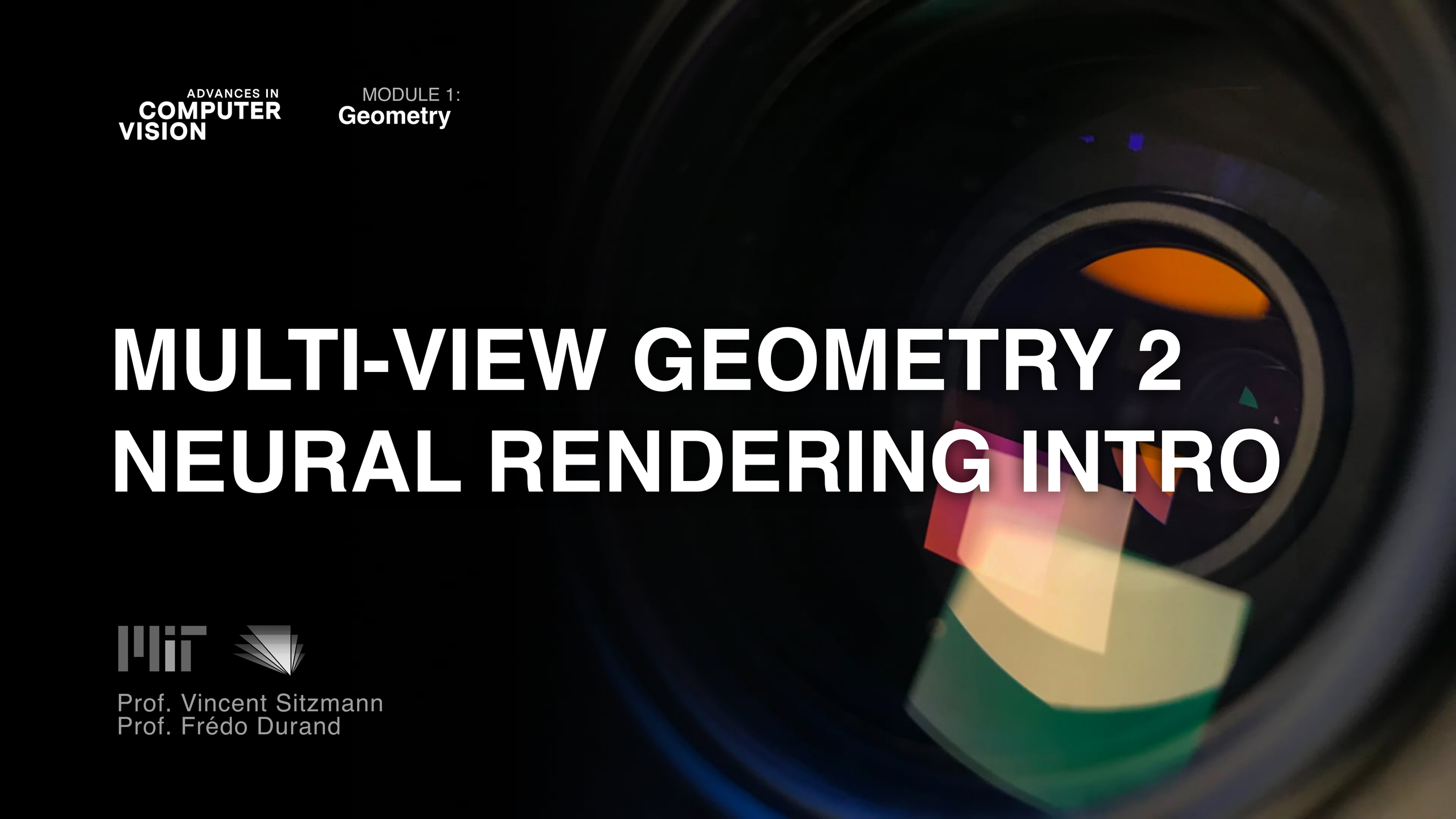
ADVANCES IN COMPUTER VISION

MODULE 1:
Geometry

MULTI-VIEW GEOMETRY 2
NEURAL RENDERING INTRO



Prof. Vincent Sitzmann
Prof. Frédo Durand

Mid Term

- You are expected to be there
  in person on April 9 at 1pm

- If that’s a problem, get S^3 or Grad Support to let
  us know before March 31



Final projects

- Culmination, highlight (only point?) of the class
  - Have fun! Explore! Impress us!
    Maybe get a paper out of it.
- Double dipping is fine
  (connection to your PhD/urop research)
- Groups of 2-3 encouraged.
  - Let us/piaza know if you're looking for teammates.
- Guidelines should be released on Tuesday March 17



Talk by Irfan Essa

- Tuesday March 17 at 12pm (just before class) location TBD
- https://www.csail.mit.edu/event/visual-computing-seminar-wizard-oz-experiential-time-machine-powered-google-ai-now-playing

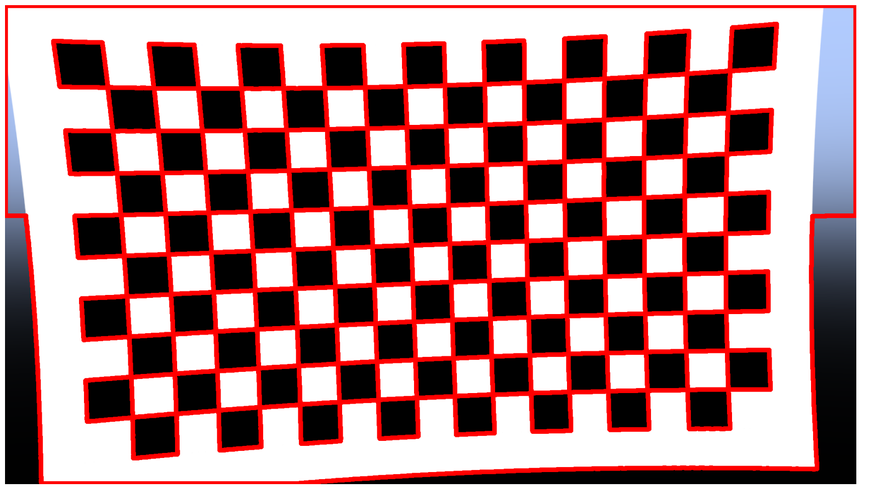


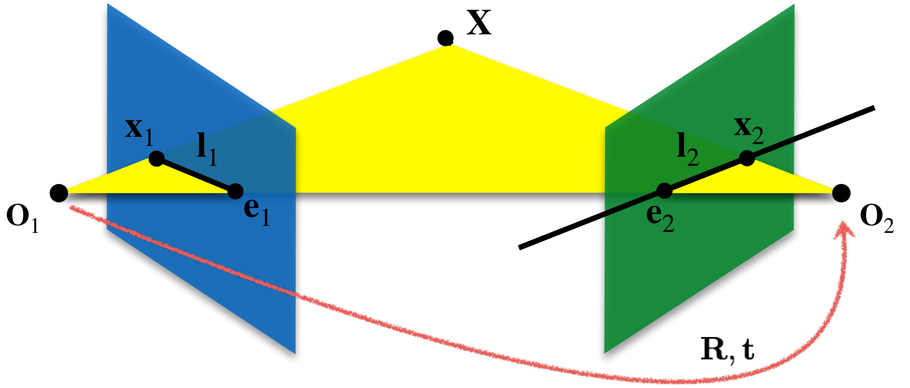
Photo by Paul Debevec



Recap camera calibration

- Sparse features + correspondences
- Intrinsic + radial calibration
  - Known target (checkerboard)
- Epipolar Geometry, fundamental matrix
- Extrinsic camera calibration with linear regression + RANSAC
- Refined with non-linear bundle adjustment (including intrinsics, extrinsics, radial)
- We also get sparse 3D points

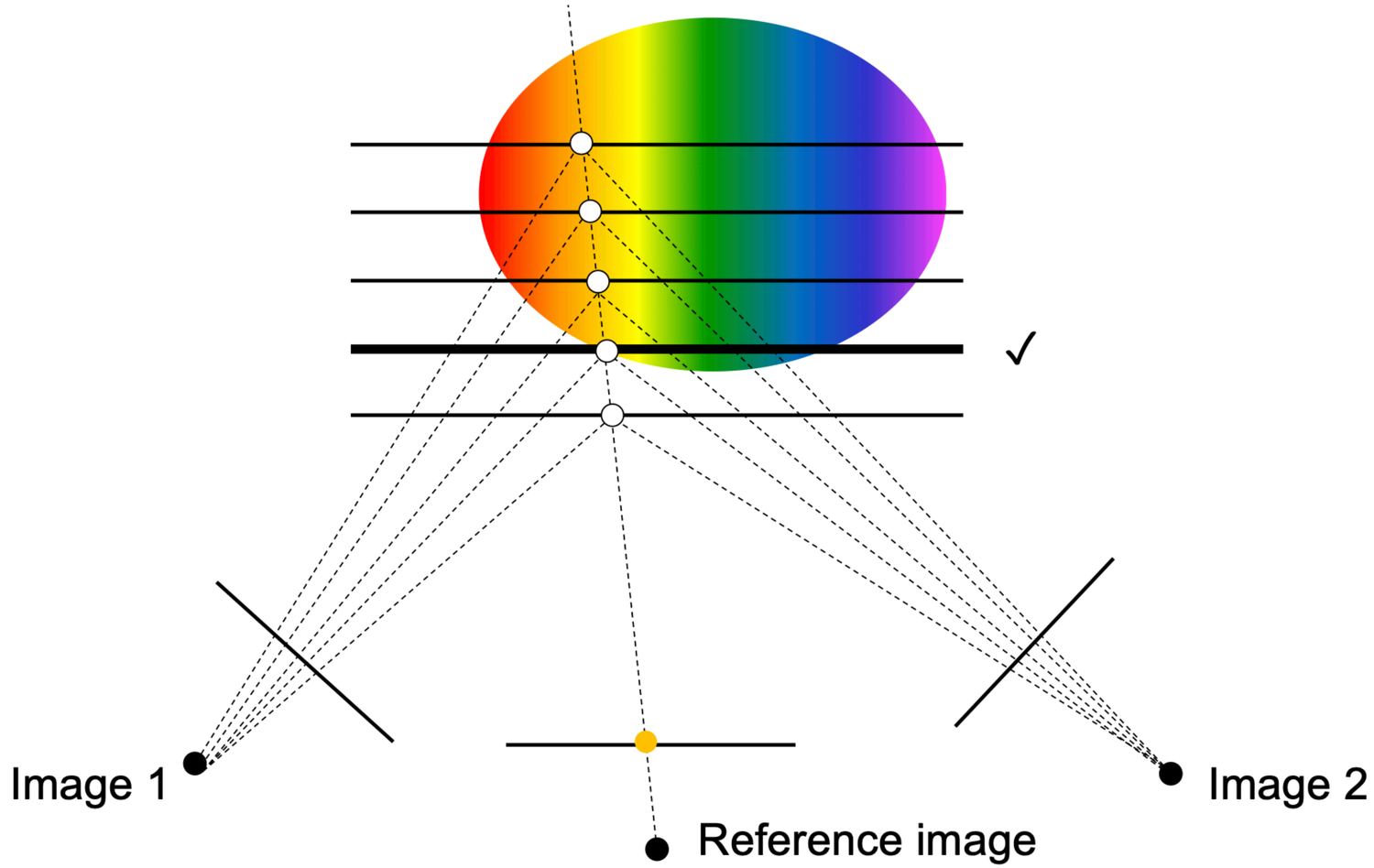






- Excellent open-source software exists

Dense multiview stereo : plane sweep



https://slazebni.cs.illinois.edu/fall24/lec19_multiview_stereo.pdf

Dense multiview stereo : Patch-based

- The State of the Art before the deep learning era
- PMVS Patch-based Multi-View Stereo, Furukawa & Ponce, 2007-2010
- Improved by PMVS 2 & Clustering Views for Multi-view Stereo (CMVS)



Fig. 1. Overall approach. From left to right: A sample input image, detected features, reconstructed patches after the initial matching, final patches after expansion and filtering, and the mesh model.

https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5226635

https://www.di.ens.fr/pmvs/

https://www.di.ens.fr/cmvs/

PMVS

- 3D is a collection of square patches: local surface approximation
  - From 100k to 10M patches
- Ensure photometric consistency across images where patch is visible

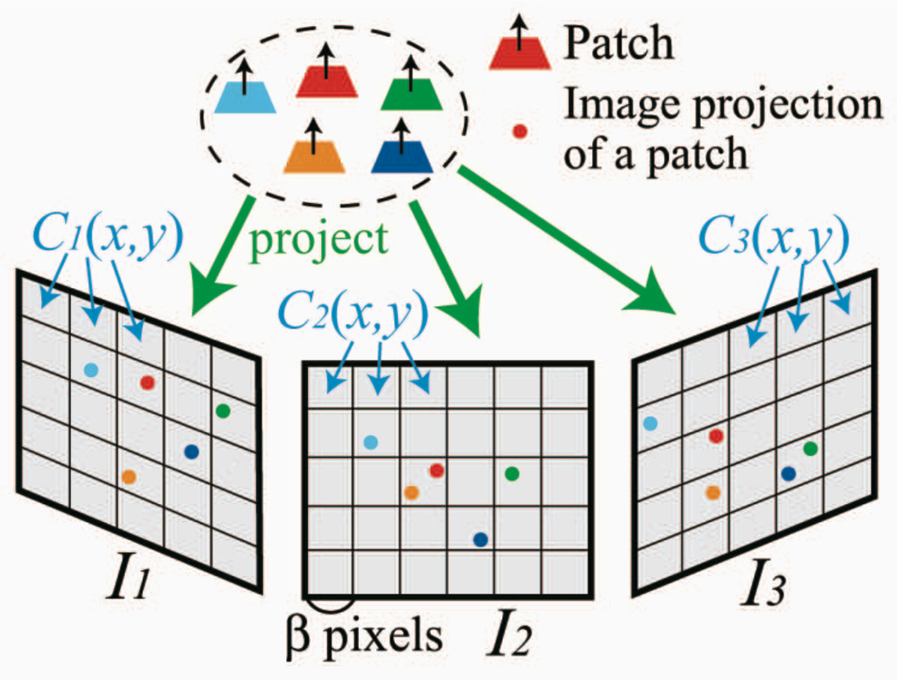


Fig. 3. We keep track of image projections of reconstructed patches in their visible images to perform fundamental tasks such as accessing neighboring patches, enforcing regularization, etc. See text for more details.

Patch can be confusing because in other parts of vision it means a kxk window.

PMVS

- Match step: Initialize patches around triangulated SfM/Harris points
- Then iterate: expand (new patches around existing patches, and filter (visibility + regularization)
- Can be turned into meshes in a. Final step

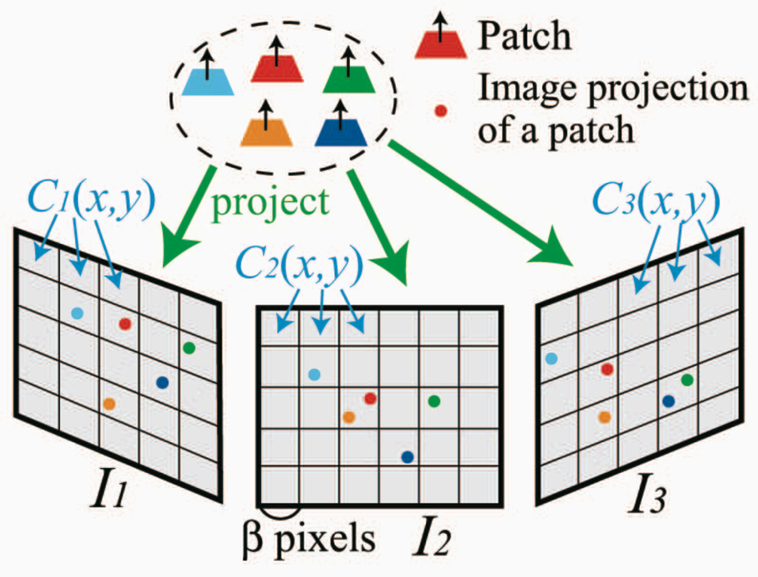


Fig. 3. We keep track of image projections of reconstructed patches in their visible images to perform fundamental tasks such as accessing neighboring patches, enforcing regularization, etc. See text for more details.

