VGGT: Visual Geometry Grounded Transformer

Jianyuan Wang[1,2] Minghao Chen[1,2] Nikita Karaev[1,2] Andrea Vedaldi[1,2]

Christian Rupprecht[1] David Novotny[2]

[1]Visual Geometry Group, University of Oxford [2]Meta AI

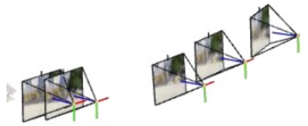
Background

Task Definition: 3D Reconstruction from input 2D images
Depth Estimation, Camera Parameter, 3D Tracking ...

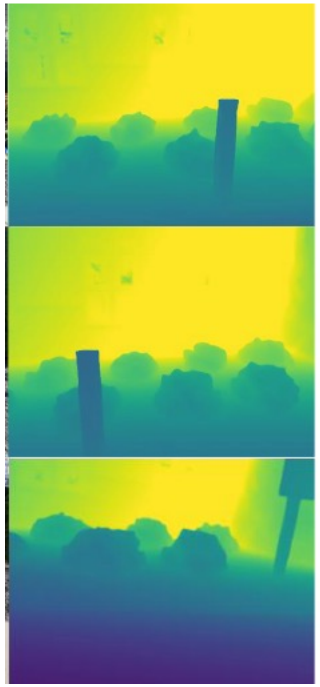
Input images

method

Reconstructed pcd



Estimated camera parameters



Estimated depth

Traditional Methods: Feature matching, polar geometry, bundle adjustment...

Time consuming, not end-to-end...

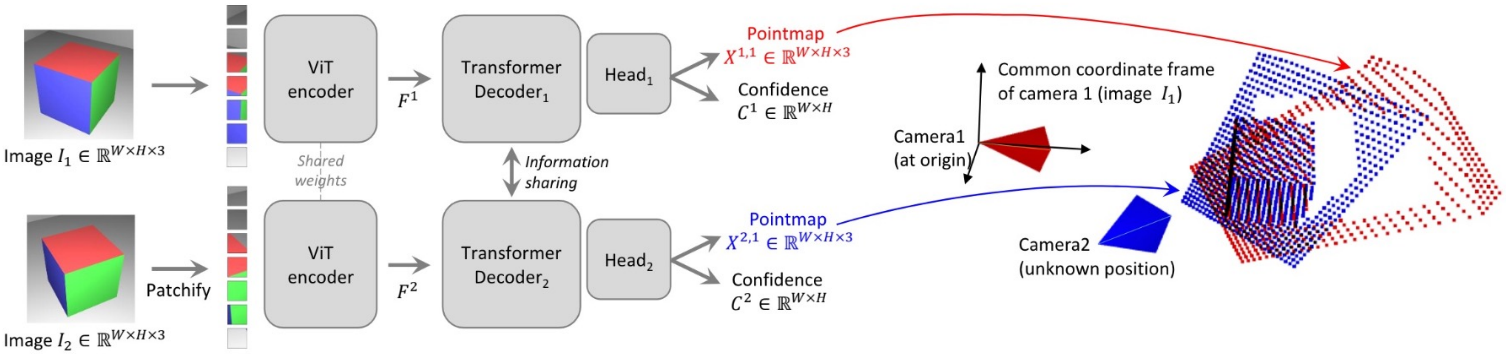
Background&Motivation

Recent Methods: Solve specific task with the help of powerful networks

DepthAnything series:

Depth Estimation only

Dust3r:



