)	DeMon [136]	✓	×	✓	×	16.7	13.4	75.0	0.0	19.0	16.2	23.7	11.5	17.6	18.3	30.4	11.9	0.08
	DeepV2D KITTI [131]	✓	×	✓	×	(20.4)	(16.3)	25.8	8.1	30.1	9.4	24.6	8.2	38.5	9.6	27.9	10.3	1.43
	DeepV2D ScanNet [131]	✓	×	✓	×	61.9	5.2	(3.8)	(60.2)	18.7	28.7	9.2	27.4	33.5	38.0	25.4	31.9	2.15
	MVSNet [161]	✓	×	✓	×	14.0	35.8	1568.0	5.7	507.7	8.3	(4429.1)	(0.1)	118.2	50.7	1327.4	20.1	0.15
	MVSNet Inv. Depth [161]	✓	×	✓	×	29.6	8.1	65.2	28.5	60.3	5.8	(28.7)	(48.9)	51.4	14.6	47.0	21.2	0.28
	Vis-MVSNet [176]	✓	×	✓	×	10.3	**54.4**	84.9	15.6	51.5	17.4	(374.2)	(1.7)	21.1	65.6	108.4	31.0	0.82
	MVS2D ScanNet [160]	✓	×	✓	×	73.4	0.0	(4.5)	(54.1)	30.7	14.4	5.0	57.9	56.4	11.1	34.0	27.5	**0.05**
	MVS2D DTU [160]	✓	×	✓	×	93.3	0.0	51.5	1.6	78.0	0.0	(1.6)	(92.3)	87.5	0.0	62.4	18.8	0.06
	Robust MVD Baseline [110]	✓	×	✓	×	**7.1**	41.9	**7.4**	**38.4**	**9.0**	**42.6**	**2.7**	**82.0**	**5.0**	**75.1**	**6.3**	**56.0**	0.06
(d)	DeMoN [136]	×	×	✓	$\|\|\mathbf{t}\|\|$	15.5	15.2	12.0	21.0	17.4	15.4	21.8	16.6	13.0	23.2	16.0	18.3	0.08
	DeepV2D KITTI [131]	×	×	✓	med	(3.1)	(74.9)	23.7	11.1	27.1	10.1	24.8	8.1	34.1	9.1	22.6	22.7	2.07
	DeepV2D ScanNet [131]	×	×	✓	med	10.0	36.2	**(4.4)**	(54.8)	11.8	29.3	7.7	33.0	8.9	46.4	8.6	39.9	3.57
	DUSt3R 224-NoCroCo	×	×	×	med	15.14	21.16	7.54	40.00	9.51	40.07	3.56	62.83	11.12	37.90	9.37	40.39	**0.05**
	DUSt3R 224	×	×	×	med	15.39	26.69	(5.86)	(50.84)	4.71	61.74	**2.76**	**77.32**	5.54	56.38	6.85	54.59	**0.05**
	DUSt3R 512	×	×	×	med	**9.11**	**39.49**	(4.93)	**(60.20)**	**2.91**	**76.91**	3.52	69.33	**3.17**	**76.68**	**4.73**	**64.52**	0.13

< Multiview depth estimations >

DUSt3R

- Experiments
  - Qualitative Results
    - 3D Reconstruction



VGGT: Visual Geometry Grounded Transformer [CVPR 2025 Best Paper Award]

VGGT

- Goal
  - Feed-forward neural network
    - 기존의 Feed forward network인 DUSt3R와 MASt3R 의 pair-wise 한계를 극복
      - 더 많은 이미지를 처리하기 위해서는 post-processing 과정이 필요함
    - 여러 view의 이미지로부터 3D attribute들을 추정가능한 모델
      - Camera parameter, point map, depth map, 3d point tracks
      - DUSt3R의 추가적인 optimization based가 필요없음