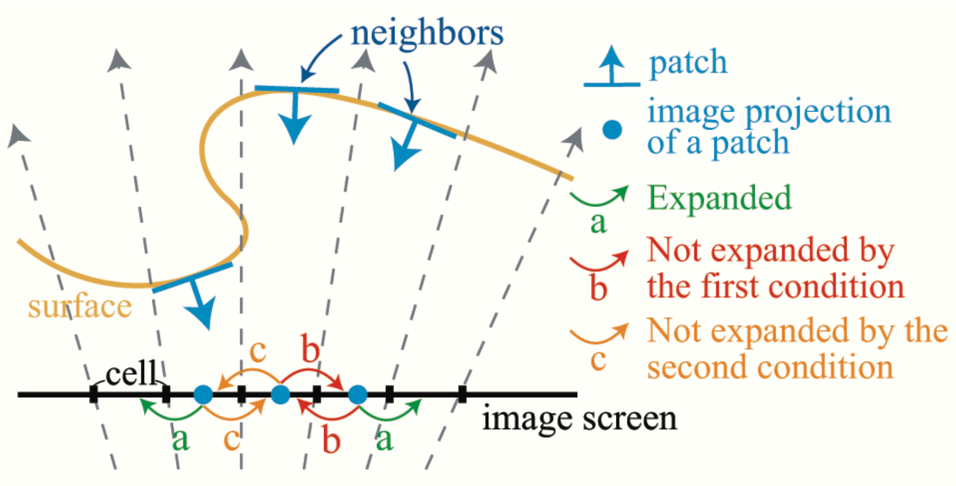
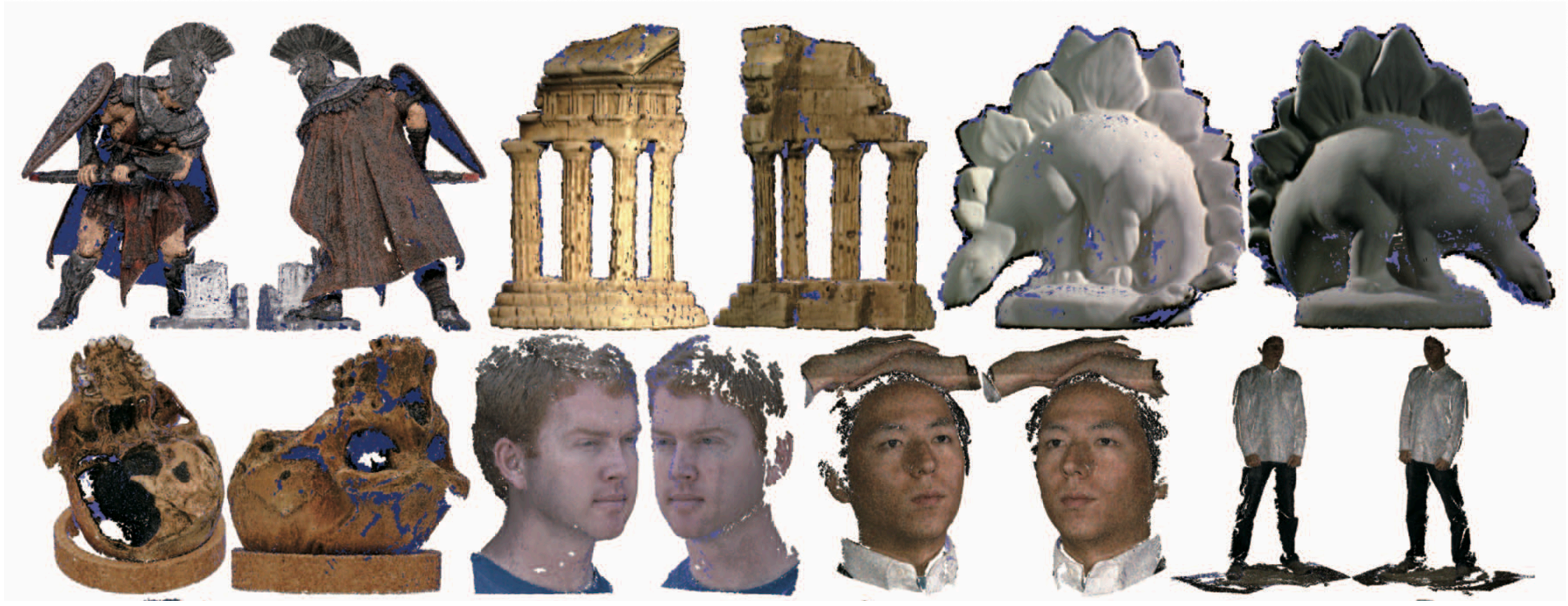


Fig. 5. (a) Given an existing patch, an expansion procedure is performed to generate new ones for the neighboring empty image cells in its visible images. The expansion procedure is not performed for an image cell (b) if there already exists a neighboring patch reconstructed there or (c) if there is a depth discontinuity when viewed from the camera. See text for more details.



Limitations

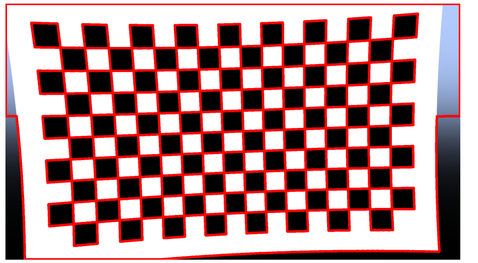
- Slow: tens of minutes to days
- Assumes well-behaved objects : clean surface, Lambertian/diffuse
- Still holes

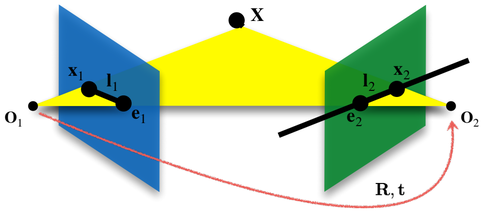




Discussion of traditional 3D reconstruction

- Intrinsic calibration, key points, correspondences, RANSAC + 8 points, bundle adjustment, dense reconstruction with patches, mesh reconstruction + texture
- Long pipeline, each step implemented manually and imperfect, errors/robustness issues accumulate
- Slow
- Final results have holes, robustness is limited







- But again, SfM (but not dense reconstruction) is still the first step in many modern methods to get camera calibration



Deep Learning Era

Not covered, will do next time.

Skipping intermediate steps/papers

- All modules of SfM (Structure from Motion) have undergone neural improvements
  - advanced feature description,
  - more accurate image matching,
  - feature metric refinement,
  - and neural bundle adjustment+.

E.g.

https://arxiv.org/pdf/1712.07629

https://arxiv.org/pdf/1905.03561

https://arxiv.org/pdf/2404.14351

https://arxiv.org/pdf/2404.15259

https://arxiv.org/pdf/2312.04563

https://openaccess.thecvf.com/content_CVPR_2020/papers/Gu_Cas
View_Stereo_and_Stereo_Matching_CVPR_2020_paper.pdf

SuperPoint: Self-Supervised Interest Point Detection and Description

Daniel DeTone
Magic Leap
Sunnyvale, CA
ddetone@magicleap.com

Tomasz Malisiewicz
Magic Leap
Sunnyvale, CA
tmalisiewicz@magicleap.com

Andrew Rabinovich
Magic Leap
Sunnyvale, CA
arabinovich@magicleap.com

D2-Net: A Trainable CNN for *Joint Description and Detection* of Local Features

Mihai Dusmanu[1,2,3] Ignacio Rocco[1,2] Tomas Pajdla[4] Marc Pollefeys[3,5]

Scene Coordinate Reconstruction: Posing of Image Collections via Incremental Learning of a Relocalizer

Eric Brachmann[1], Jamie Wynn[1], Shuai Chen[2], Tommaso Cavallari[1],
Áron Monszpart[1], Daniyar Turmukhambetov[1], and Victor Adrian Prisacariu[1,2]

FlowMap: High-Quality Camera Poses, Intrinsics, and Depth via Gradient Descent

Cameron Smith*, David Charatan*,
Ayush Tewari, and Vincent Sitzmann

Visual Geometry Grounded Deep Structure From Motion

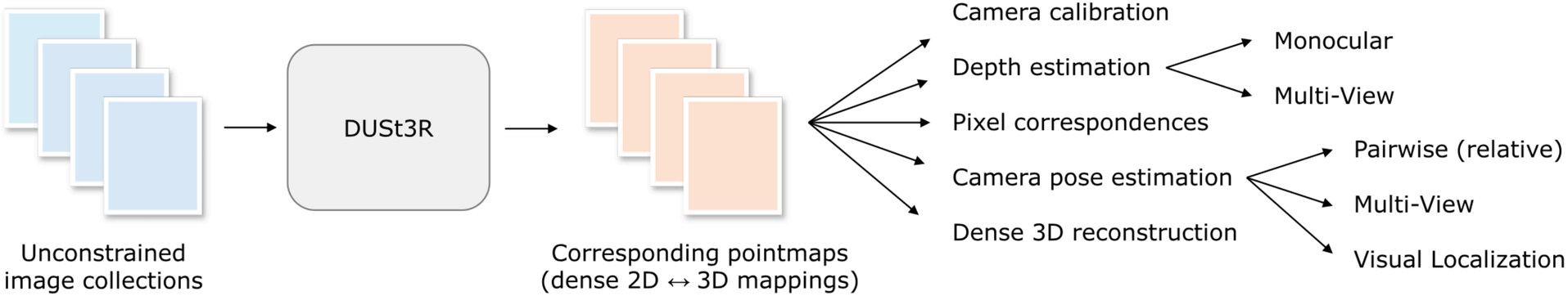
Jianyuan Wang[1,2] Nikita Karaev [1,2] Christian Rupprecht[1] David Novotny[2]

Cascade Cost Volume for High-Resolution Multi-View Stereo and Stereo Matching

Xiaodong Gu[1*] Zhiwen Fan[1*] Siyu Zhu[1] Zuozhuo Dai[1] Feitong Tan[12†] Ping Tan[1]
[1]Alibaba A.I. Labs [2]Simon Fraser University

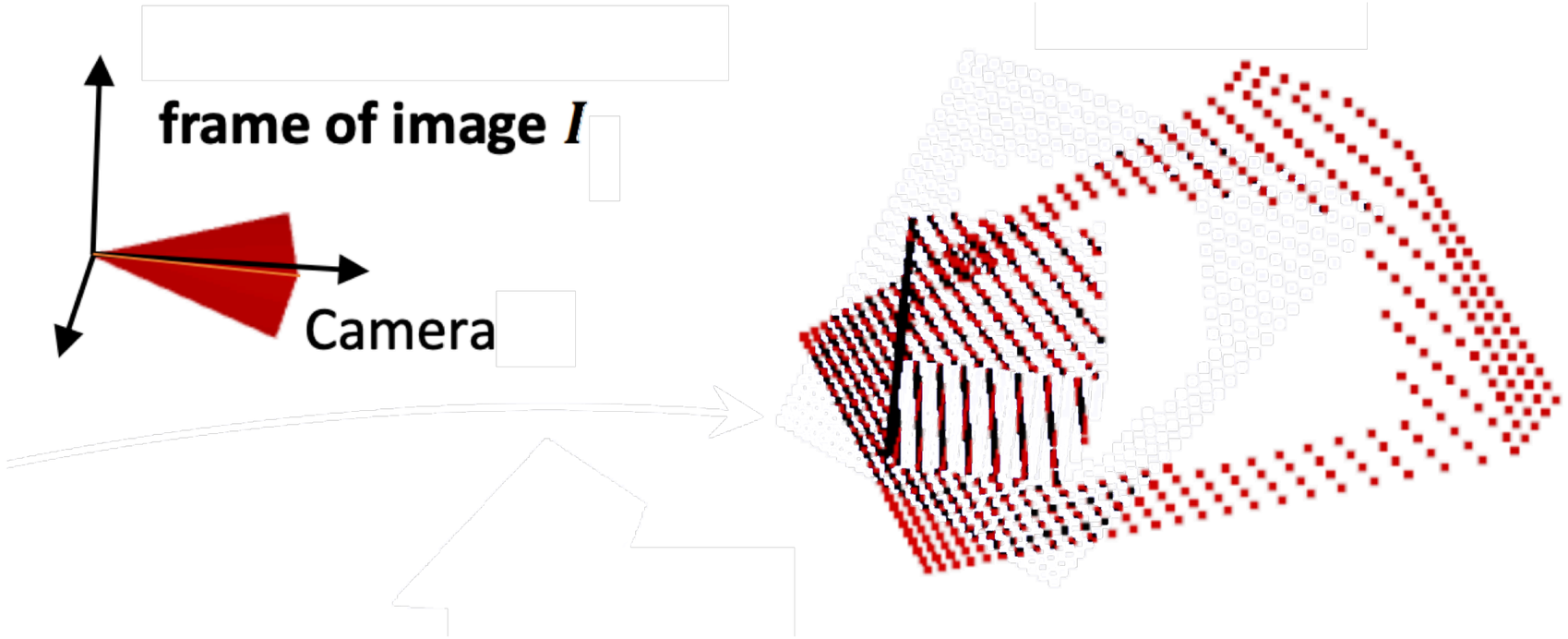
DUST3R: Dense Unconstrained Stereo 3D Reconstruction

- Supervised (synthetically generated, reconstructed from SfM softwares, or captured using dedicated sensors)
- Monocular + stereo (+ multiple views as post-process)
- Only 2 images at a time
- Directly on 3D point maps
- Loss : mostly 3D distance on ground truth point maps (+confidence)
- Camera pose inferred as post-process (regression on point maps)

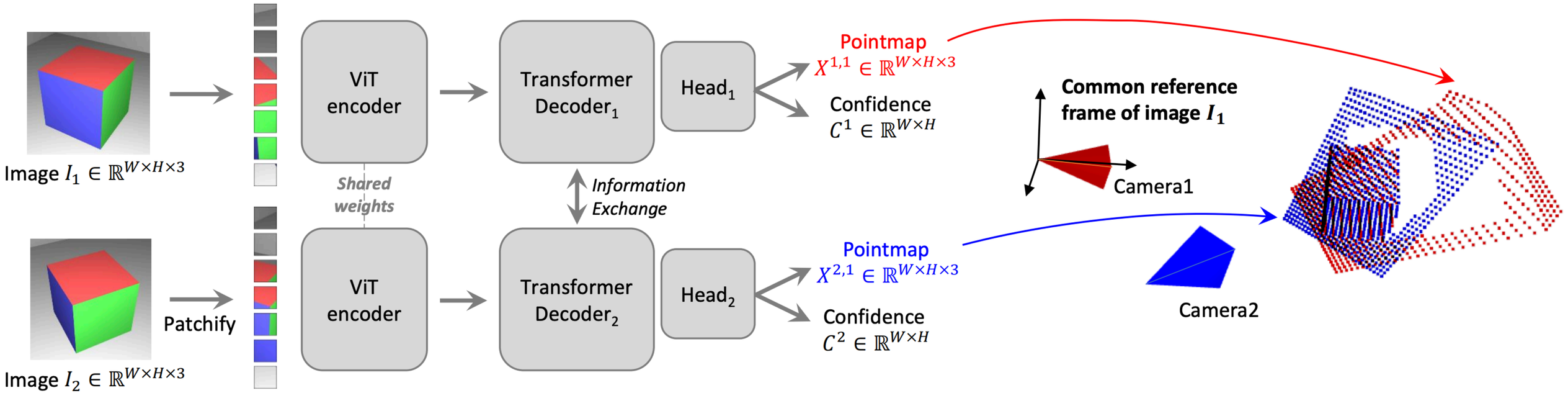


Point map

- Dense 2D field of 3D points
  (un-projection of an image according to per-pixel depth)



DUST3R: Pose SfM as supervised learning problem



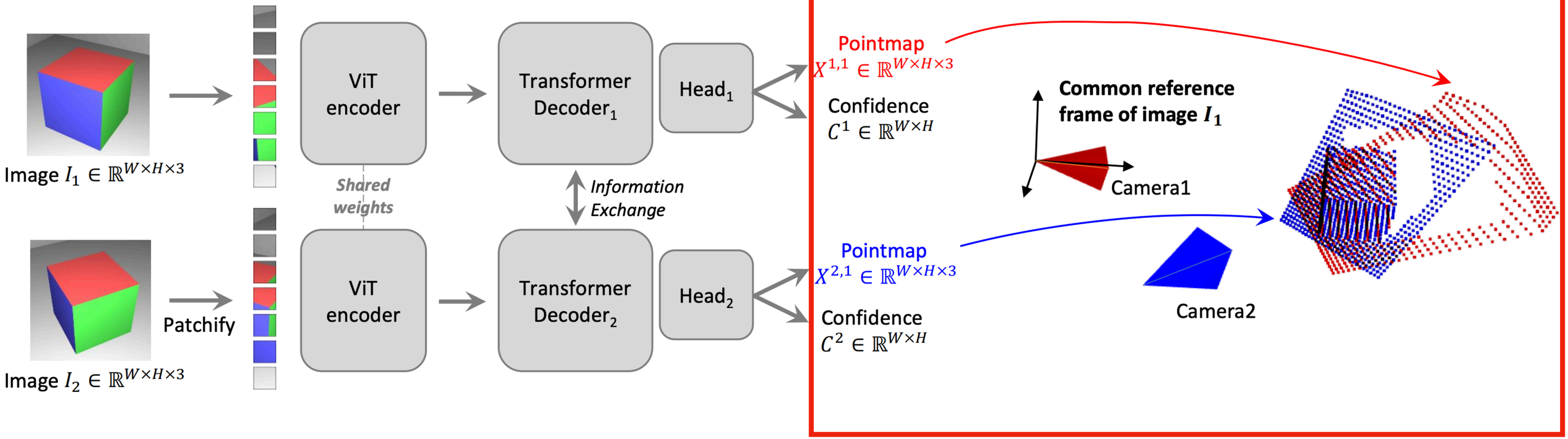
Both predict 3D coordinates in frame of camera 1

No explicit matching, no camera pose calibration: directly predict 3D coordinates

Camera pose can be obtained as a regression post-processbetween two point maps.

Wang et al., DUSt3R: Geometric 3D Vision Made Easy, CVPR 2024

DUST3R: Pose SfM as supervised learning problem



What made it work:

- Curating large supervised dataset from SfM reconstructions
- PointMap parameterization

Plus I suspect monocular cues do a lot of lifting

Wang et al., DUSt3R: Geometric 3D Vision Made Easy, CVPR 2024

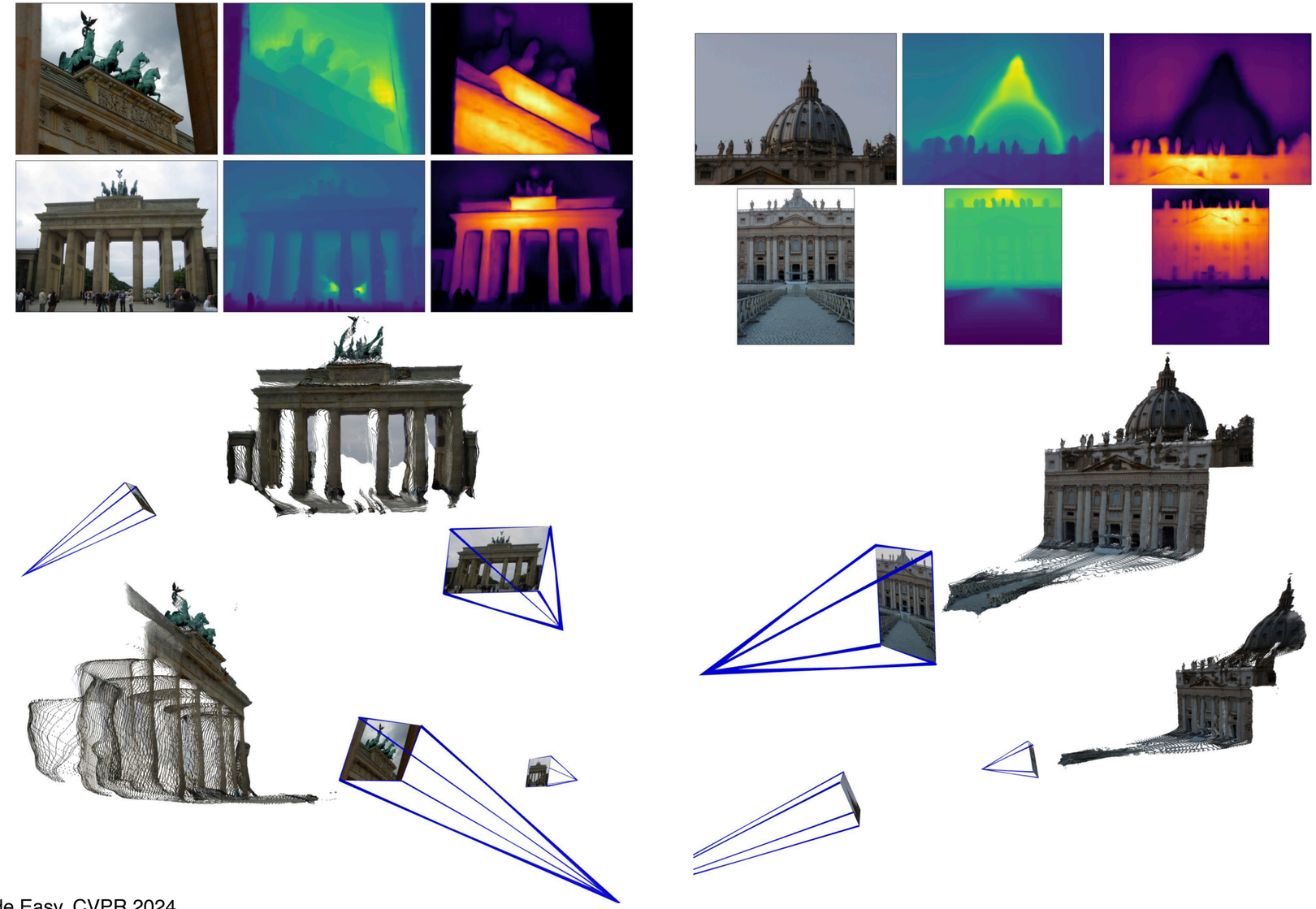
DUST3R: Pose SfM as supervised learning problem