Need post processing when processing more than 2 images

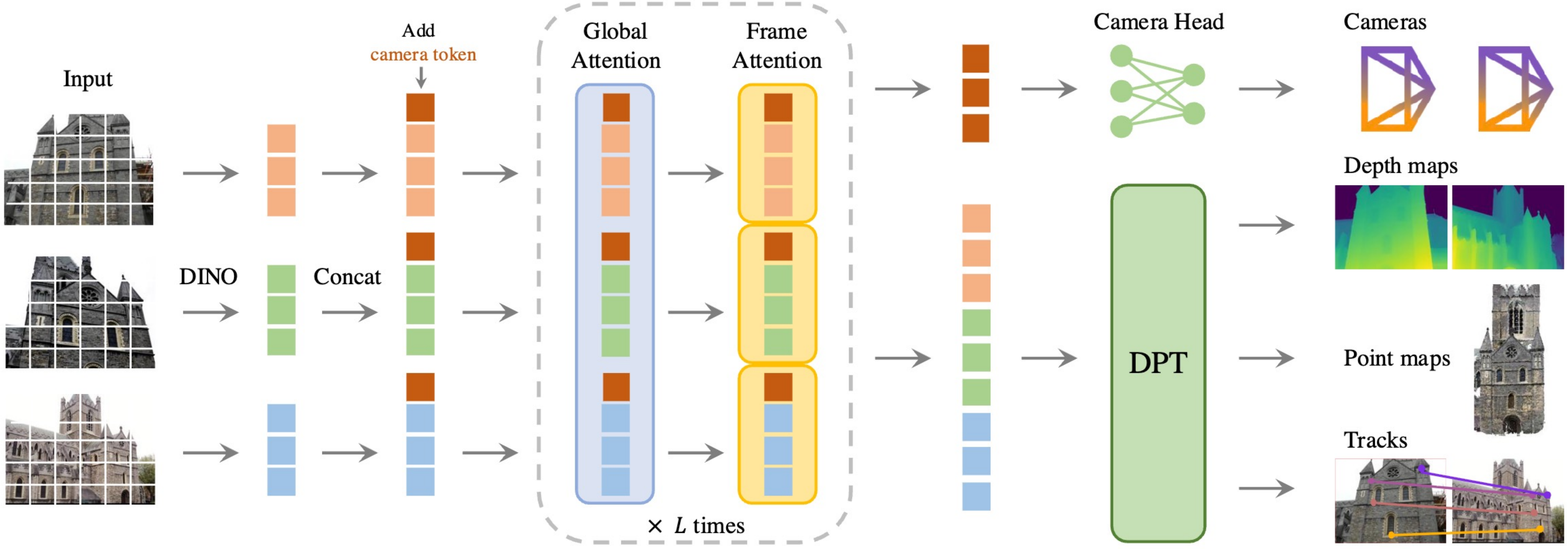
Motivation: Train a powerful network to solve all tasks with a single forward pass

Pipeline

Task Definition: Input 2D images, directly output camera parameters, depth maps, point maps and features for tracking

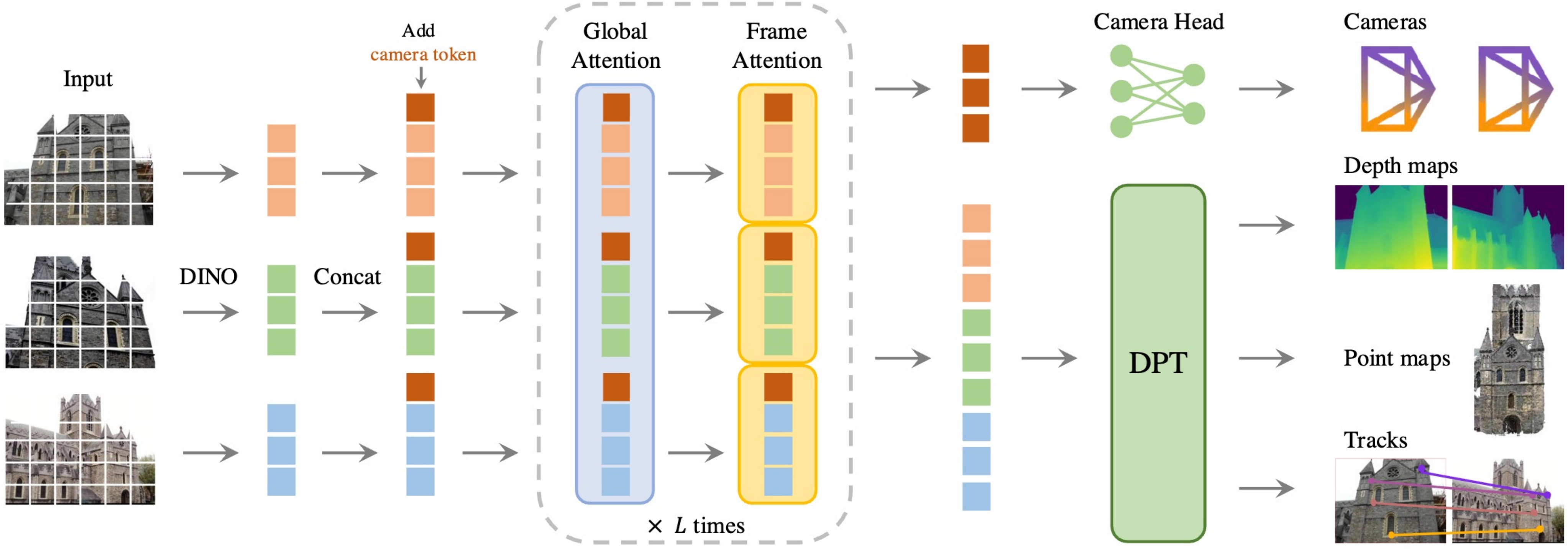
Challenges:

1. How to align point maps of each image?
2. How to deal with various number of input images?



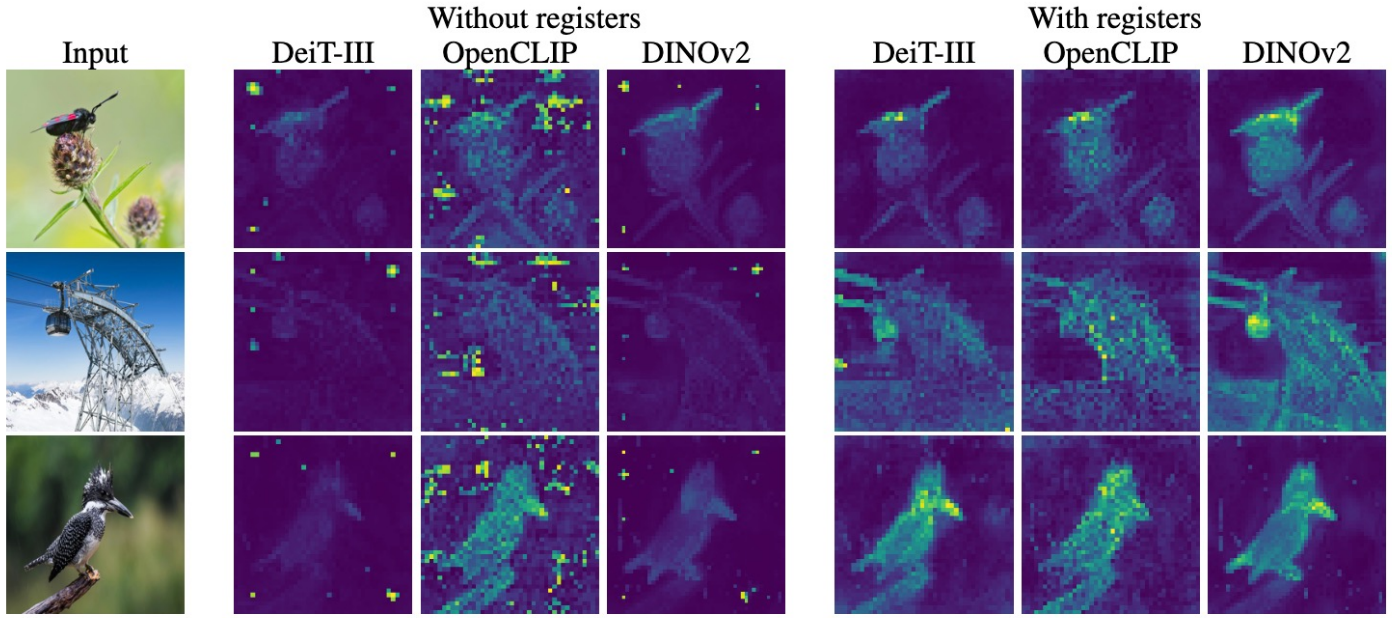
Principle design

1. Set the first input image as the reference image.
2. Introduce camera token to estimate camera parameters.
3. Introduce register token to reduce influence of global tokens.
4. Introduce two set of camera and register tokens to distinguish the first image as reference frame.