< Goal of VGGT >

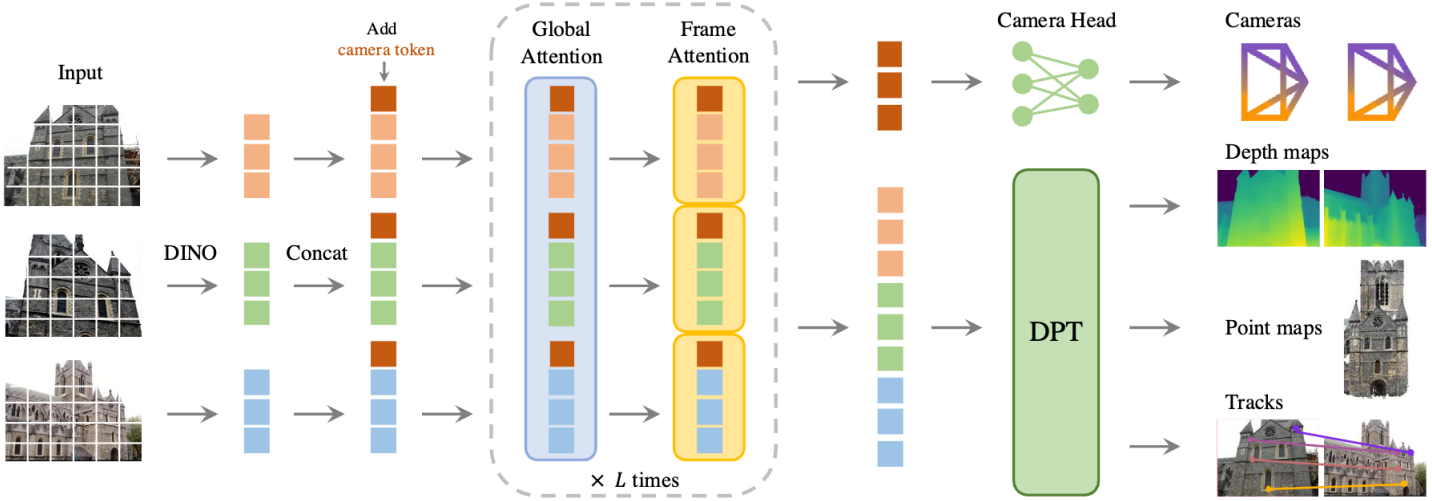
VGGT

- Method
  - Problem definition and notation
    - N개의 RGB image sequence를 입력받아, VGGT의 transformer는 3D annotation들로 매핑하는 역할을 수행함

$\therefore f\left((I_i)_{i=1}^N\right) = (g_i, D_i, P_i, T_i)_{i=1}^N$

✓ $g_i \in \mathbb{R}^9$: camera parameter, $D_i \in \mathbb{R}^{H\times W}$: depth map

✓ $P_i \in \mathbb{R}^{3\times H\times W}$, $T_i \in \mathbb{R}^{C\times H\times W}$: feature for point-tracking