Input

RGB images

No calibration, No poses

DUST3R: Pose SfM as supervised learning problem



Wang et al., DUSt3R: Geometric 3D Vision Made Easy, CVPR 2024

Limitations of DUST3R

Only pairs of images at a time

Limited resolution

  All limited by memory

Not very precise (largely because no correspondences)

Follow ups

MAST3R

FAST3R

Instasplat

Grounding Image Matching in 3D with MASt3R

Vincent Leroy Yohann Cabon Jerome Revaud
NAVER LABS Europe
https://github.com/naver/mast3r

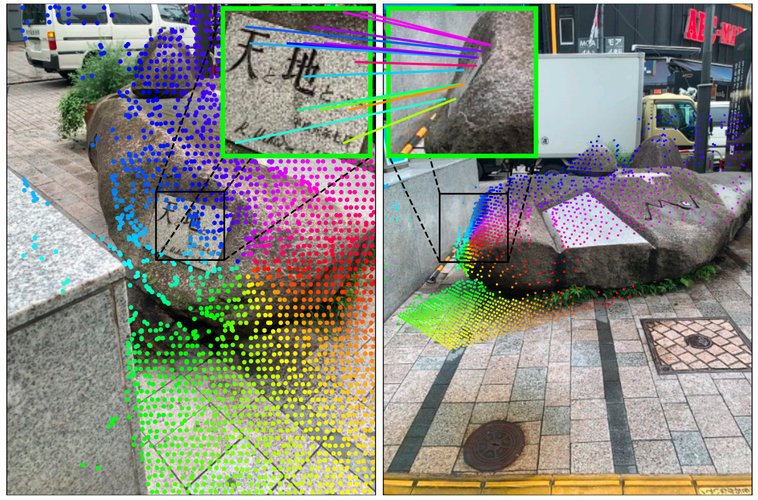


Figure 1: **Dense Correspondences.** MASt3R extends DUSt3R as it predicts dense correspondences, even in regions where camera motion significantly degrades the visual similarity. Focal length can be derived from the predicted 3D geometry, making our approach a standalone method for camera calibration, camera pose estimation and 3D scene reconstruction, attaining and improving the performance of the state of the art on several extremely challenging benchmarks.

MASt3R-SLAM: Real-Time Dense SLAM with 3D Reconstruction Priors

Riku Murai* Eric Dexheimer* Andrew J. Davison
Imperial College London
{riku.murai15, e.dexheimer21, a.davison}@imperial.ac.uk

Abstract

We present a real-time monocular dense SLAM system designed bottom-up from MASt3R, a two-view 3D reconstruction and matching prior. Equipped with this strong prior, our system is robust on in-the-wild video sequences despite making no assumption on a fixed or parametric camera model beyond a unique camera centre. We introduce efficient methods for pointmap matching, camera tracking and local fusion, graph construction and loop closure, and second-order global optimisation. With known calibration, a simple modification to the system achieves state-of-the-art performance across various benchmarks. Altogether, we propose a plug-and-play monocular SLAM system capable of producing globally consistent poses and dense geometry while operating at 15 FPS.

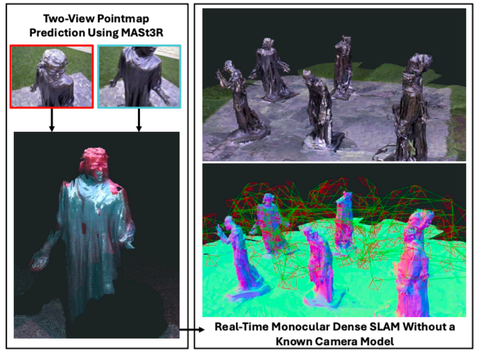


Figure 1. Reconstruction from our dense monocular SLAM system on the Burghers sequence [56]. Using two-view predictions from MASt3R shown on the left, our system achieves globally consistent

InstantSplat: Sparse-view SfM-free Gaussian Splatting in Seconds

Zhiwen Fan[1,2†*], Wenyan Cong[1*], Kairun Wen[3*], Kevin Wang[1], Jian Zhang[3], Xinghao Ding[3], Danfei Xu[2,4], Boris Ivanovic[2], Marco Pavone[2,5], Georgios Pavlakos[1], Zhangyang Wang[1], Yue Wang[2,6]
*Authors contributed equally; † Z. Fan is the Project Lead.
[1]University of Texas at Austin [2]Nvidia Research [3]Xiamen University
[4]Georgia Institute of Technology [5]Stanford University [6]University of Southern California
Project Website: https://instantsplat.github.io

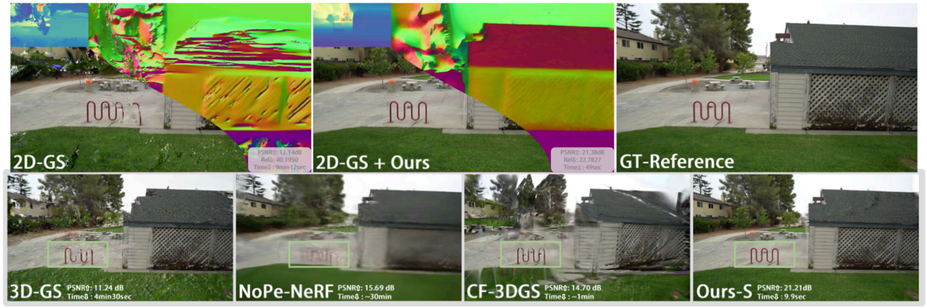
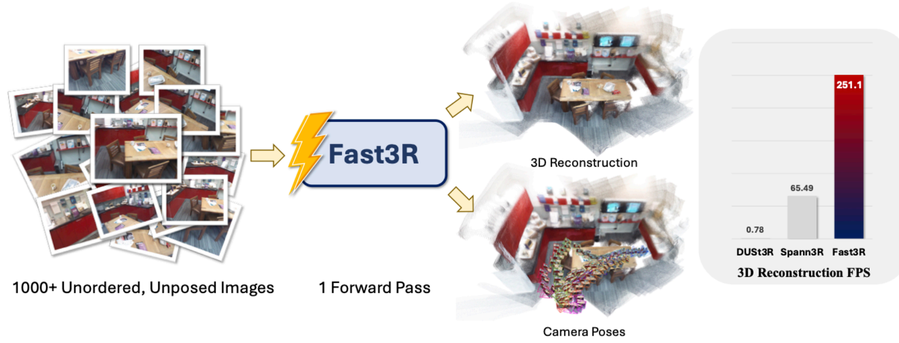


Figure 1. **InstantSplat** processes sparse-view, unposed images to reconstruct a radiance field, capturing detailed scenes rapidly without relying on Structure-from-Motion. Optimization occurs under self-supervision with the support of the large pretrained model, MASt3R [23].

Fast3R: Towards 3D Reconstruction of 1000+ Images in One Forward Pass

Jianing Yang♣♦ Alexander Sax♣ Kevin J. Liang♣ Mikael Henaff♣ Hao Tang♣
Ang Cao♣♦ Joyce Chai♦ Franziska Meier♣ Matt Feiszli♣

♣ Meta FAIR ♦ University of Michigan
https://fast3r-3d.github.io



MAST3R: Matching And Stereo 3D Reconstruction

Adds correspondences to DUST3R -> more precision

Correspondences directly in 3D, not 2D, and with full context, not just patch by patch

Plus sub-quadratic matching

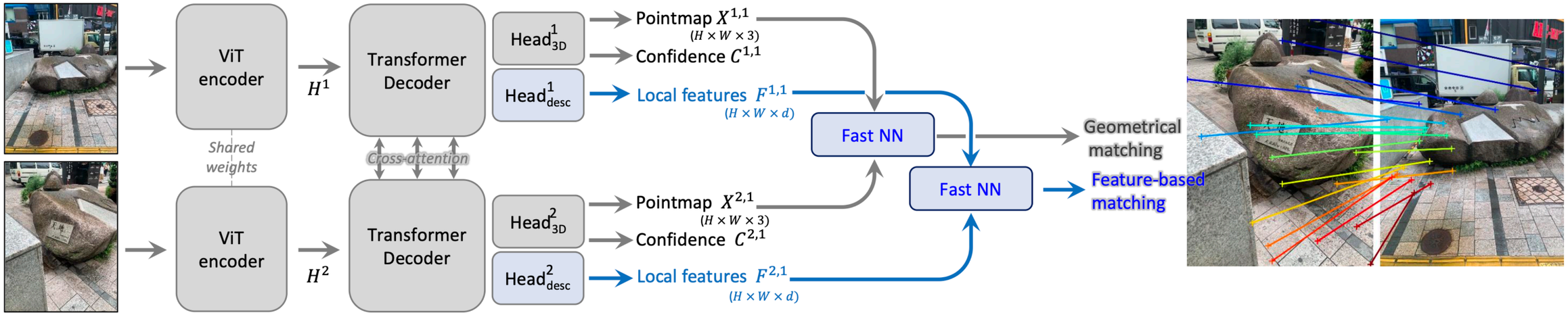


Figure 2: Overview of the proposed approach. Given two input images to match, our network regresses for each image and each input pixel a 3D point, a confidence value and a local feature. Plugging either 3D points or local features into our fast reciprocal NN matcher (3.3) yields robust correspondences. Compared to the DUSt3R framework which we build upon, our contributions are highlighted in blue.

MAST3R: Matching And Stereo 3D Reconstruction

New head that outputs dense features maps for matching

More or less cross-entropy classification loss to encourage single pixel-accurate match

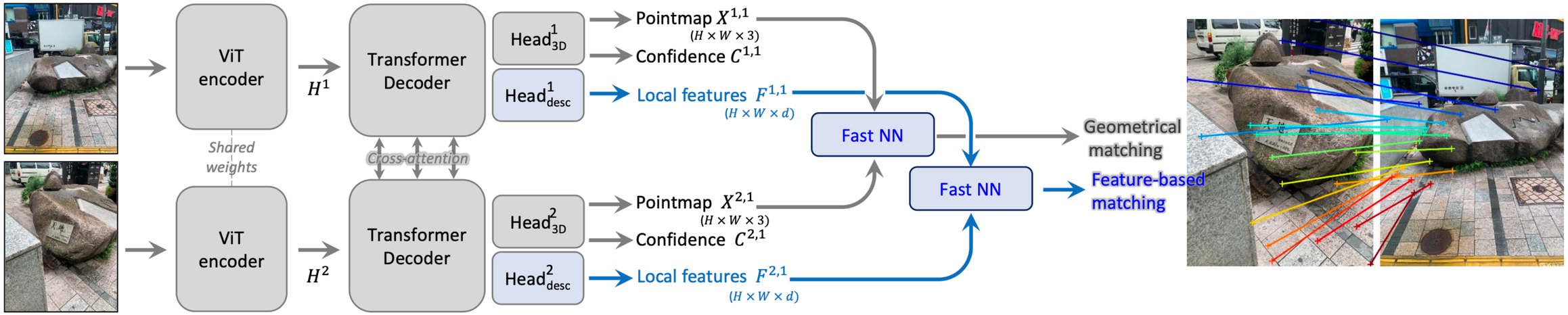


Figure 2: Overview of the proposed approach. Given two input images to match, our network regresses for each image and each input pixel a 3D point, a confidence value and a local feature. Plugging either 3D points or local features into our fast reciprocal NN matcher (3.3) yields robust correspondences. Compared to the DUSt3R framework which we build upon, our contributions are highlighted in blue.

Fast reciprocal matching

New but non-neural fast dense matching method

Subsample + remove pixel for which a good match (cycle) has already been found

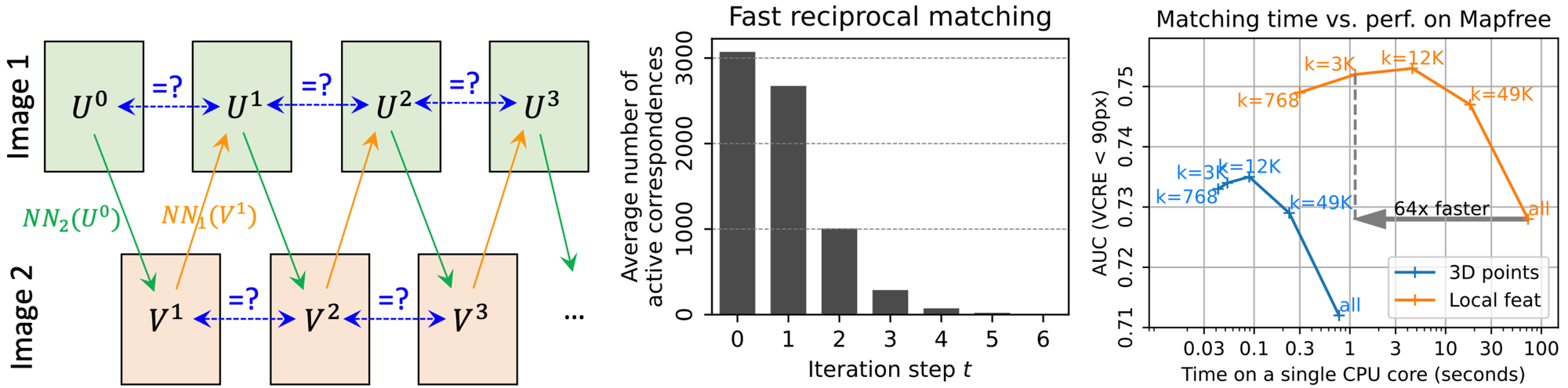
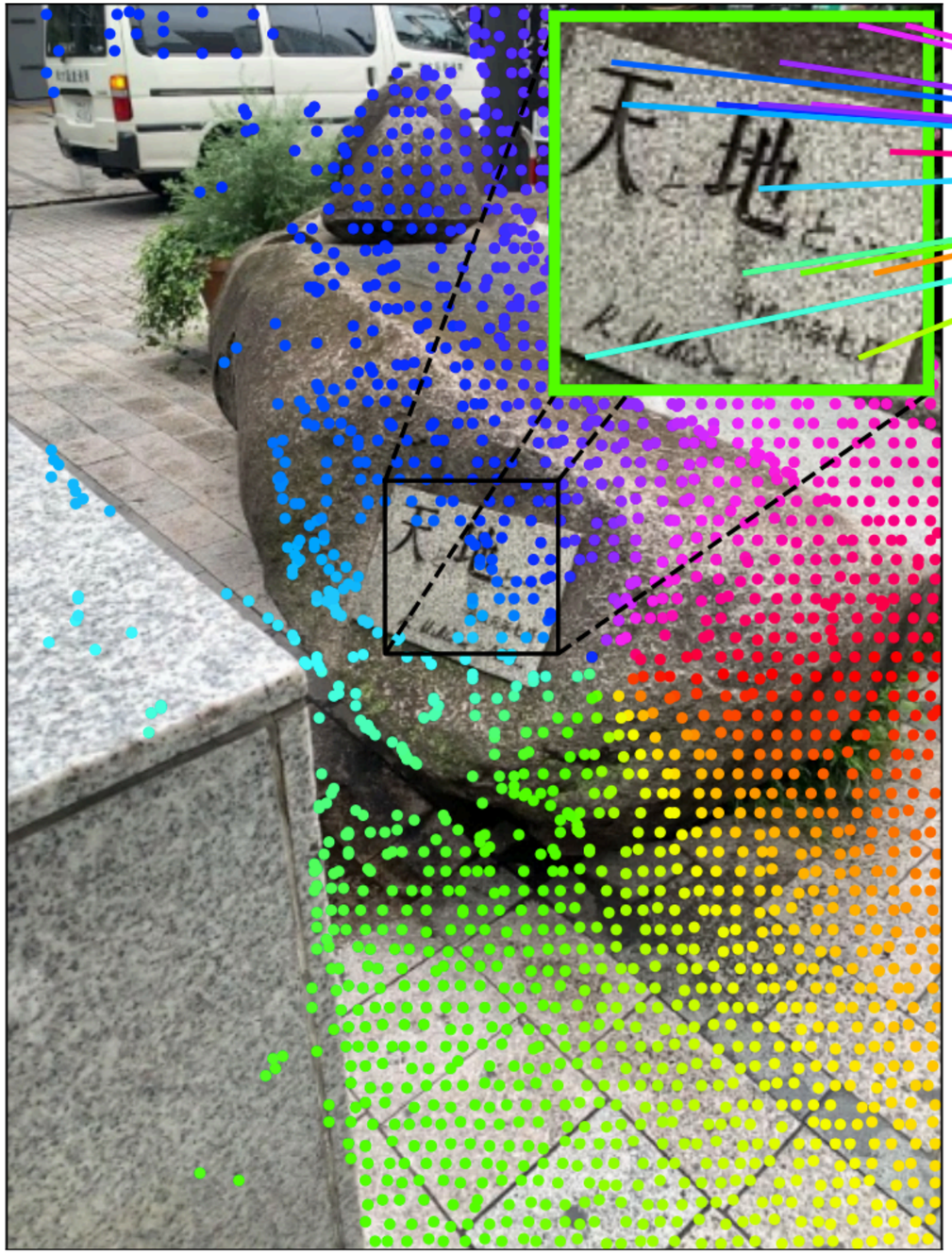