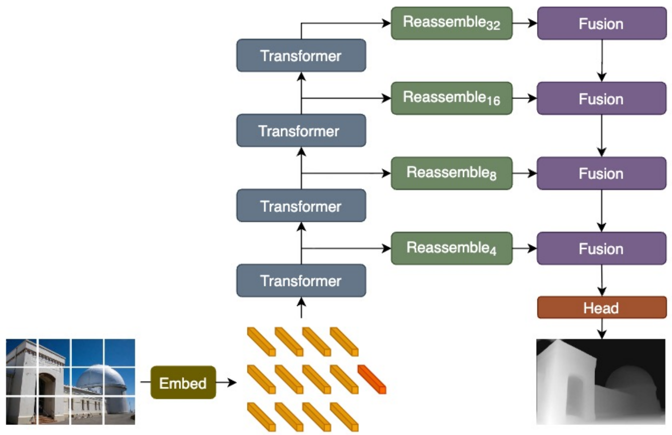


Reference

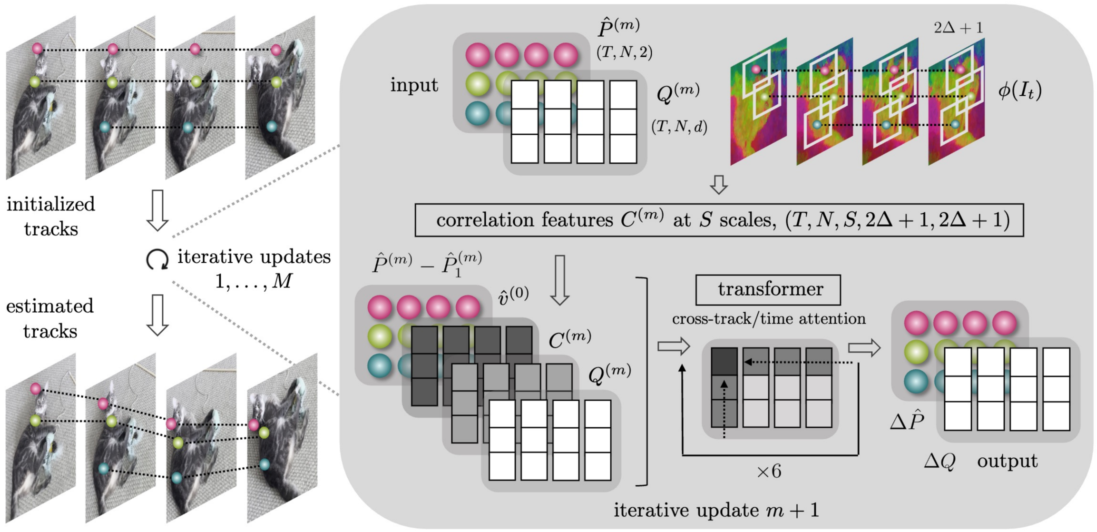
1. Vit Needs Registers



2. Vit for dense prediction



3. Cotracker



Experiments

1. Camera Pose Estimation

Methods	Re10K *(unseen)* AUC@30 ↑	CO3Dv2 AUC@30 ↑	Time
Colmap+SPSG [92]	45.2	25.3	∼ 15s
PixSfM [66]	49.4	30.1	> 20s
PoseDiff [124]	48.0	66.5	∼ 7s
DUSt3R [129]	67.7	76.7	∼ 7s
MASt3R [62]	76.4	81.8	∼ 9s
VGGSfM v2 [125]	78.9	83.4	∼ 10s
MV-DUSt3R [111] ‡	71.3	69.5	∼ 0.6s
CUT3R [127] ‡	75.3	82.8	∼ 0.6s
FLARE [156] ‡	78.8	83.3	∼ 0.5s
Fast3R [141] ‡	72.7	82.5	∼ **0.2s**
Ours (Feed-Forward)	85.3	88.2	∼ **0.2s**
Ours (with BA)	**93.5**	**91.8**	∼ 1.8s

Table 1. **Camera Pose Estimation on RealEstate10K [161] and CO3Dv2 [88]** with 10 random frames. All metrics the higher the better. None of the methods were trained on the Re10K dataset. Runtime were measured using one H100 GPU. Methods marked with ‡ represent concurrent work.