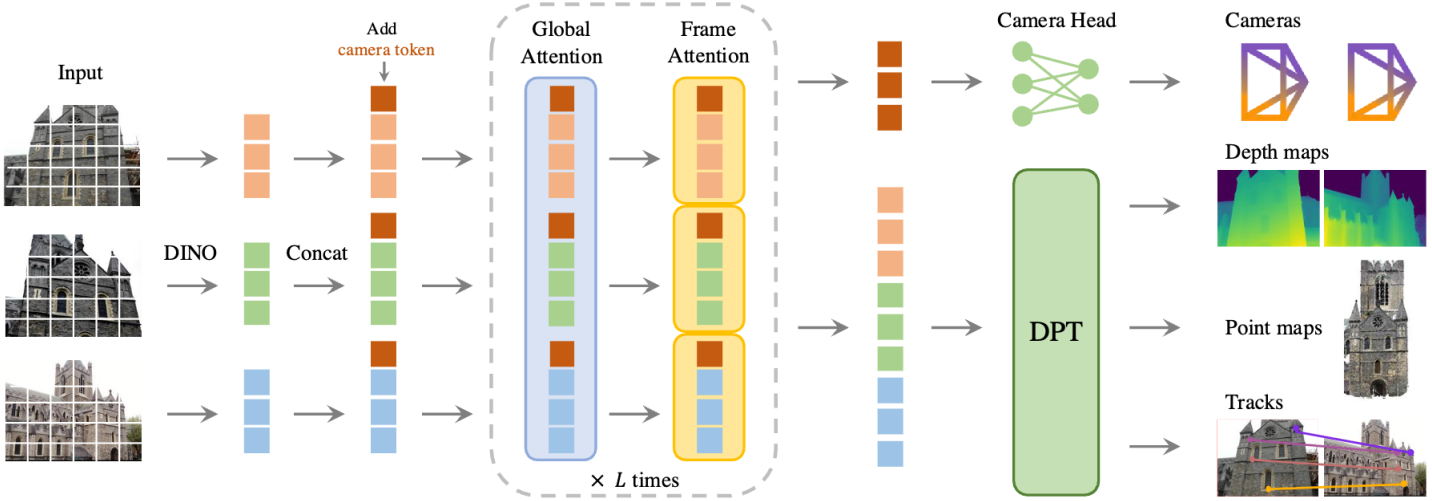


< Architecture Overview >

부산대학교 PUSAN NATIONAL UNIVERSITY
CVSP Lab Computer Vision & Signal Processing

VGGT

- Method
  - Problem definition and notation
    - Order of Predictions
      - 입력 이미지 셋의 첫번째 이미지는 reference frame으로 사용되며, 나머지는 임의적으로 permutation equivariant함
    - Over-complete Predictions
      - VGGT의 여러 3D quantities를 독립적으로 예측하지만, 이들은 이론적으로 서로 유도 가능함
      - 하지만 여러 quantities를 explicitly 추정하는 것이 전체 성능을 향상시킴

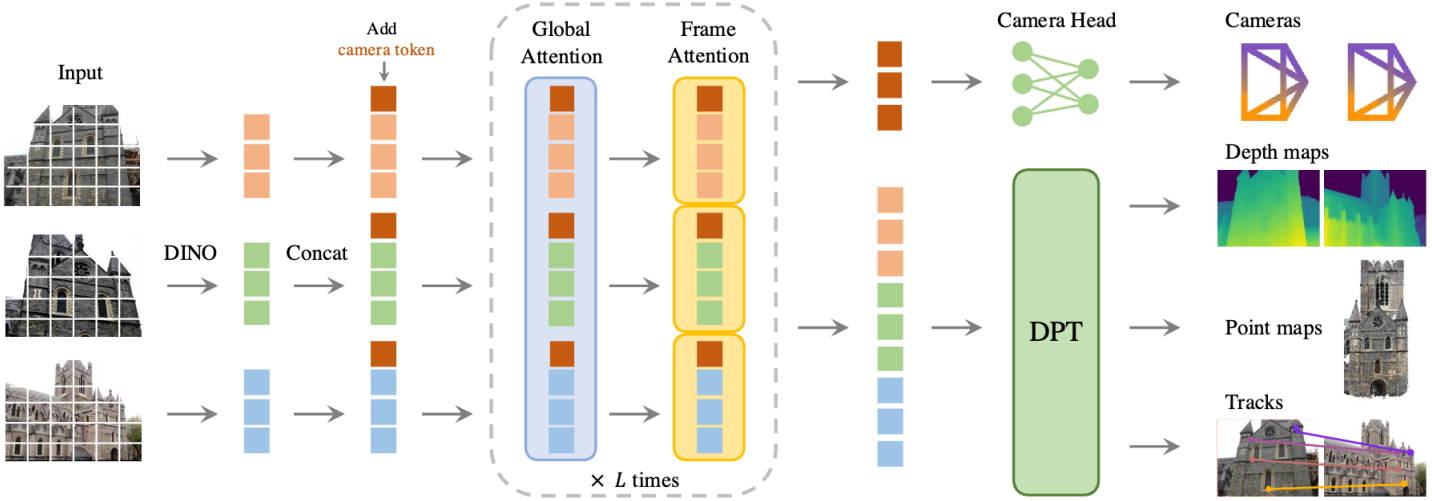


< Architecture Overview >

부산대학교 PUSAN NATIONAL UNIVERSITY
CVSP Lab Computer Vision & Signal Processing

VGGT

- Method
  - Feature Backbone
    - 최근 3D deep learning을 따라 모델에 3D inductive biases를 최소화하고, 데이터로부터 스스로 학습되도록 함
      - 입력 이미지 I를 DINO [3]를 통해 K개의 토큰 $t^I \in \mathbb{R}^{K \times C}$으로 변환함
      - 모든 프레임에 대한 토큰을 $t^I = \bigcup_{i=1}^{N} \{t_i^I\}$로 결합한 뒤, main network에 입력하여 frame-wise와 global self-attention layers를 번갈아 적용함