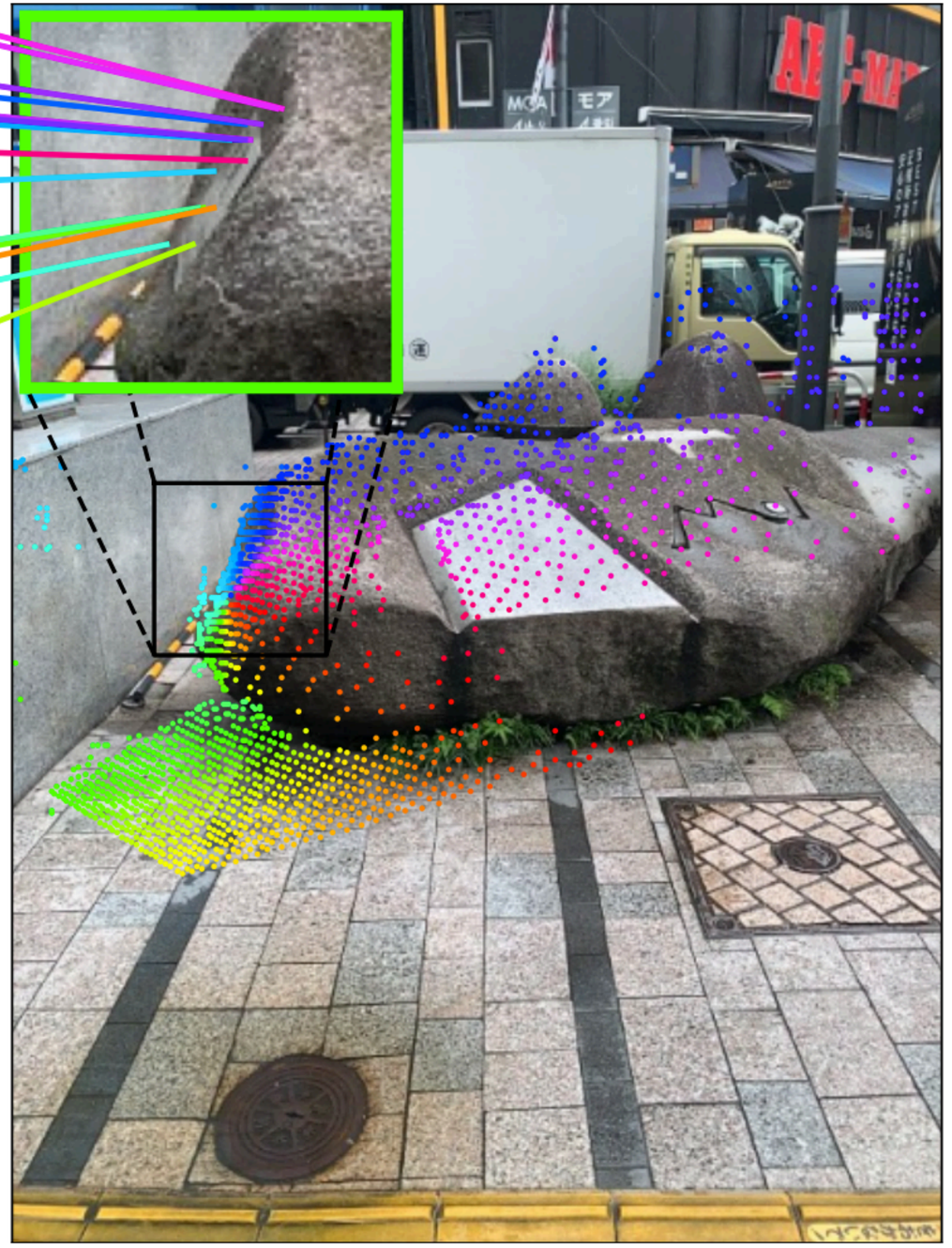
Figure 3: Fast reciprocal matching. **Left**: Illustration of the fast matching process, starting from an initial subset of pixels U^0 and propagating it iteratively using NN search. Searching for cycles (blue arrows) detect reciprocal correspondences and allows to accelerate the subsequent steps, by removing points that converged. **Center**: Average number of remaining points in U^t at iteration $t = 1 \ldots 6$. After only 5 iterations, nearly all points have already converged to a reciprocal match. **Right**: Performance-versus-time trade-off on the Map-free dataset.

G

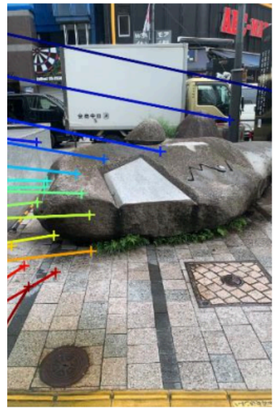




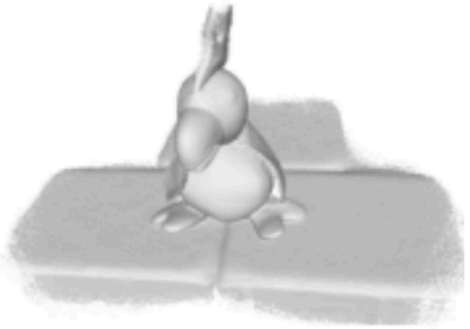


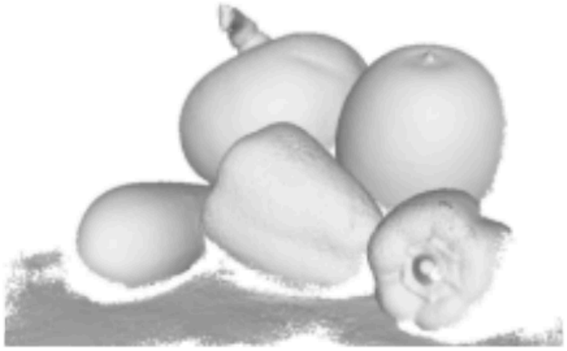


St3R

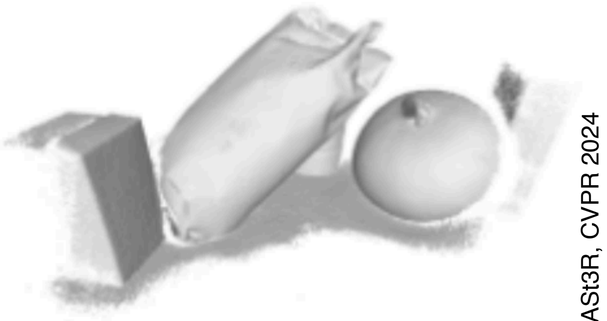


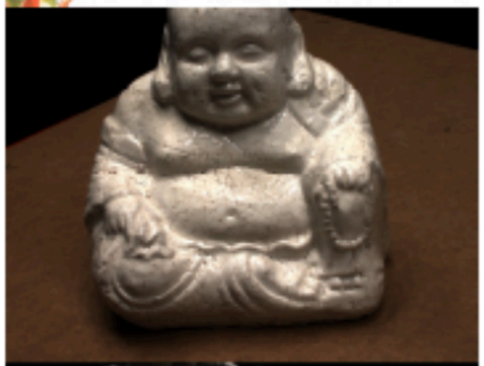
Leroy et al., Grounding Image Matching in 3D with MASt3R, CVPR 2024












MAST3R

Leroy et al., Grounding Image Matching in 3D with MASt3R, CVPR 2024

MASt3R-SLAM: Real-Time Dense SLAM with 3D Reconstruction Priors

Riku Murai* Eric Dexheimer* Andrew J. Davison
Imperial College London
{riku.murai15, e.dexheimer21, a.davison}@imperial.ac.uk

Abstract

We present a real-time monocular dense SLAM system designed bottom-up from MASt3R, a two-view 3D reconstruction and matching prior. Equipped with this strong prior, our system is robust on in-the-wild video sequences despite making no assumption on a fixed or parametric camera model beyond a unique camera centre. We introduce efficient methods for pointmap matching, camera tracking and local fusion, graph construction and loop closure, and second-order global optimisation. With known calibration, a simple modification to the system achieves state-of-the-art performance across various benchmarks. Altogether, we propose a plug-and-play monocular SLAM system capable of producing globally consistent poses and dense geometry while operating at 15 FPS.

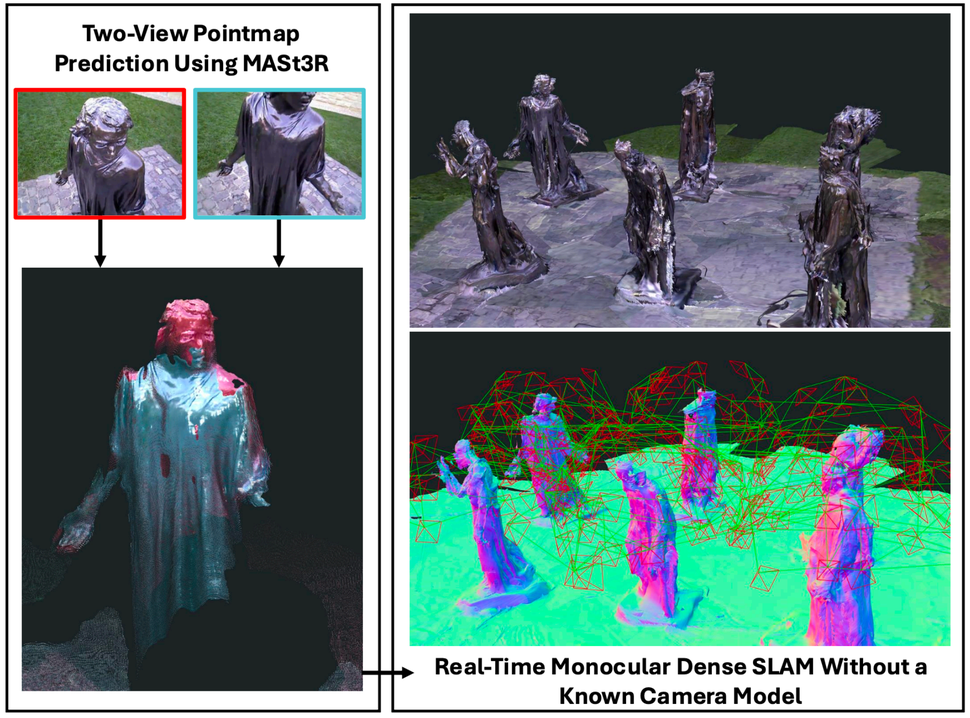
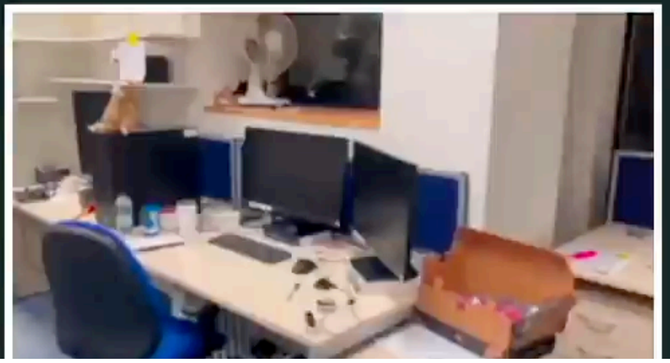
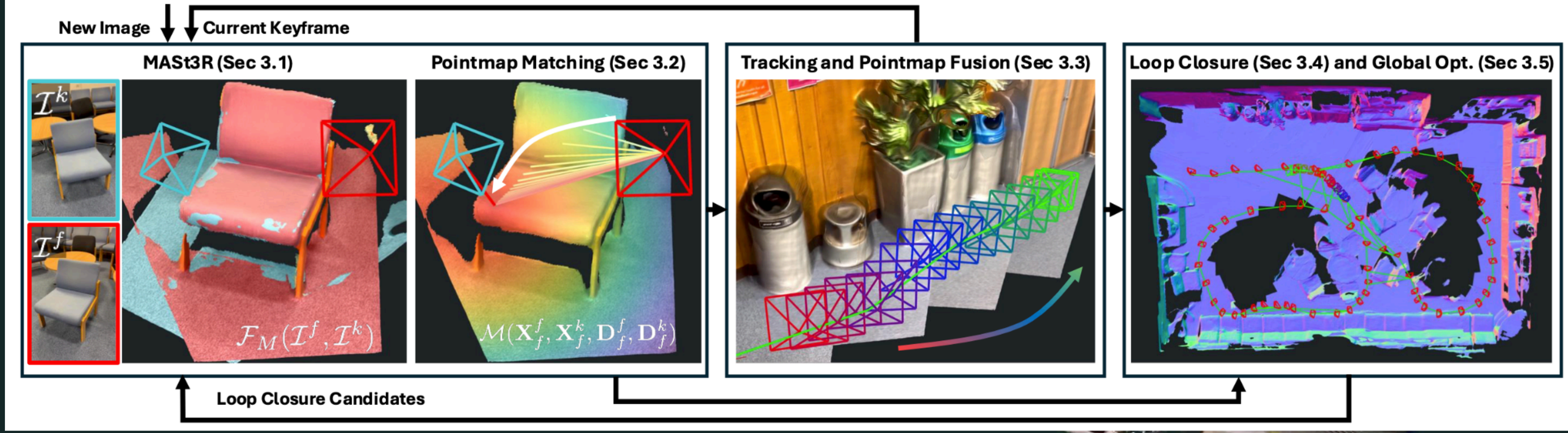


Figure 1. Reconstruction from our dense monocular SLAM system on the Burghers sequence [56]. Using two-view predictions from MASt3R shown on the left, our system achieves globally consistent poses and geometry in real-time without a known camera model.

1. Introduction



Uncalibrated RGB
8x Playback



Murai, Dexheimer et al., MASt3R-SLAM: Real-Time Dense SLAM with 3D Reconstruction Priors, CVPR 2025

Fast3R: Towards 3D Reconstruction of 1000+ Images in One Forward Pass

Jianing Yang[♣♦] Alexander Sax[♣] Kevin J. Liang[♣] Mikael Henaff[♣] Hao Tang[♣]
Ang Cao[♣♦] Joyce Chai[♦] Franziska Meier[♣] Matt Feiszli[♣]

[♣] Meta FAIR [♦] University of Michigan
https://fast3r-3d.github.io

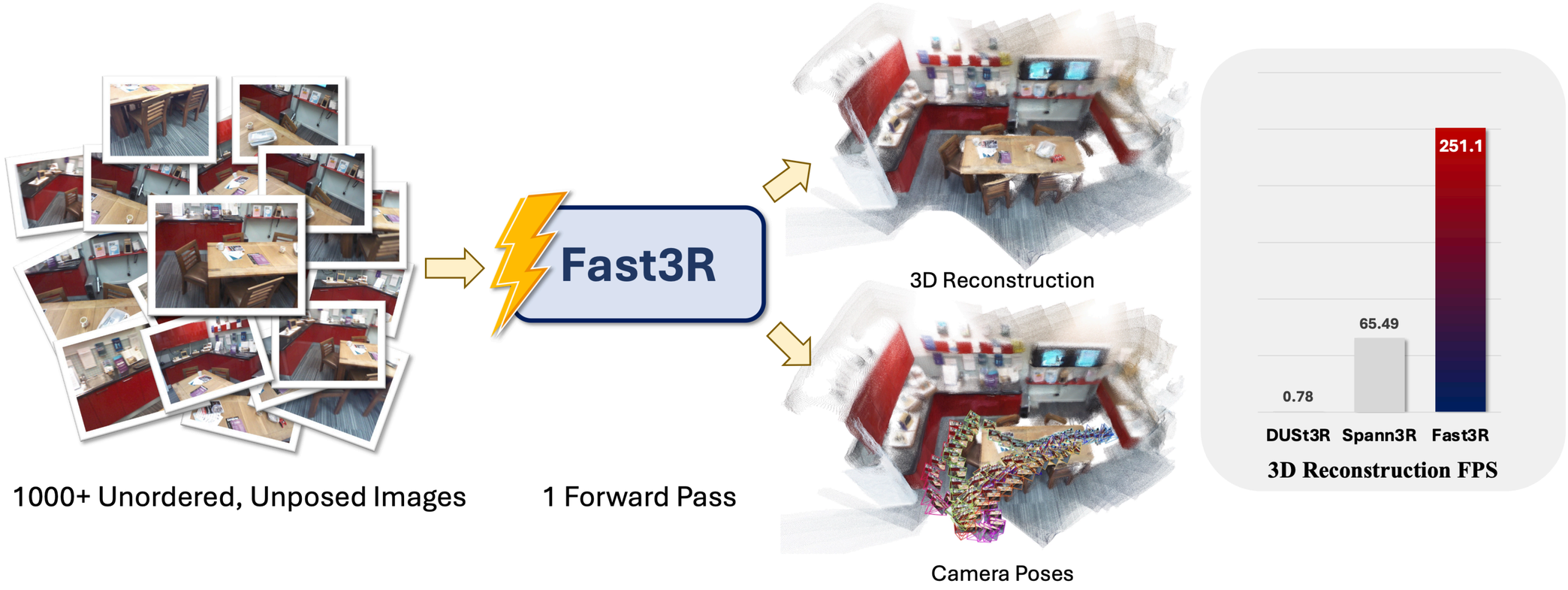


Figure 1. **Fast3R** is a method towards 3D reconstructing 1000+ unordered, unposed images in a single forward pass.