2. Multi-view depth prediction

Known GT camera	Method	Acc.↓	Comp.↓	Overall↓
✓	Gipuma [40]	**0.283**	0.873	0.578
✓	MVSNet [144]	0.396	0.527	0.462
✓	CIDER [139]	0.417	0.437	0.427
✓	PatchmatchNet [121]	0.427	0.377	0.417
✓	MASt3R [62]	0.403	0.344	0.374
✓	GeoMVSNet [157]	0.331	**0.259**	**0.295**
✗	DUSt3R [129]	2.677	0.805	1.741
✗	Ours	**0.389**	**0.374**	**0.382**

Table 2. **Dense MVS Estimation on the DTU [51] Dataset.** Methods operating with known ground-truth camera are in the top part of the table, while the bottom part contains the methods that do not know the ground-truth camera.

Experiments

3. Point map estimation

Methods	Acc.↓	Comp.↓	Overall↓	Time
DUSt3R	1.167	0.842	1.005	$\sim$ 7s
MASt3R	0.968	0.684	0.826	$\sim$ 9s
Ours (Point)	0.901	0.518	0.709	$\sim$ 0.2s
Ours (Depth + Cam)	**0.873**	**0.482**	**0.677**	$\sim$ 0.2s

Table 3. **Point Map Estimation on ETH3D [97].** DUSt3R and MASt3R use global alignment while ours is feed-forward and, hence, much faster. The row *Ours (Point)* indicates the results using the point map head directly, while *Ours (Depth + Cam)* denotes constructing point clouds from the depth map head combined with the camera head.

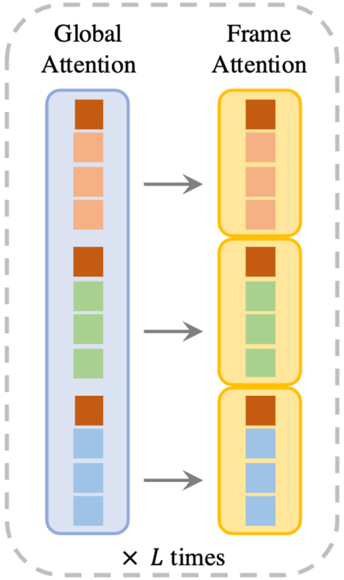
4. Dynamic point tracking

Method	Kinetics			RGB-S			DAVIS		
	AJ	δ^{vis}_{avg}	OA	AJ	δ^{vis}_{avg}	OA	AJ	δ^{vis}_{avg}	OA
TAPTR [63]	49.0	64.4	85.2	60.8	76.2	87.0	63.0	76.1	91.1
LocoTrack [13]	52.9	66.8	85.3	69.7	83.2	89.5	62.9	75.3	87.2
BootsTAPIR [26]	54.6	68.4	86.5	70.8	83.0	89.9	61.4	73.6	88.7
CoTracker [56]	49.6	64.3	83.3	67.4	78.9	85.2	61.8	76.1	88.3
CoTracker + Ours	**57.2**	**69.0**	**88.9**	**72.1**	**84.0**	**91.6**	**64.7**	**77.5**	**91.4**

Table 8. **Dynamic Point Tracking Results on the TAP-Vid benchmarks.** Although our model was not designed for dynamic scenes, simply fine-tuning CoTracker with our pretrained weights significantly enhances performance, demonstrating the robustness and effectiveness of our learned features.

Ablations

1. **Attention structure**



ETH3D Dataset	Acc.↓	Comp.↓	Overall↓
Cross-Attention	1.287	0.835	1.061
Global Self-Attention Only	1.032	0.621	0.827
Alternating-Attention	**0.901**	**0.518**	**0.709**

2. **Multi-task training**

w. $\mathcal{L}_{camera}$	w. $\mathcal{L}_{depth}$	w. $\mathcal{L}_{track}$	Acc.↓	Comp.↓	Overall↓
✗	✓	✓	1.042	0.627	0.834
✓	✗	✓	0.920	0.534	0.727
✓	✓	✗	0.976	0.603	0.790
✓	✓	✓	**0.901**	**0.518**	**0.709**