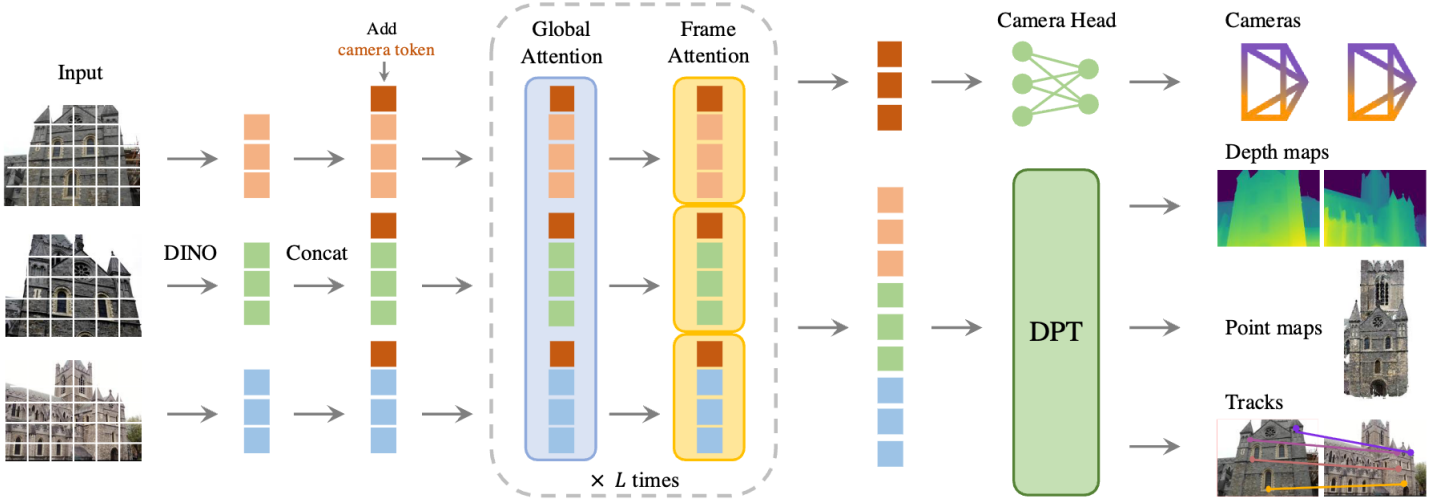


< Architecture Overview >

부산대학교 PUSAN NATIONAL UNIVERSITY
CVSP Lab Computer Vision & Signal Processing

VGGT

- Method
  - Feature Backbone
    - Alternating-Attention
      - Standard transformer 구조에서 Alternating-Attention(AA)를 통해 개별 frame과 global을 처리함
        - ✓Frame-wise self-attention은 tokens t_k^I의 개별 프레임을 독립적으로 처리함
        - ✓Global self-attention은 전체 프레임의 토큰 t^I를 joint하게 처리함
        - ✓이를 통해 여러 이미지의 정보를 통합하고, 개별 이미지의 내부 구조를 다시 강화함