Randomized positional encoding to train on fewer images than test time

Fusion transformer only sees ~200 tokens per input image

Flash Attention to keep cost of fusion transformer under control

Predict both global and local point maps

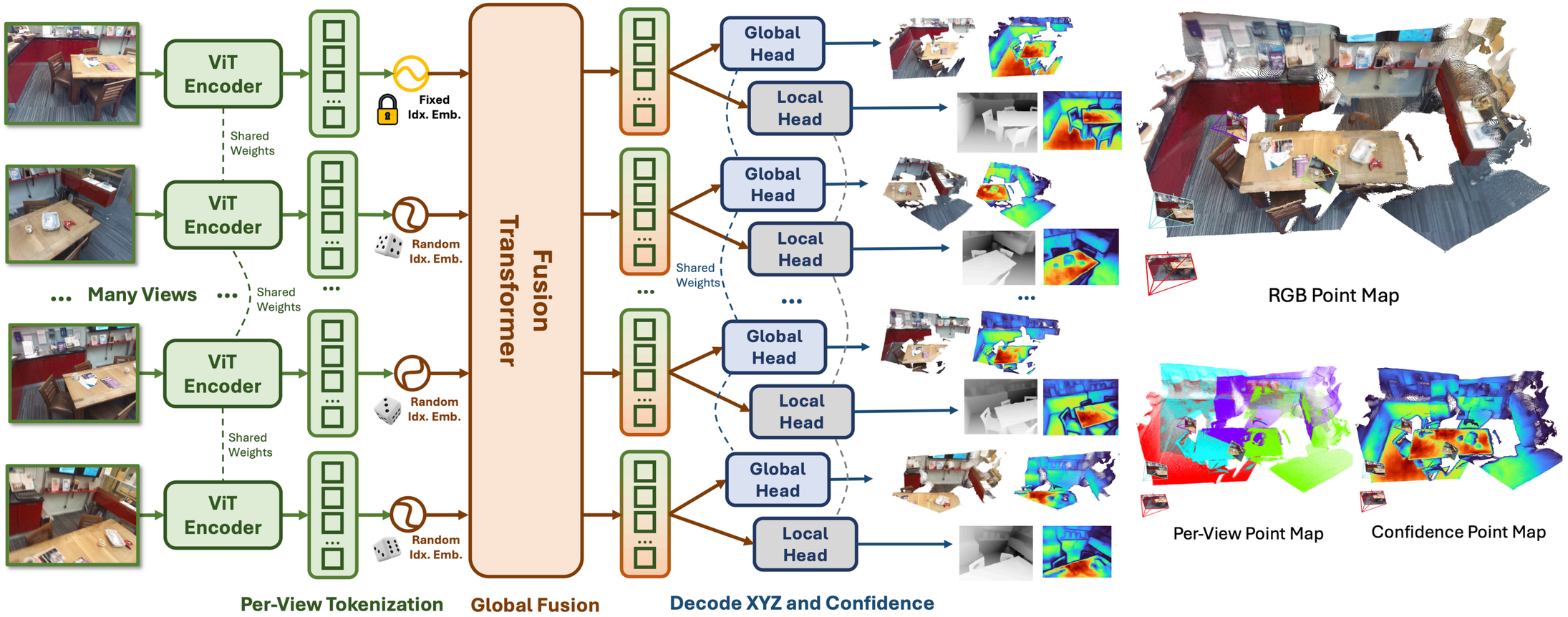


Figure 2. **Model architecture of Fast3R**. Built upon a novel Transformer-based architecture which supports bidirectional information flow, Fast3R is able to process dense input views simultaneously.

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

1651v1 [cs.CV] 14 Mar 2025



Figure 1. **VGGT** is a large feed-forward transformer with minimal 3D-inductive biases trained on a trove of 3D-annotated data. It accepts up to hundreds of images and predicts cameras, point maps, depth maps, and point tracks for all images at once in less than a second, which often outperforms optimization-based alternatives without further processing.

https://vgg-t.github.io/

VGGT

- Multiple images
- Multiple tasks: camera parameter estimation, multi-view depth estimation, dense point cloud reconstruction, point tracking
- Supervised trained on a large number of publicly available datasets with 3D annotations.
- Fast



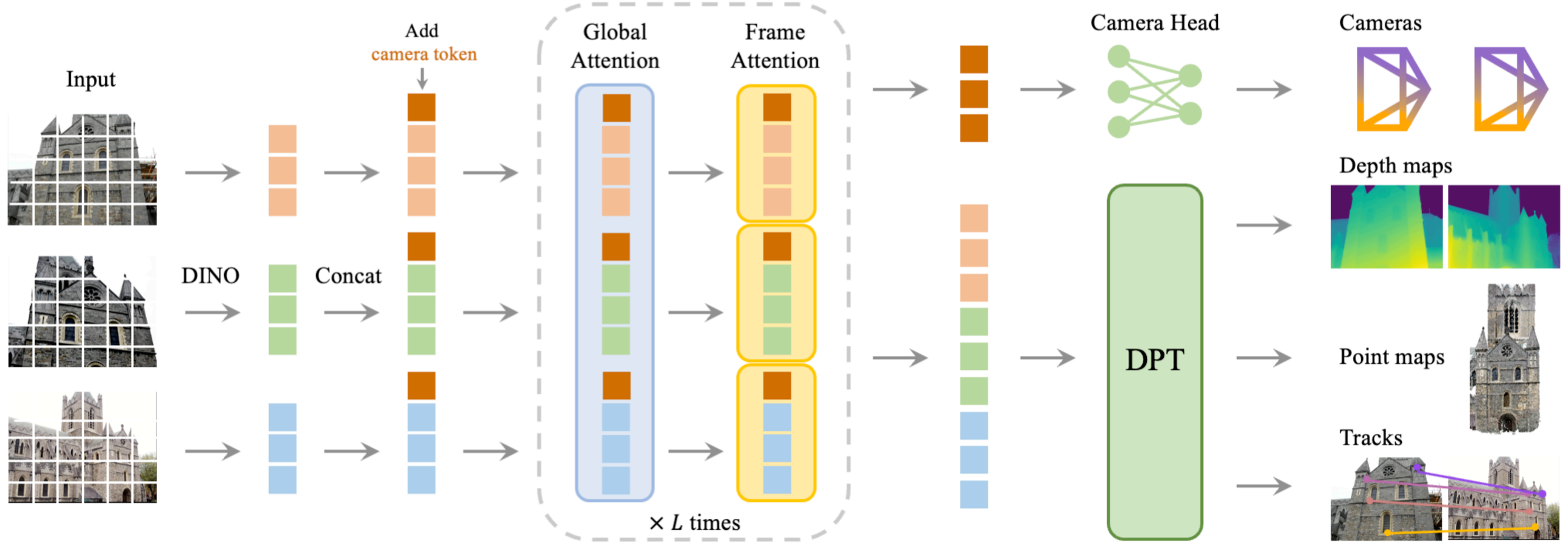
VGGT

- Pretrained features (DINO)
- Specialized attention strategy: alternate local and global
- Multiple heads (camera, depth, point maps, tracks)



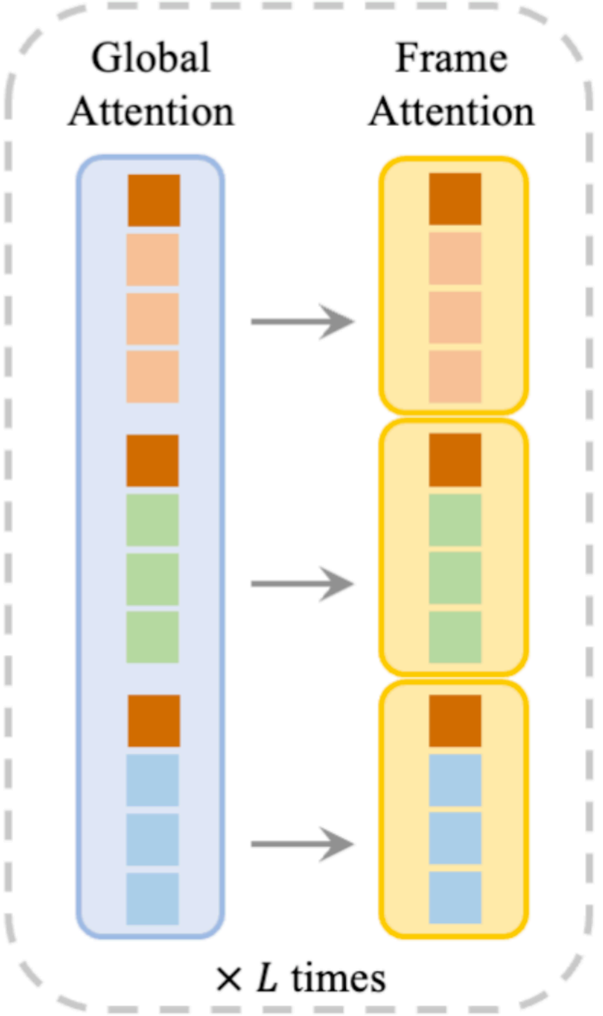
32 Views




Input
Add
camera token
Global
Attention
Frame
Attention
Camera Head
Cameras
DINO
Concat
Depth maps
DPT
Point maps
Tracks
× L times

Alternating attention

- frame-wise self-attention attends to the tokens tIk within each frame separately,
- global self-attention attends to the tokens tI across all frames jointly.
- This strikes a balance between integrating information across different images and normalizing the activations for the tokens within each image
- L= 24 layers of global and frame-wise attention



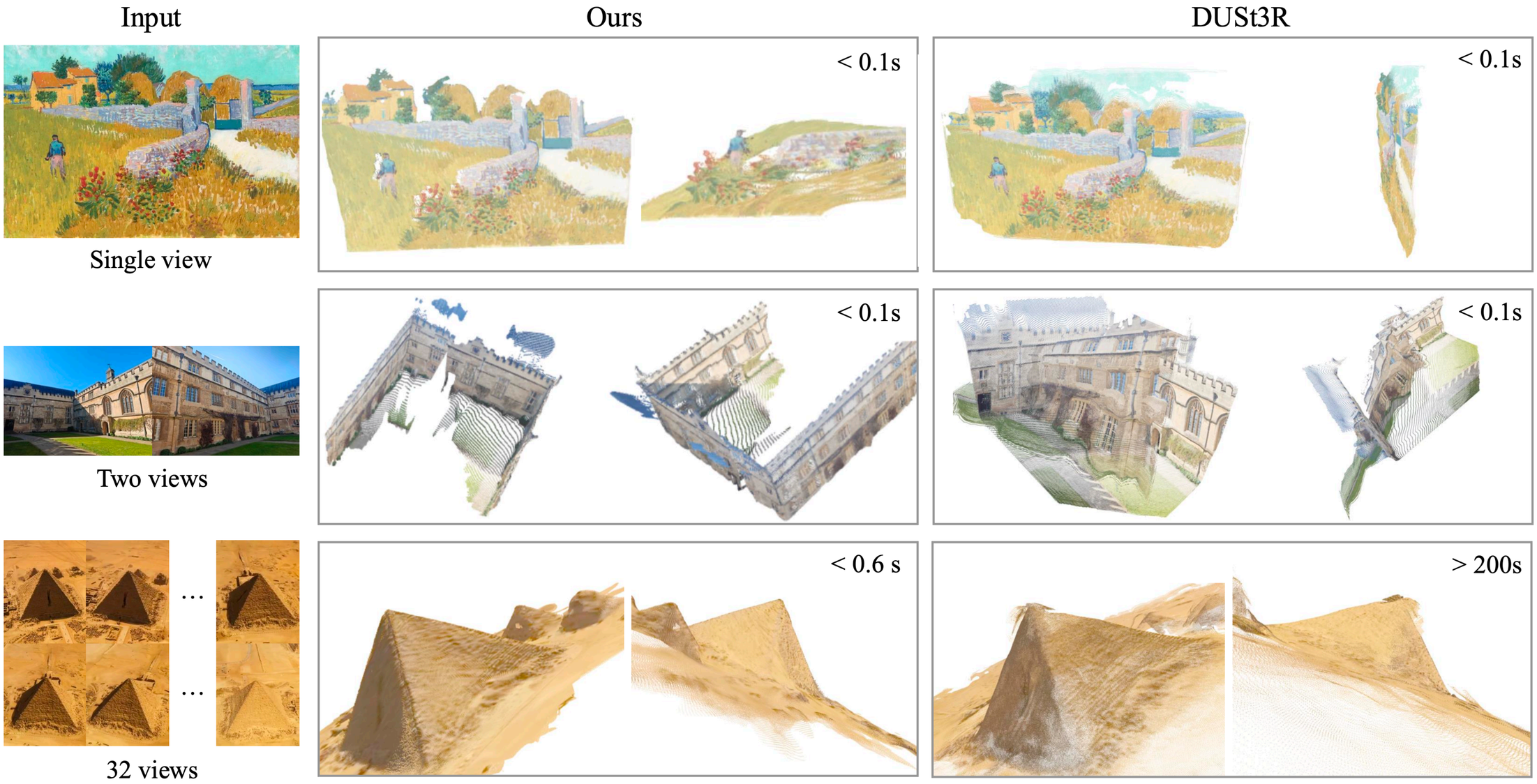


Figure 3. **Qualitative comparison of our predicted 3D points to DUSt3R on in-the-wild images.** As shown in the top row, our method successfully predicts the geometric structure of an oil painting, while DUSt3R predicts a slightly distorted plane. In the second row, our method correctly recovers a 3D scene from two images with no overlap, while DUSt3R fails. The third row provides a challenging example with repeated textures, while our prediction is still high-quality. We do not include examples with more than 32 frames, as DUSt3R runs out of memory beyond this limit.

32 Views





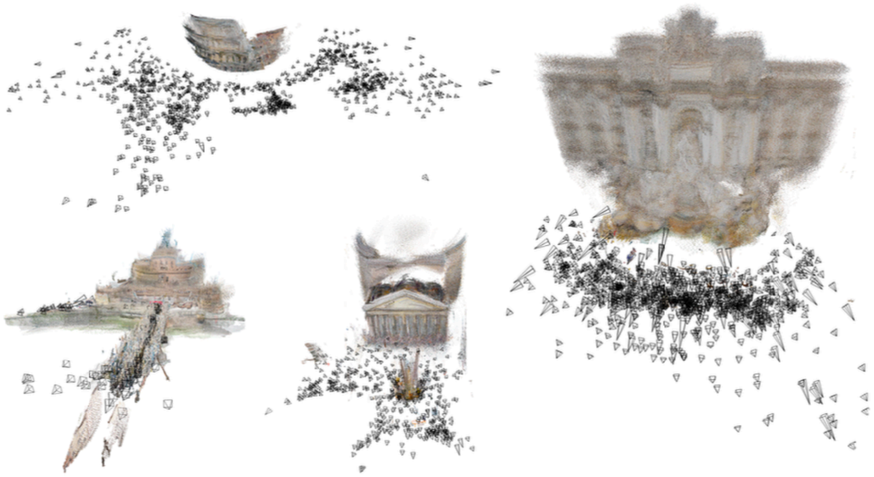
Figure 5. **Visualization of Rigid and Dynamic Point Tracking.** Top: VGGT's tracking module $\mathcal{T}$ outputs keypoint tracks for an unordered set of input images depicting a static scene. Bottom: We finetune the backbone of VGGT to enhance a dynamic point tracker CoTracker [56], which processes sequential inputs.

VGGT extensions

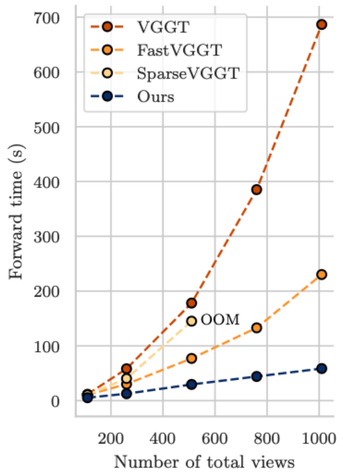
- Further improve scalability, SLAM
- VGGT-Long and VGGT-SLAM.
- TTT3R & VGG-T³
  - Test-Time training for compression of Key-Value

VGG-T^3: Offline Feed-Forward 3D Reconstruction at Scale

Sven Elflein[1,2,3] Ruilong Li[1] Sérgio Agostinho[1]
Zan Gojcic[1] Laura Leal-Taixé[1] Qunjie Zhou[1] Aljosa Osep[1]
[1]NVIDIA [2]Vector Institute [3]University of Toronto



(a) Reconstructing of Rome landmarks: Colosseum, Castel Sant'Angelo, Pantheon and Trevi fountain.



(b) Num. images *vs*. inference time.

Figure 1. **Reconstructing Rome landmarks with 1-minute time budget.** We present VGG-T^3, an offline feed-forward 3D reconstruction method that scales linearly w.r.t. input views (Fig. 1b). As a result, we can reconstruct large scenes from a large number of unposed input views, such as landmarks from tourist-sourced images, in less than a minute via single forward pass (Fig. 1a).

pi^3

- Permutation invariant (no special first view)
- Relative cameras up to similarity transform

π^3: PERMUTATION-EQUIVARIANT VISUAL GEOMETRY LEARNING

Yifan Wang[1,2*] **Jianjun Zhou**[2,3,4*] **Haoyi Zhu**[2,5] **Wenzheng Chang**[1,2] **Yang Zhou**[2,6]
Zizun Li[2,5] **Junyi Chen**[1,2] **Jiangmiao Pang**[2] **Chunhua Shen**[4] **Tong He**[2,3†]

[1]Shanghai Jiao Tong University [2]Shanghai AI Laboratory [3]Shanghai Innovation Institute
[4]Zhejiang University [5]University of Science and Technology of China [6]Fudan University
https://github.com/yyfz/Pi3

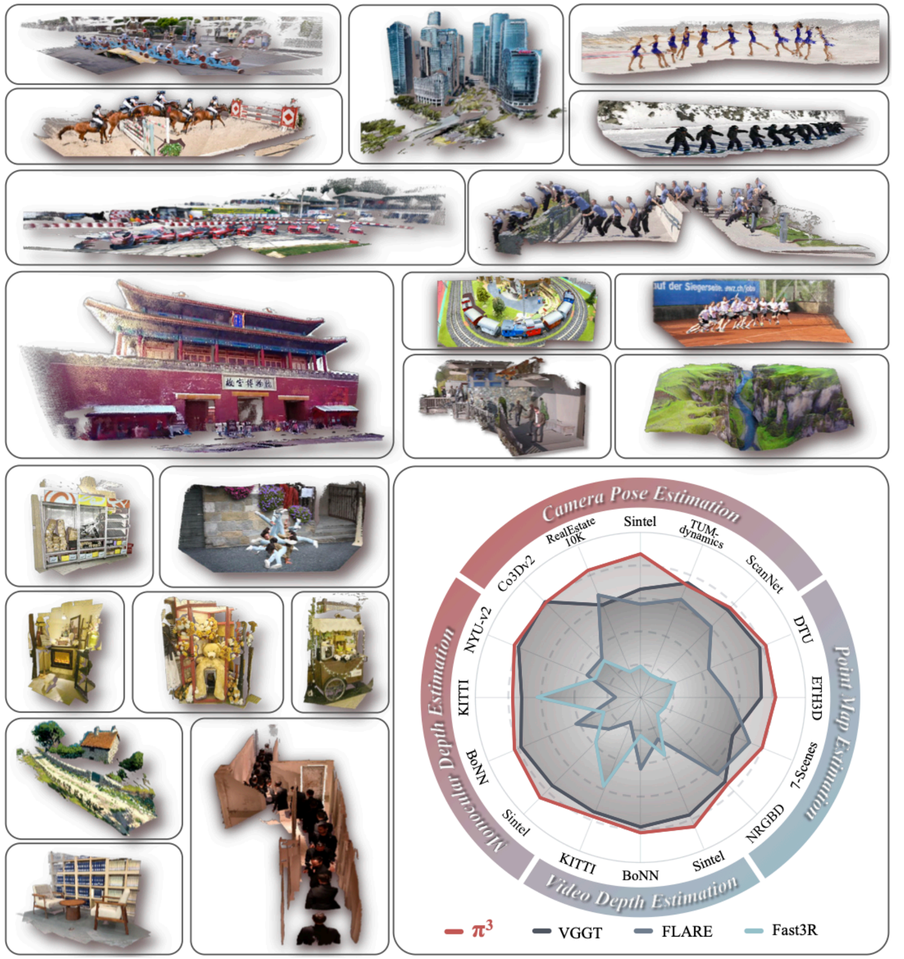


Figure 1: π^3 effectively reconstructs a diverse set of open-domain images in a feed-forward manner, encompassing various scenes such as indoor, outdoor, and aerial-view, as well as cartoons, with both dynamic and static content.

LoGeR: Long-Context Geometric Reconstruction with Hybrid Memory

Junyi Zhang[1,2] **Charles Herrmann**[1,*] **Junhwa Hur**[1,*] **Chen Sun**[1] **Ming-Hsuan Yang**[1]
Forrester Cole[1] **Trevor Darrell**[2] **Deqing Sun**[1,†]

[1]Google DeepMind [2]UC Berkeley

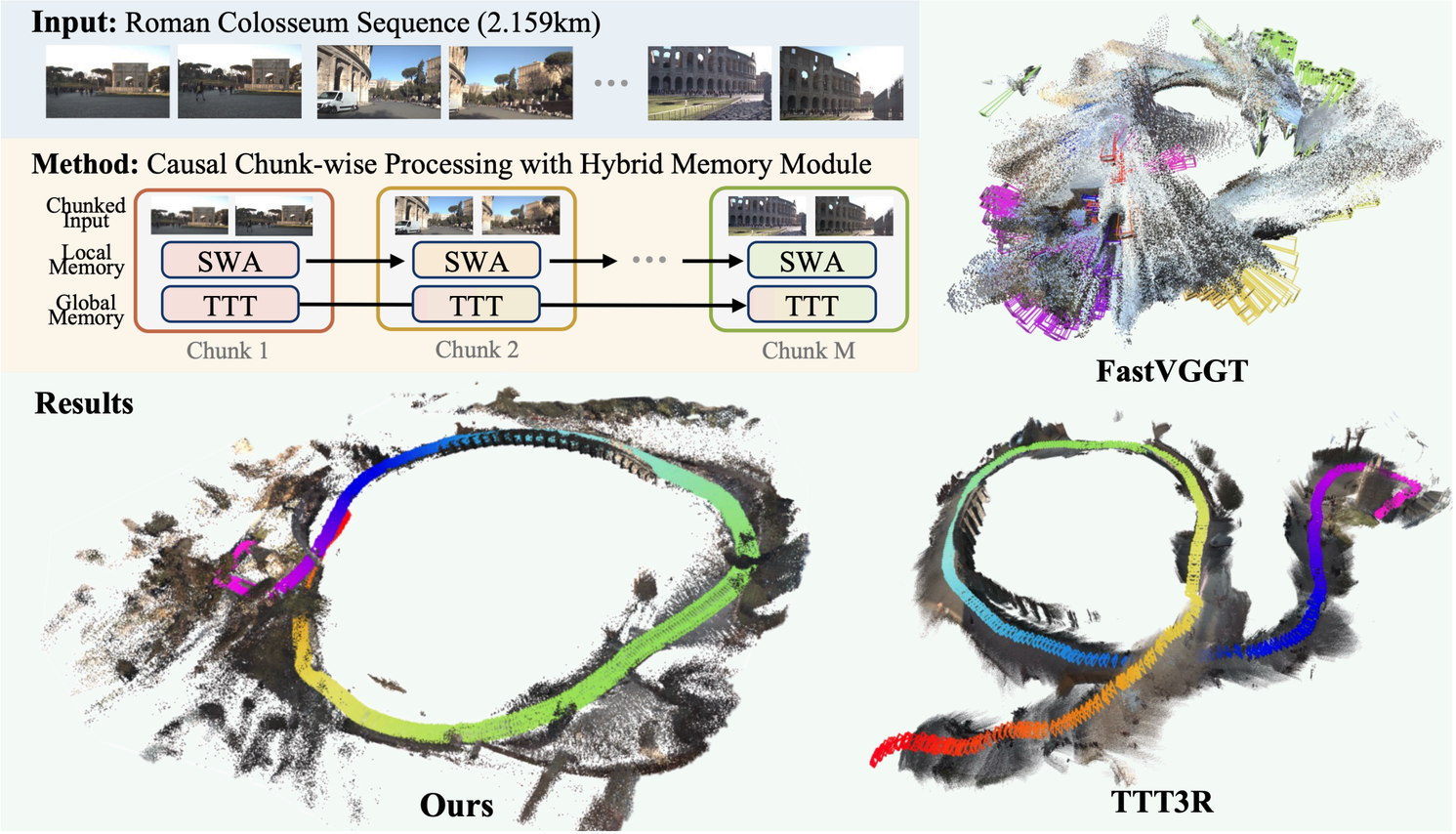


Figure 1. **Our proposed method and visual comparison.** For very long videos, we advocate chunk-based processing where bidirectional attention handles intra-chunk reasoning with inter-chunk alignment handled by our proposed *hybrid memory module*, composing Sliding Window Attention (SWA) for detailed local memory and Test-Time Training (TTT) for compressed global context. LoGeR shows clear improvement over prior methods on expansive VBR (Brizi et al., 2024) dataset, yielding superior loop closures and finer geometric details.

https://loger-project.github.io/

Conclusion on correspondence and 3D vision

For any given *task* (correspondence estimation, 3D reconstruction, depth prediction…): collect a supervised training set and deploy a generalist-as-possible architecture.

To collect the dataset, you either simulate or you rely on hand-crafted prior work.

Network architecture and training details matter, but you should get a sign of life even if you get these details slightly wrong.

You will notice once you try to apply these to real-world problems (robotics, self-driving, …) that there is always more complexity and generality hidden away!

Challenges and frontier

scalability, long context

- Test-Time training helps, but see also LoGER

Beyond shape + texture

- More complex material appearance
- Relightability
- Motion

Less supervision

Removing inductive biases

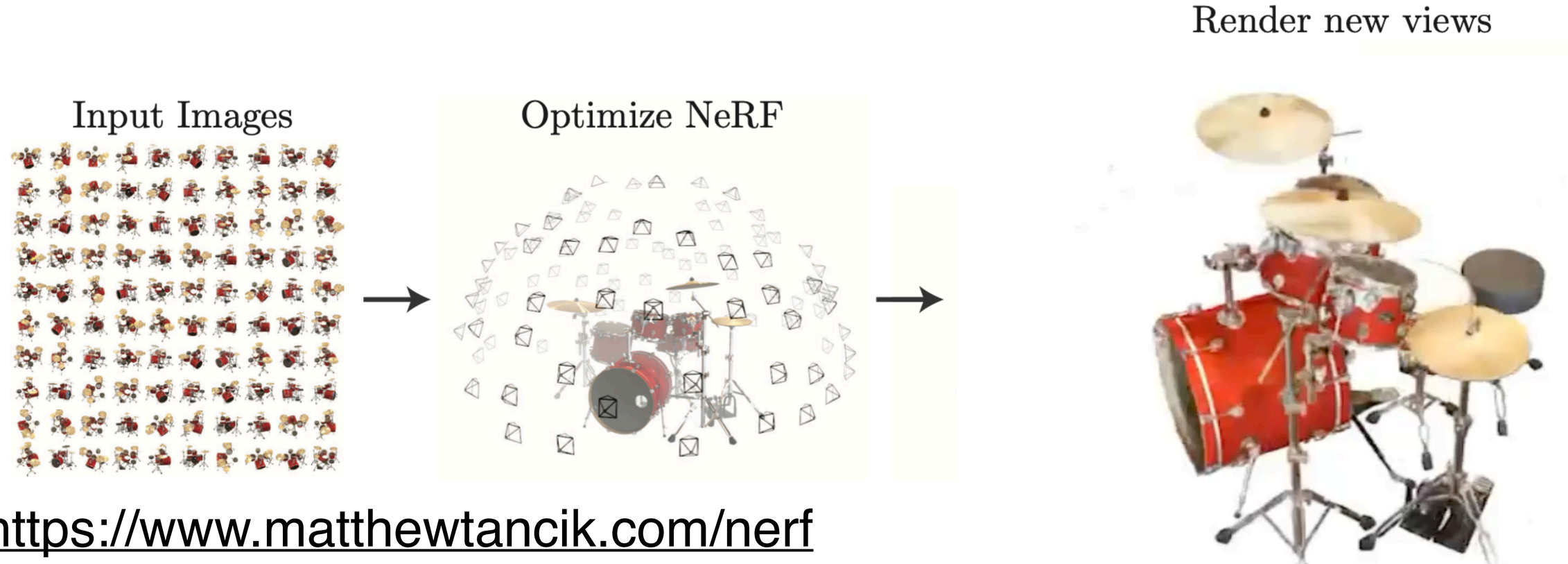
Novel View Synthesis

Novel View Synthesis

Input : one or more images of a 3D scene, often with camera calibration

Output new views of the same scene from novel viewpoints (but same illumination, etc.)

3D can be used but is not part of the task



https://www.matthewtancik.com/nerf

Why NVS matters

In its own right (e.g. e-commerce, urban mapping, special effects, etc.)

Paradigm of link between 2D and 3D
but has good data thanks to cycle back to 2D

Because it’s fun, because we can

Campus Dataset
22,000 images
1.3km

normal speed video in supplemental



https://www-sop.inria.fr/reves/Basilic/2024/KMKWLD24/

https://occlugaussian.github.io/

State of the art before deep learning

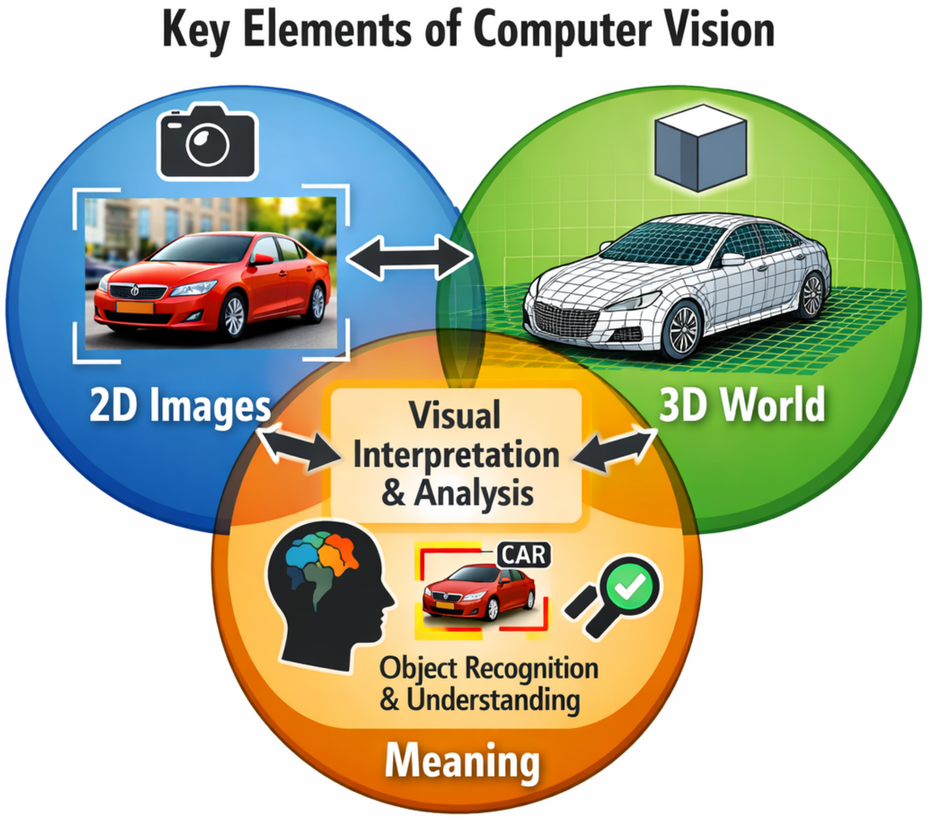
Back to Computer Vision ingredients and task

3D->2D : rendering (computer graphics

2D -> 3D : 3D reconstruction

2D -> another 2D but from a. Different viewpoint (Novel View Synthesis)

For which 2D->3D can be an intermediate task or not (fake task?)



State of the art before deep learning: 3D->2D

Hard surfaces

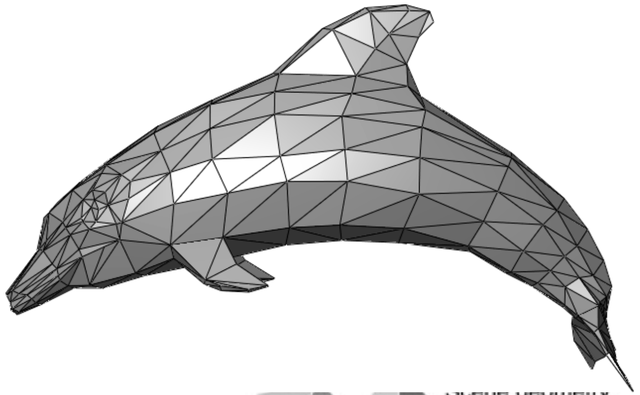
- Triangles, or maybe polynomial/rational surfaces
- Textures + BRDF (what fraction of incoming light at a given angle is reflected in another angle.
- Real-time rendering does rasterization (project triangles onto images), direct illumination, plus hacks for fake indirect/global illumination
- High-quality rendering: ray tracing (send rays through image to 3D scene), full global illumination by sampling indirect rays.
- Poor handling of transparency and fuzziness (because requires ordering)
- Not differentiable (yet)
- Inputs aree hard/costly to get (100s of artists), hard to match reality
- Underlying models (lighting, animation) are sophisticated but not perfect

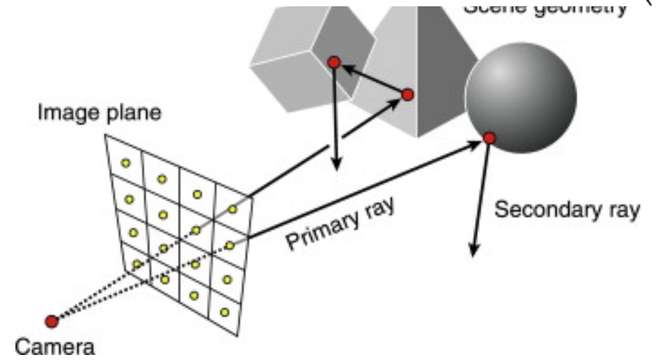
Volumetric rendering

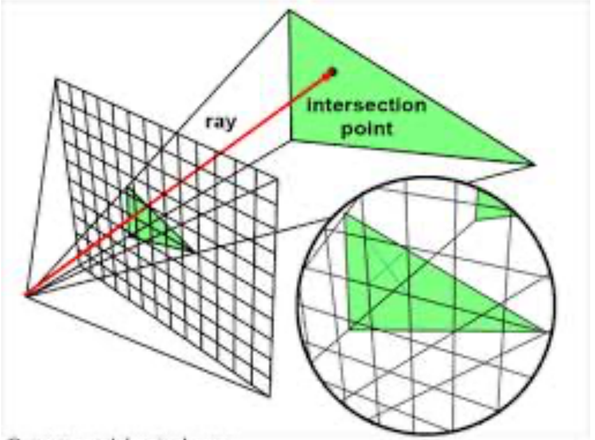
- Limited to smoke, fog, and fluid simulation

Point-based graphics

- Limited use for rendering
- Common for 3D reconstruction (because 3D scanners/lidar), (limited) work on geometry processing