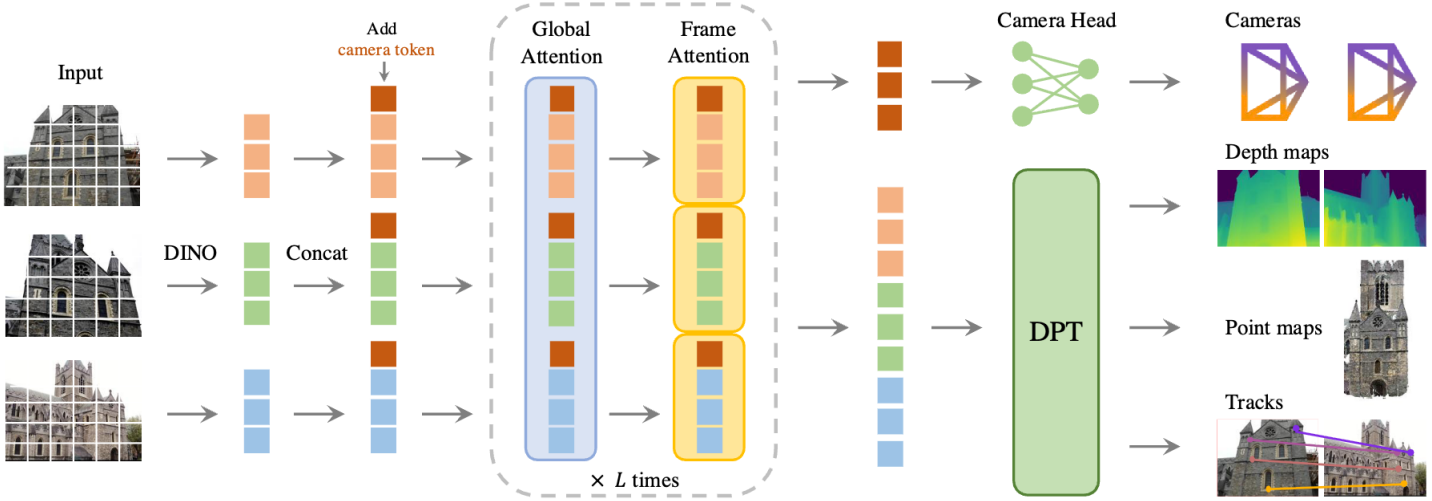


< Architecture Overview >

부산대학교 PUSAN NATIONAL UNIVERSITY
CVSP Lab Computer Vision & Signal Processing

VGGT

- Method
  - Prediction heads
    - 각 이미지 토큰에 camera token과 register tokens 을 추가하여 이미지별 3D 정보를 담음
      - 개별 이미지 토큰 t_i^I에 camera token $t_i^g \in \mathbb{R}^{1\times C'}$와 4개의 register tokens $t_i^R \in \mathbb{R}^{4\times C'}$를 추가함
    - 첫 프레임을 기준 좌표계로 설정하기 위해, 첫 프레임의 camera token과 register token만 다르게 초기화함
      - Camera token은 pose 관련한 정보를 요약, Register token은 복잡한 3D 정보를 담기 위한 보조 토큰으로 활용됨

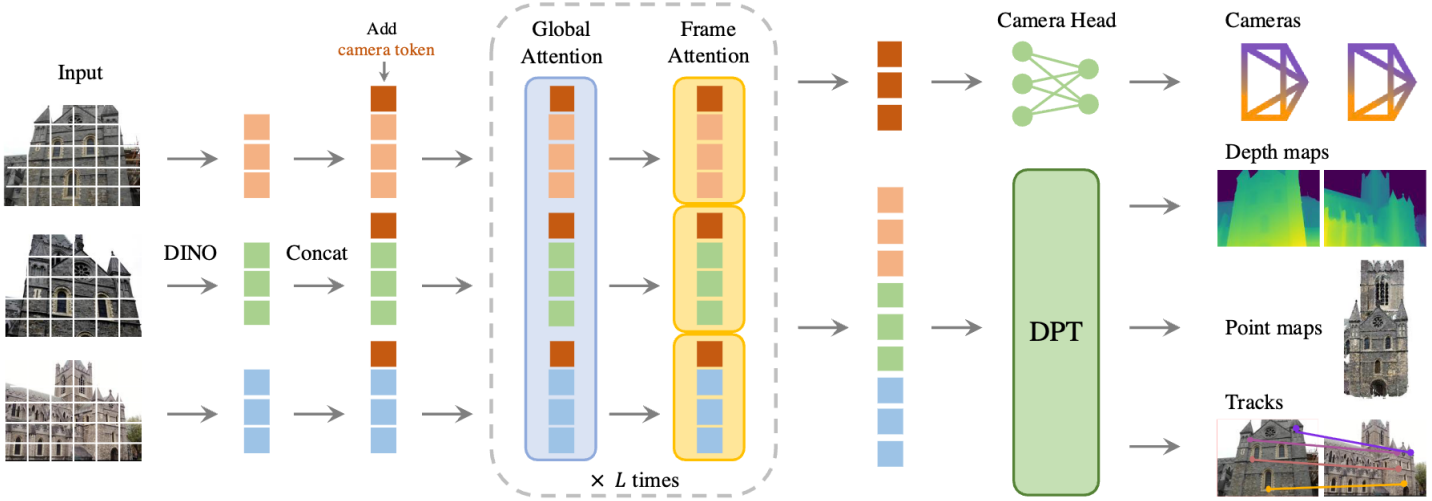


< Architecture Overview >

부산대학교 PUSAN NATIONAL UNIVERSITY
CVSP Lab Computer Vision & Signal Processing

VGGT

- Method
  - Prediction heads
    - 결합된 토큰들을 AA transformer에 전달하여 output tokens $\left(\hat{t}_i^I, \hat{t}_i^g, t_i^R\right)_{i=1}^N$를 출력함
      - 출력된 camera와 register token은 AA transformer의 frame-wise self-attention layer로 인해 frame-specific한 정보를 담게됨
    - 출력 토큰 중 t_i^R은 버리고, 나머지 토큰만 추정에 활용함

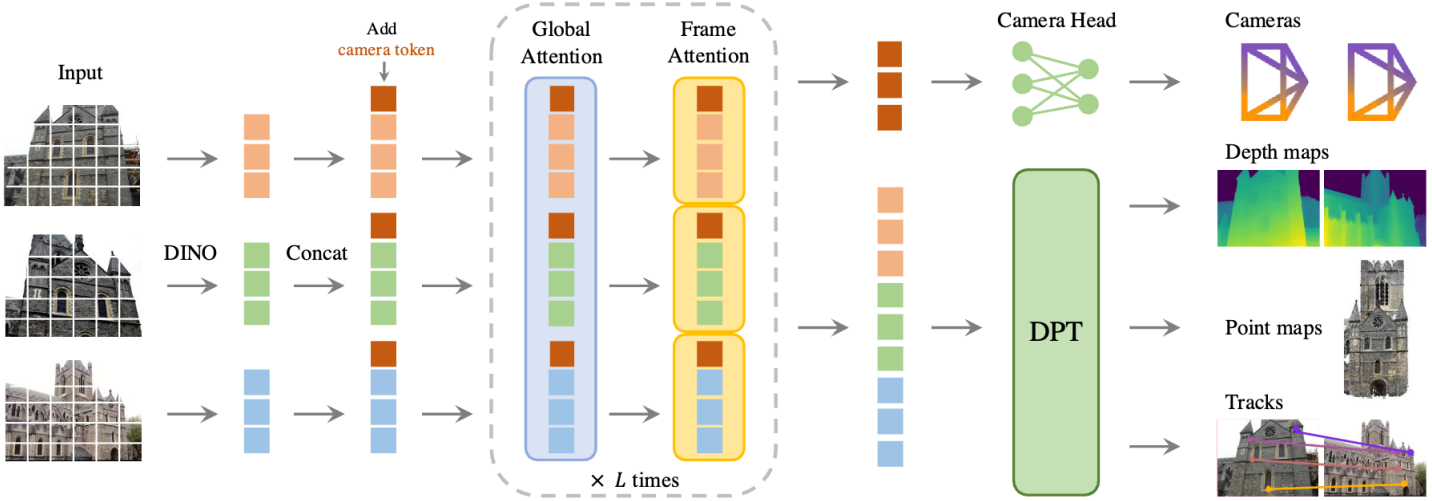


< Architecture Overview >

부산대학교 PUSAN NATIONAL UNIVERSITY
CVSP Lab Computer Vision & Signal Processing

VGGT

- Method
  - Prediction heads
    - Coordinate Frame.
      - 모델은 첫번째 camera g_1의 좌표계에서 모든 카메라를 추정함
      - 따라서 첫 카메라는 extrinsic은 Identity임
        - ✓First quaternion $q_1 = [0,0,0,1]$, translation vector $t_1 = [0,0,0]$
        - ✓다르게 초기화된 camera와 register token을 통해 첫번째 카메라를 모델이 찾을 수 있음