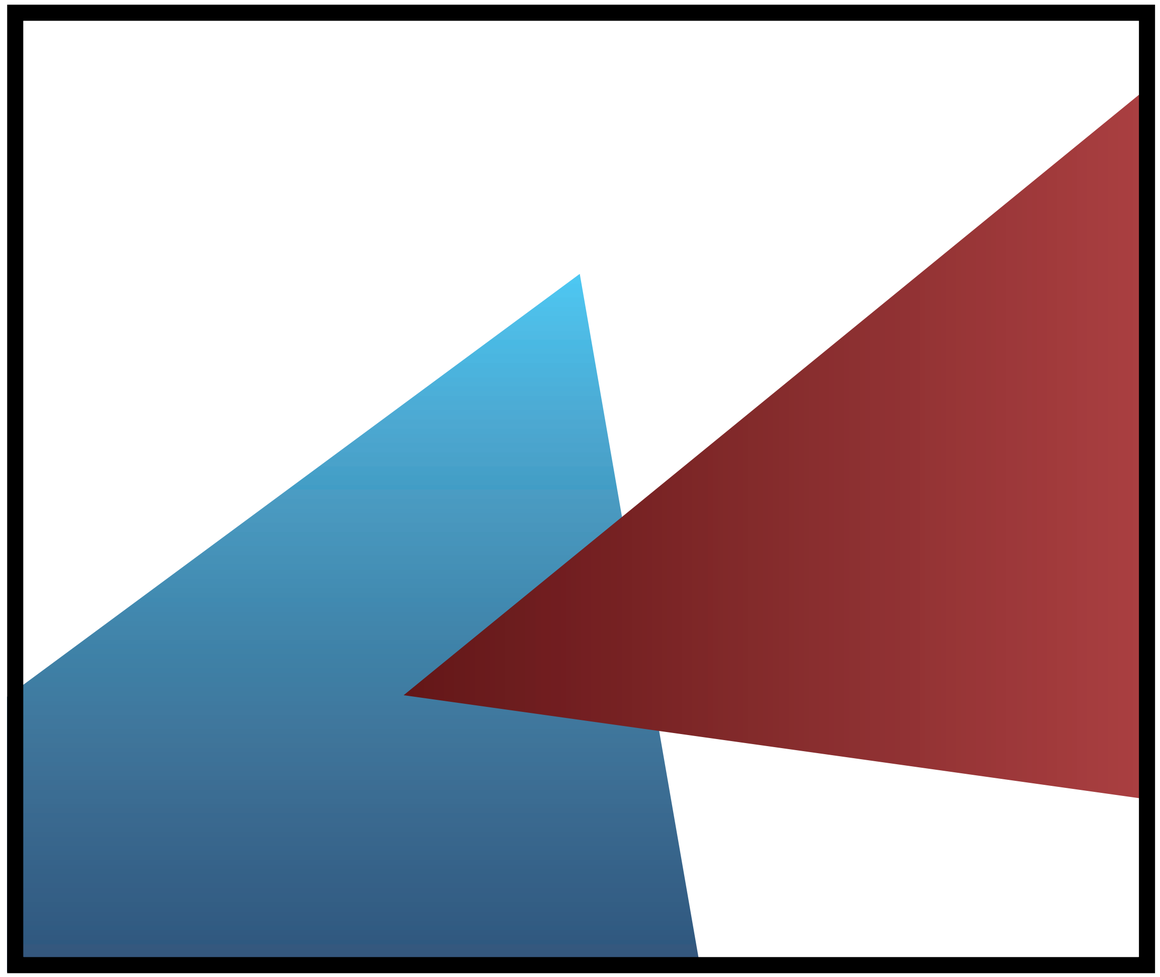


Rendering Differentiability Challenge: visibility

- Discontinuities don’t play well with derivatives

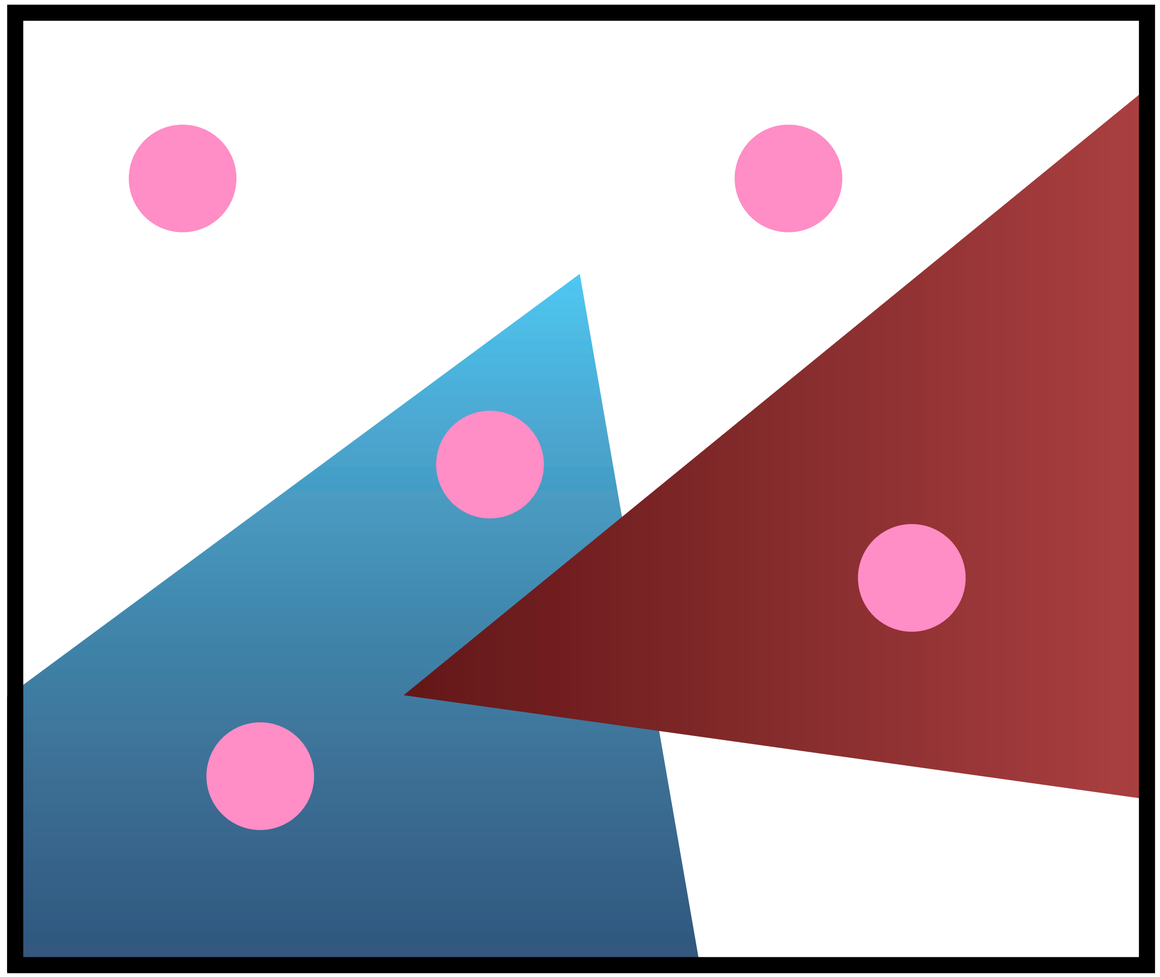


Toy example: two triangles



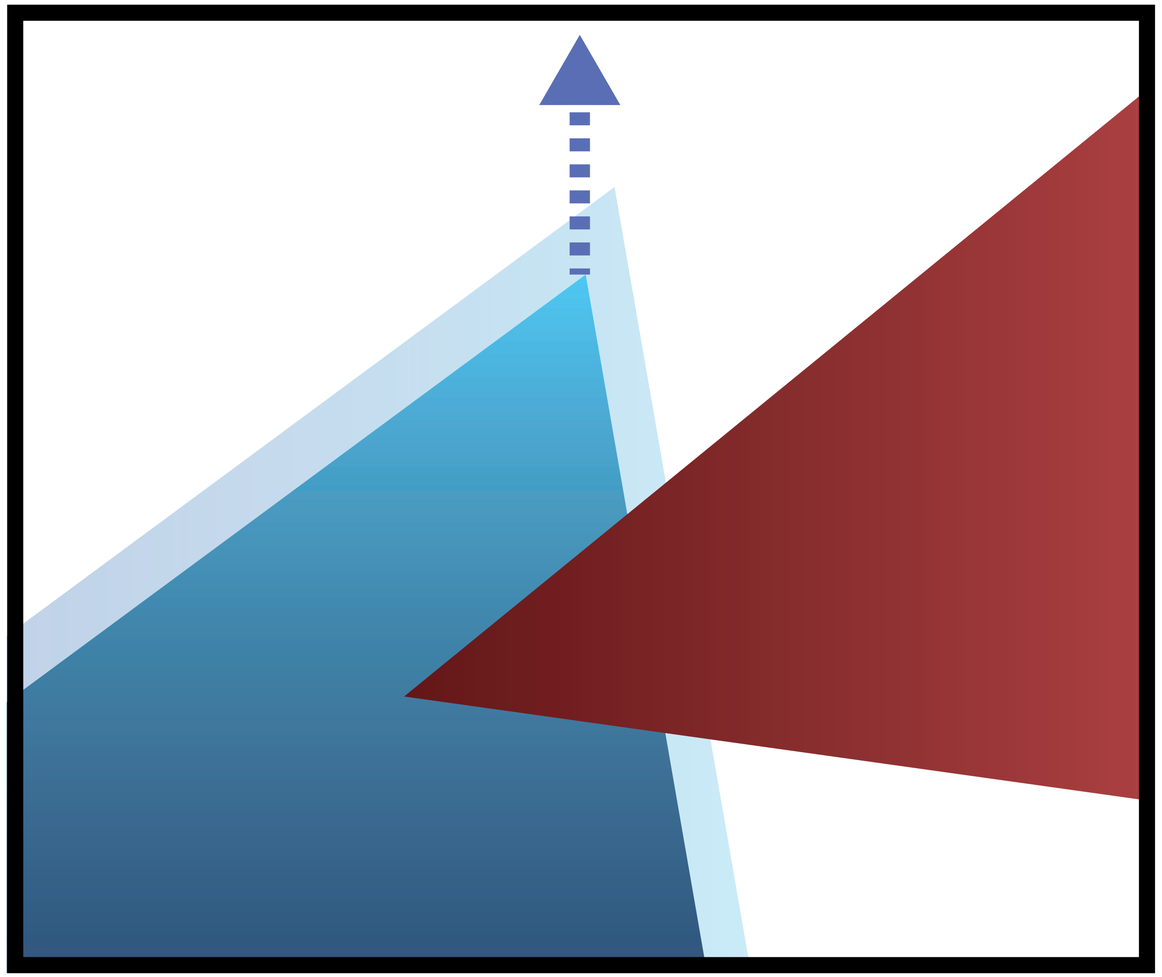
pixel

Rendering = sampling pixels



Computing derivatives

Q:
color change
when blue triangle
moves up?

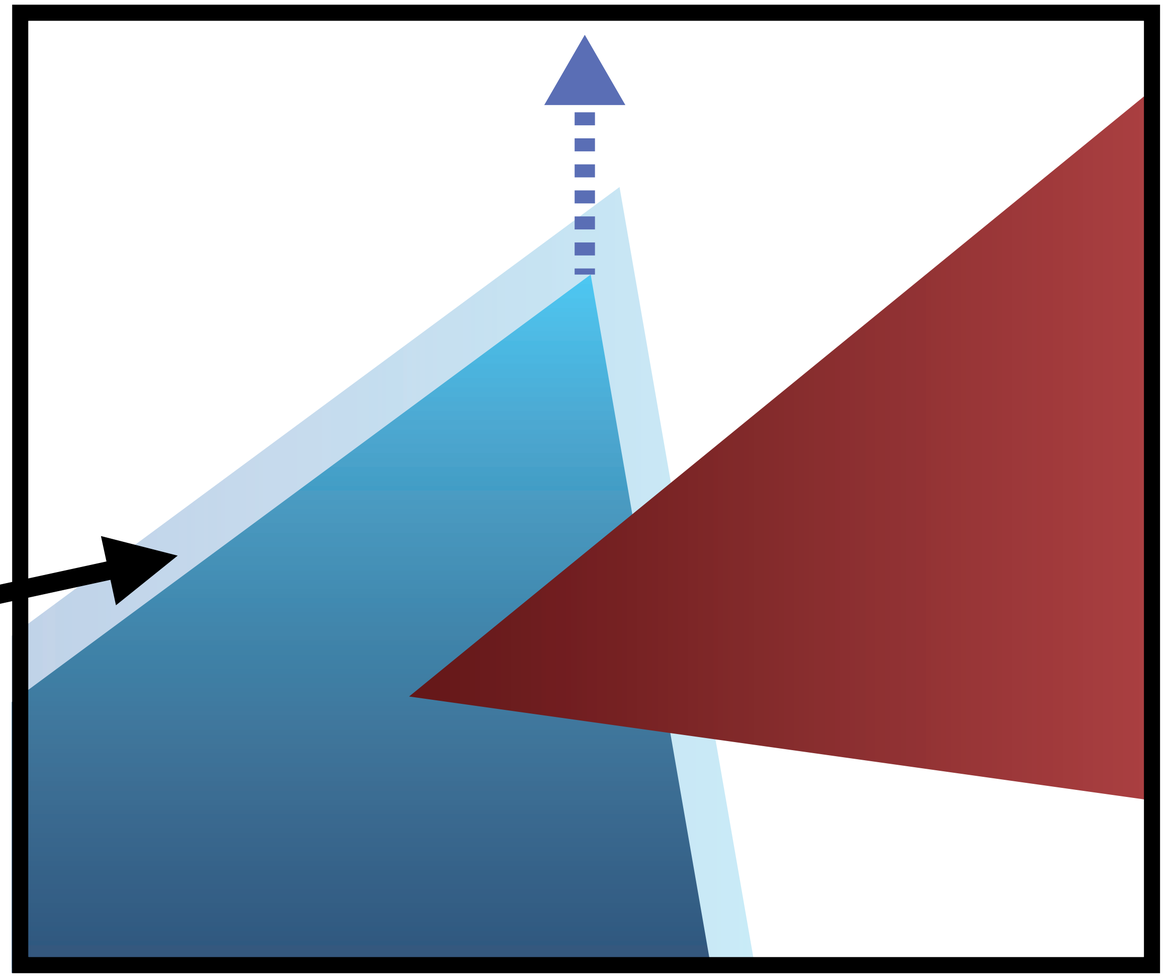


pixel

Computing derivatives

Q:
color change
when blue triangle
moves up?

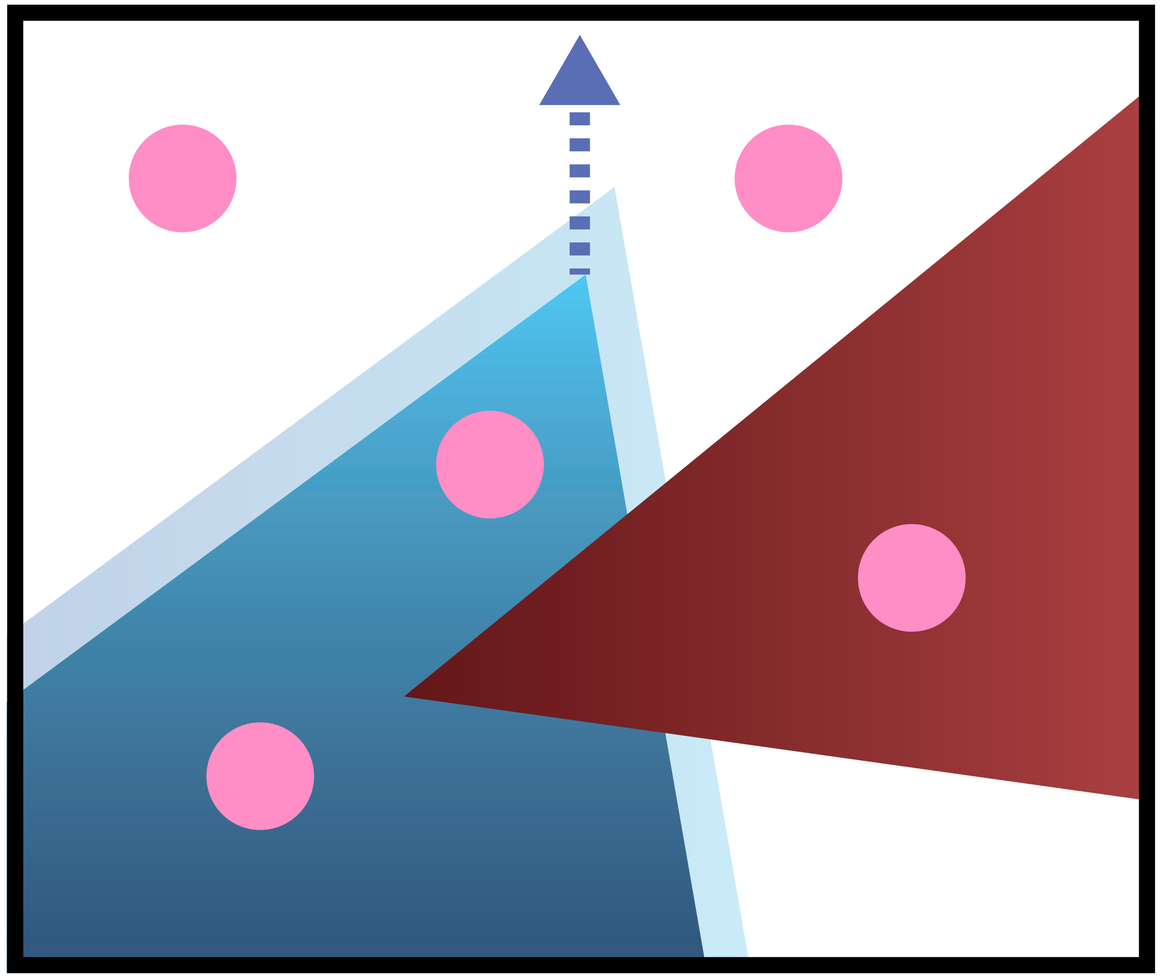
A:
more blue
less white



pixel

Point samples only give local information

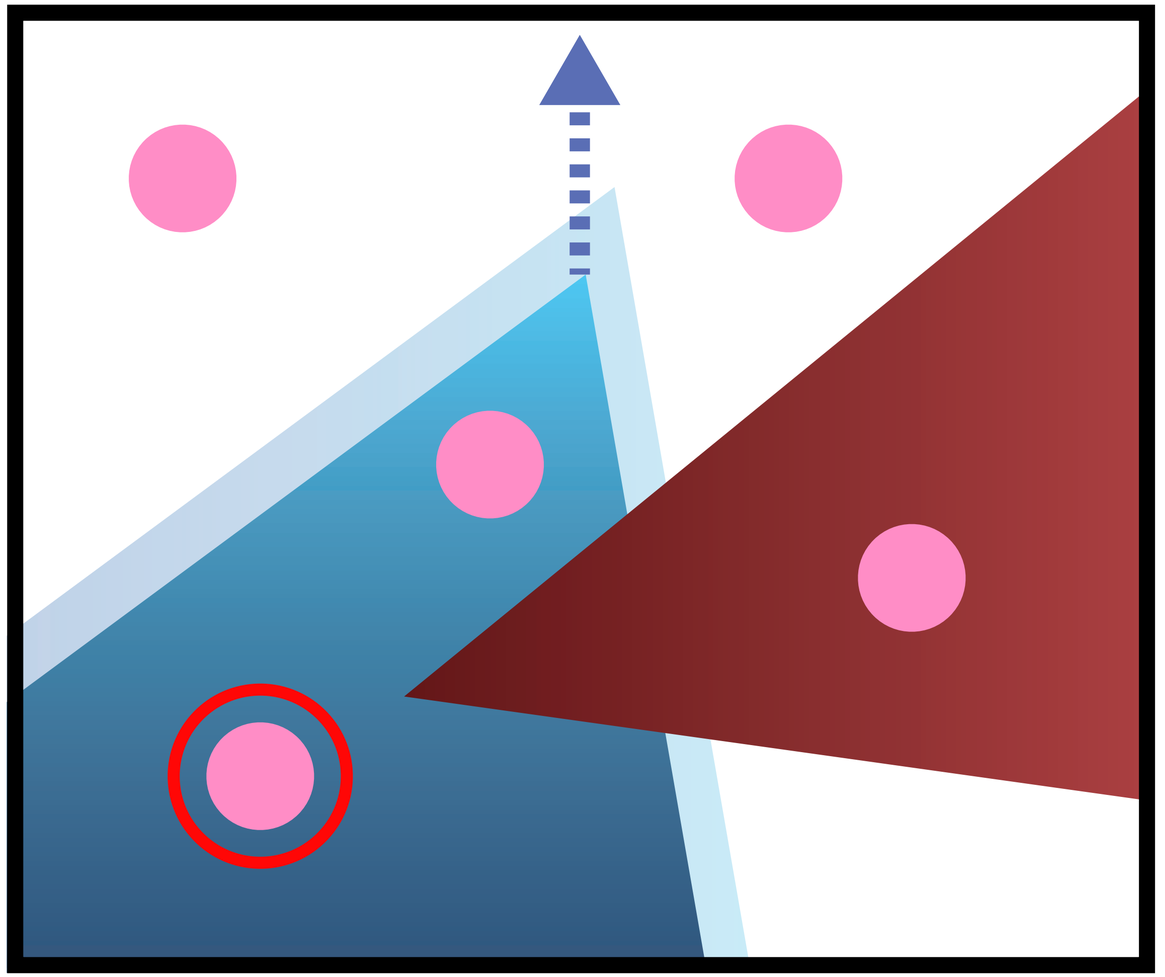
color change
when blue triangle
moves up?



pixel

Point samples only give local information

color change
when blue triangle
moves up?