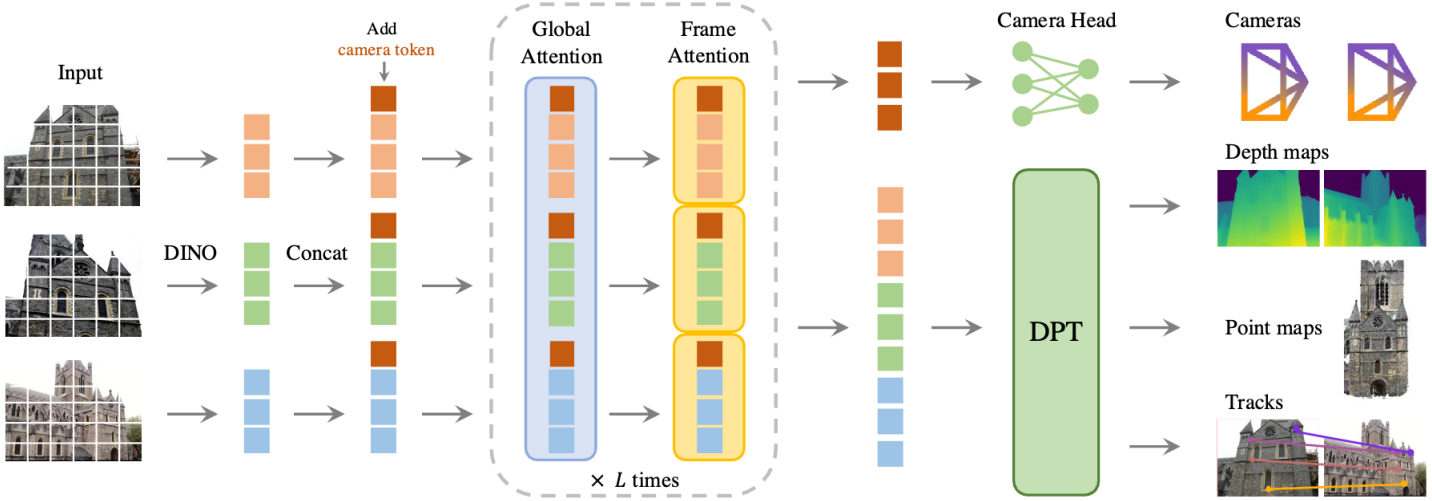


< Architecture Overview >

부산대학교 PUSAN NATIONAL UNIVERSITY
CVSP Lab Computer Vision & Signal Processing

VGGT

- Method
  - Prediction heads
    - Camera Predictions.
      - Camera parameter $(g_1)_{i=1}^N$은 camera token $(\hat{t}_i^g)_{i=1}^N$을 추가적인 layer를 거쳐 추정함
        - ✓Camera Head를 통해 camera intrinsic과 extrinsic를 추정

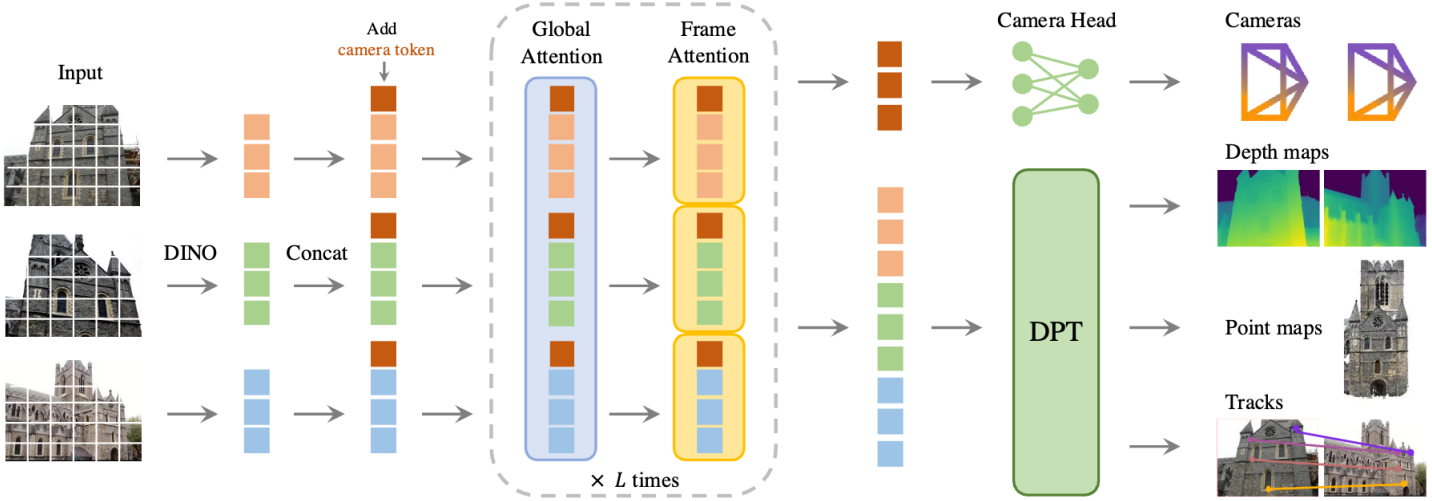


< Architecture Overview >

부산대학교 PUSAN NATIONAL UNIVERSITY
CVSP Lab Computer Vision & Signal Processing

VGGT

- Method
  - Prediction heads
    - Dense Predictions.
      - 이미지 토큰 $\hat{t}_i^I$는 dense output을 추정하는 데 사용됨
        - ✓Depth maps D_i, point maps P_i
      - 구체적으로 $\hat{t}_i^I$는 DPT Layer를 거쳐 feature maps $F_i \in \mathbb{R}^{C'' \times H \times W}$로 변환됨
      - 이후 3×3 convolutional laye를 통해 대응하는 depth와 point maps를 예측함



< Architecture Overview >

부산대학교 PUSAN NATIONAL UNIVERSITY
CVSP Lab Computer Vision & Signal Processing

VGGT

- Method
  - Training Losses
    - VGGT model은 end-to-end 방식의 multi-task loss로 학습함

$\mathcal{L} = \mathcal{L}_{camera} + \mathcal{L}_{depth} + \mathcal{L}_{pmap}$

✓ $\mathcal{L}_{camera} = \sum_{i=1}^{N} ||\hat{g}_i - g_i||$

✓ $\mathcal{L}_{depth} = \sum_{i=1}^{N} \left|\left|\Sigma_i^D \odot (\hat{D}_i - D_i)\right|\right| + \left|\left|\Sigma_i^D \odot (\nabla\hat{D}_i - \nabla D_i)\right|\right| - \alpha log\, \Sigma_i^D$ _

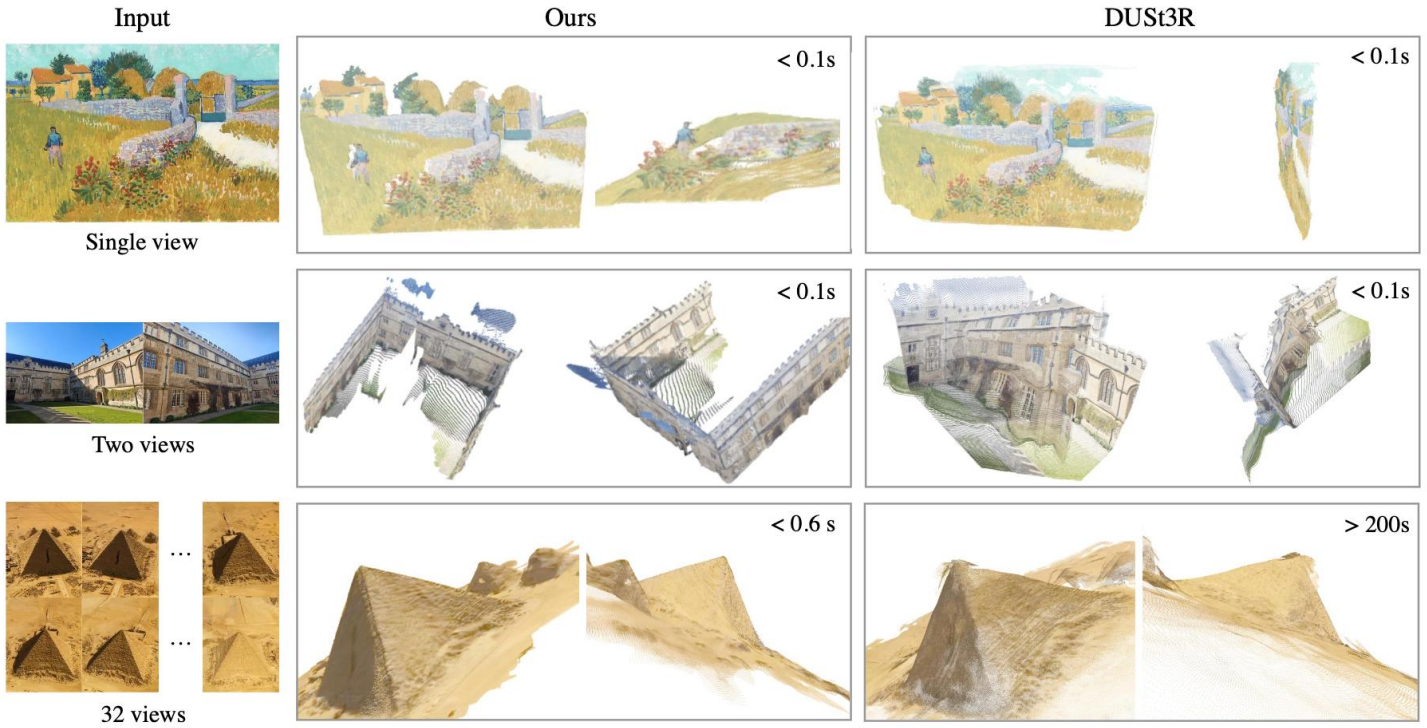
- Σ_i^D _: uncertainty map
- $\odot$: channel-broadcast element-wise product

✓ $\mathcal{L}_{pmap} = \sum_{i=1}^{N} \left|\left|\Sigma_i^P \odot (\hat{P}_i - P_i)\right|\right| + \left|\left|\Sigma_i^P \odot (\nabla\hat{P}_i - \nabla P_i)\right|\right| - \alpha log\, \Sigma_i^P$ _

세 개의 Loss는 비슷한 범위를 가져 weight를 사용하지 않음

VGGT

- Experiments



< Pointmap Results >

Next Research

- Real to Sim.



< Monocular Video Input >



< Human + Scene Reconstruction >



< Training in Simulation >



< Real-World Demo >

Next Research

- Hand & Scene Reconstruction