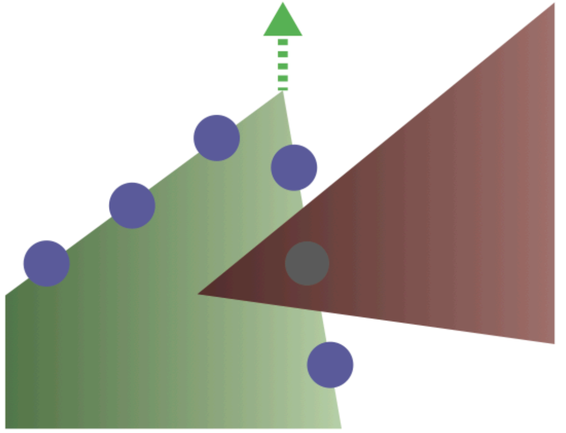
pixel

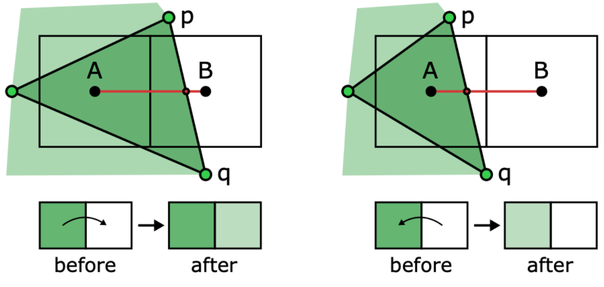
Plus numerically very stiff, very sparse real gradient

Recap challenges for differentiable rendering of hard surfaces



- Discontinuities make life hard
- Even if pixel is defined as (differentiable) integral of discontinuous function, it gets computed by sampling, which misses the contributions of discontinuities.
  - Methods exist (see e.g. https://people.csail.mit.edu/tzumao/diffrt/ ; https://people.csail.mit.edu/sbangaru/projects/was-2020/index.html ; https://arxiv.org/pdf/2011.03277 (used for nvdiffrast) but needs extra work
  - For serious mathematical treatment: https://dl.acm.org/doi/10.1145/3729263
- Even when you can get them, derivatives are sparse and spiky and make optimization hard
- **Bottom line : the link between 3D and 2D is not naturally friendly to training NN or solving inverse problems with SGD**
  - **Especially with traditional hard surface representations**

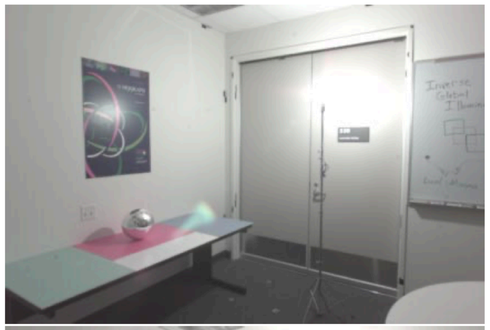


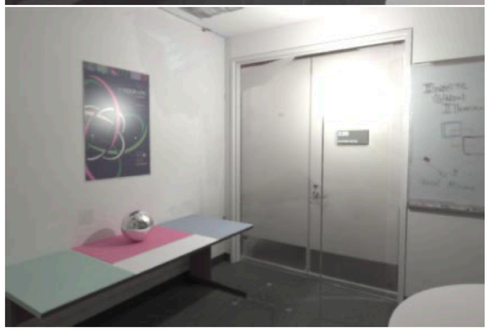


State of the art before deep learning: 2D->3D

- **3D reconstruction**
  - Focus on 3D shape alone,
    sometimes add texture (with illumination baked in, not intrinsic color of objects)
  - PMVS : SIFT like features, Camera calibration, SfM, dense matches, often followed my mesh refinement
- **Inverse rendering, inverse global illumination**
  - Slightly different problem statement: must be relightable
  - 3D shape + surface appearance (texture, BRDF) +lighting
  - Too ill posed, no gradient, not enough data/input views?







State of the art before deep learning: 2D->2D Novel View Synthesis (Image-Based Rendering)

- **Basic solution: 2D->3 followed by 3D->2D, often Lambertian-only**
- To capture view-dependent colors (BRDF effect given illumination), use the color from different inputs views (view-dependent texture mapping, Debevec et al. 97)
- On the other end of the spectrum, directly model 4D light field (per ray color)
  rays/lines are 4D (e.g. intersection with 2 planes)
- **In practice, typically combination of 3D model and the input views,**
  with different emphasis on the 3D vs the views vs 4D space of rays/lines.
  - Unstructured lumigraph: more emphasis on views
  - Surface light fields : more emphasis on 3D model





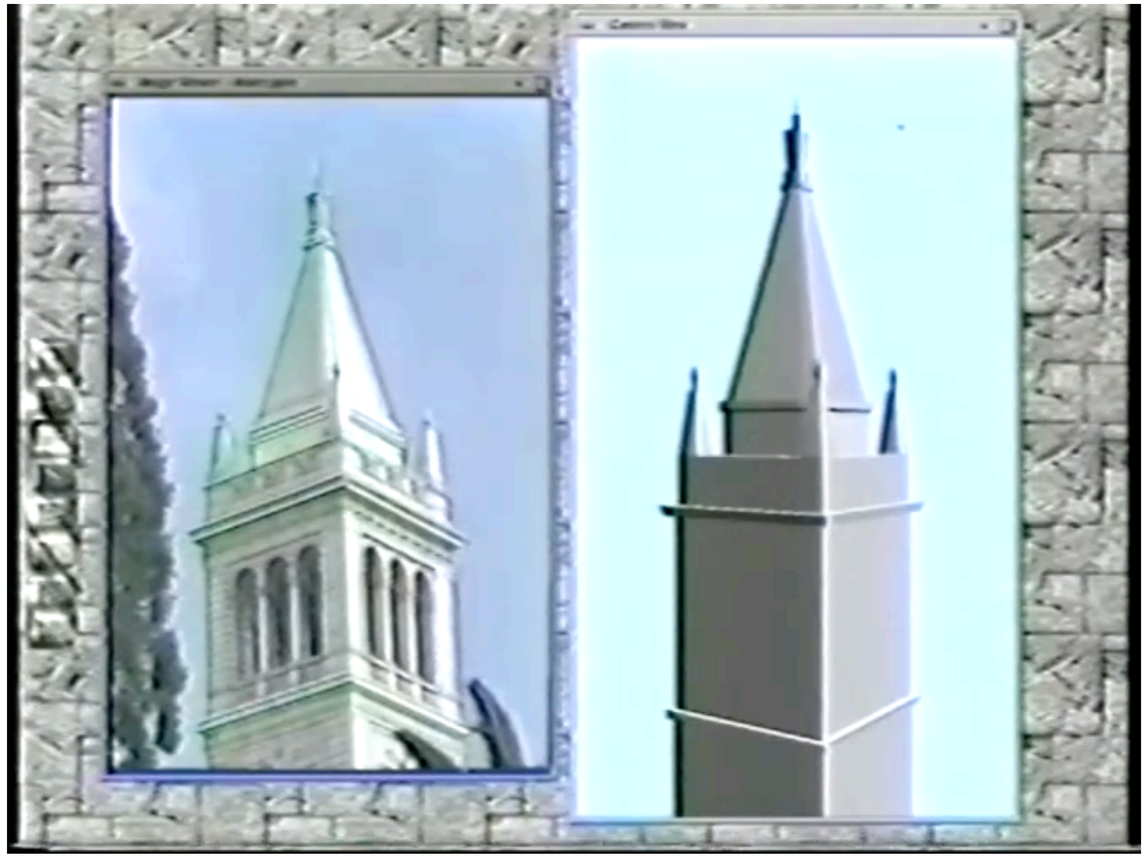
Mesh + view-dependent texture

Modeling and Rendering Architecture from Photographs: A hybrid geometry- and image-based approach

Paul E. Debevec Camillo J. Taylor Jitendra Malik

https://people.eecs.berkeley.edu/~malik/papers/debevecTM96.pdf





4D light field

Light Field Rendering

Marc Levoy and Pat Hanrahan
Computer Science Department
Stanford University

4D representation of all lines that leave a scene

2-plane parameterization

Dense samples + quadrilinear reconstruction

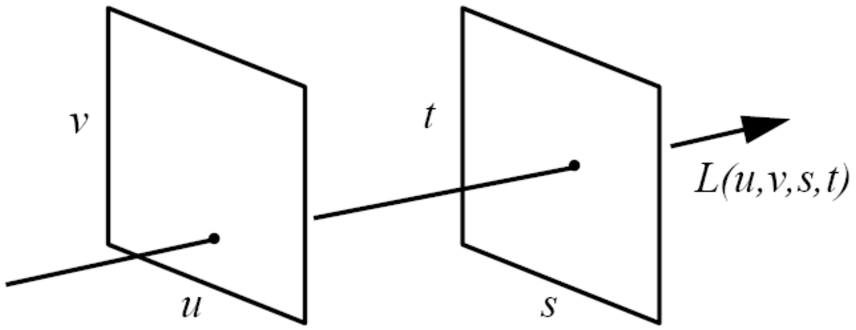


Figure 1: The light slab representation.

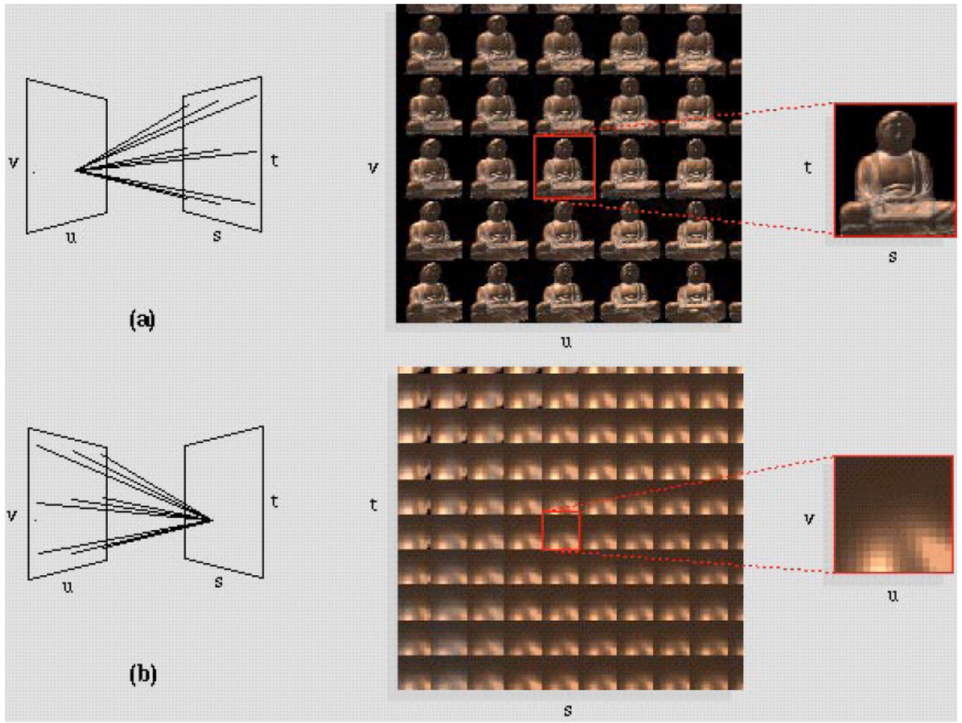


Figure 6: Two visualizations of a light field. (a) Each image in the array represents the rays arriving at one point on the uv plane from all points on the st plane, as shown at left. (b) Each image represents the rays leaving one point on the st plane bound for all points on the uv plane. The images in (a) are off-axis (i.e. sheared) perspective views of the scene, while the images in (b) look like reflectance maps. The latter occurs because the object has been placed astride the focal plane, making sets of rays leaving points on the focal plane similar in character to sets of rays leaving points on the object.

https://graphics.stanford.edu/papers/light/light-lores-corrected.pdf

(Unstructured) Lumigraph

Add a 3D proxy shape to do sharper ray reconstruction (~focus)

The Lumigraph

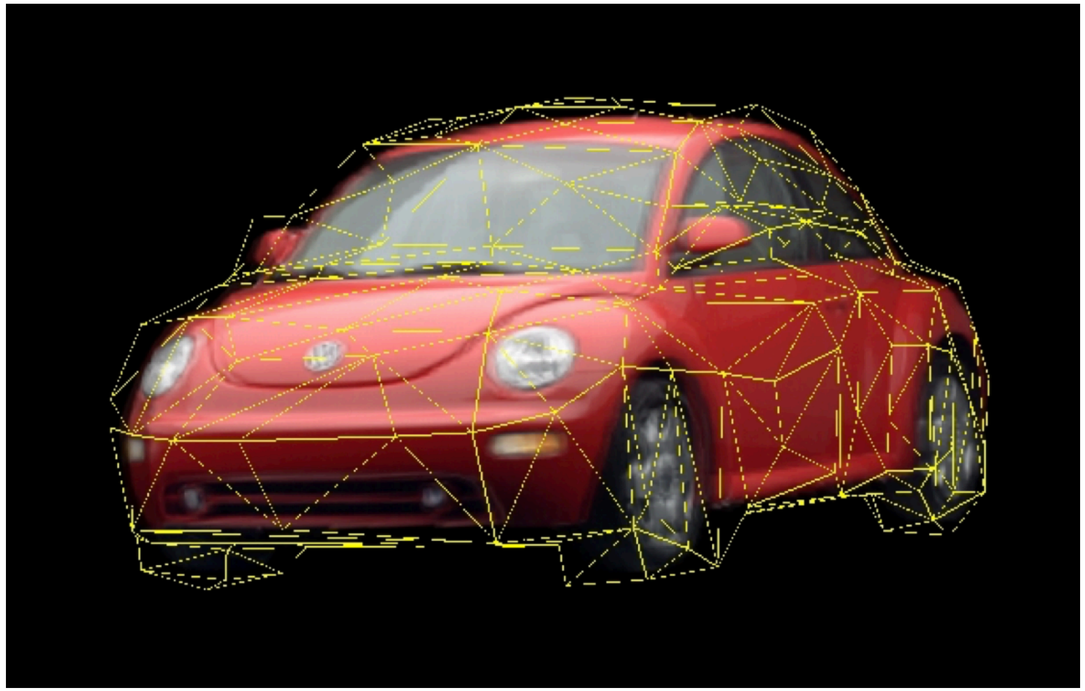
Steven J. Gortler, Microsoft Research
Radek Grzeszczuk, University of Toronto*
Richard Szeliski, Microsoft Research
Michael F. Cohen, Microsoft Research

Unstructured Lumigraph Rendering

Chris Buehler, Michael Bosse, Leonard McMillan, MIT Computer Graphics Group
Steven Gortler, Harvard University
Michael Cohen, Microsoft Research

Unstructured Light Fields

Abe Davis[†1,2] Fredo Durand[‡1] Marc Levoy[2]





https://graphics.stanford.edu/papers/unstructuredLFs/davis-unstructuredLFs-eg12.pdf

https://cs.harvard.edu/~sjg/papers/ulr.pdf

Surface light field

2D surface + 2D lumisphere at each point

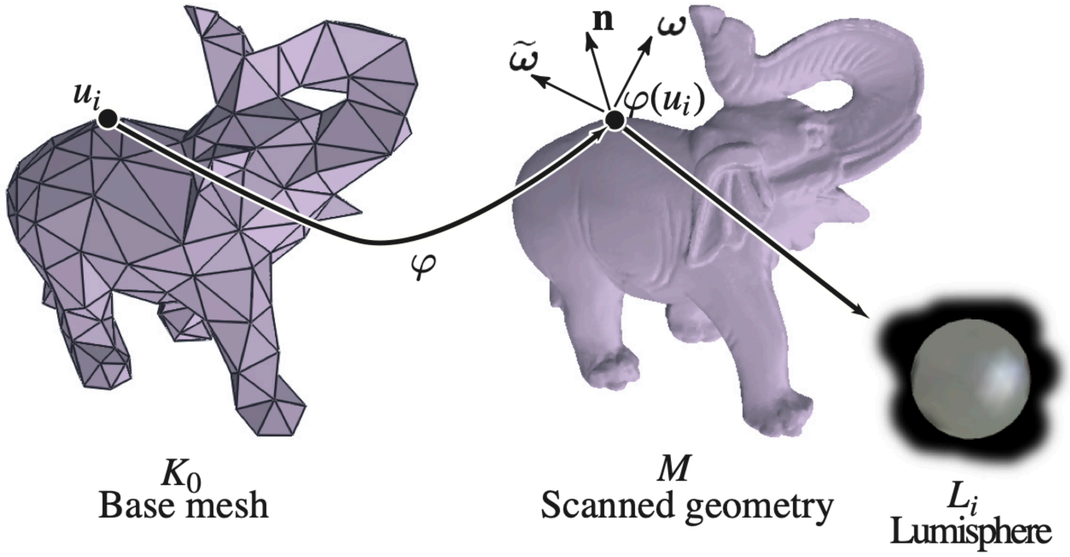


Figure 2 Representation of the surface light field. Points u on the base mesh, K_0, are mapped to the geometric surface, M, by φ. The lumisphere, L_i at the grid point u_i, represents the radiance leaving surface point $\varphi(u_i)$. Directions are denoted by ω, or $\tilde{\omega}$ after reflection through the surface normal $\mathbf{n}$ as described in Section 4.4.



https://grail.cs.washington.edu/projects/slf/

State of the art before deep learning: 2D->2D Novel View Synthesis (Image-Based Rendering)

- **Basic solution: 2D->3 followed by 3D->2D, often Lambertian-only**
- To capture view-dependent colors (BRDF effect given illumination), use the color from different inputs views (view-dependent texture mapping, Debevec et al. 97)
- On the other end of the spectrum, directly model 4D light field (per ray color)
  rays/lines are 4D (e.g. intersection with 2 planes)
- **In practice, typically combination of 3D model and the input views,**
  with different emphasis on the 3D vs the views vs 4D space of rays/lines.
  - Unstructured lumigraph: more emphasis on views
  - Surface light fields : more emphasis on 3D model





State of the art before deep learning: 2D->2D Novel View Synthesis (Image-Based Rendering)

- **Basic solution: 2D->3 followed by 3D->2D, often Lambertian-only**
- To capture view-dependent colors (BRDF effect given illumination), use the color from different inputs views (view-dependent texture mapping, Debevec et al. 97)
- On the other end of the spectrum, directly model 4D light field (per ray color)
  rays/lines are 4D (e.g. intersection with 2 planes)
- **In practice, typically combination of 3D model and the input views,**
  with different emphasis on the 3D vs the views vs 4D space of rays/lines.
  - Unstructured lumigraph: more emphasis on views
  - Surface light fields : more emphasis on 3D model





Recap: State of the art before

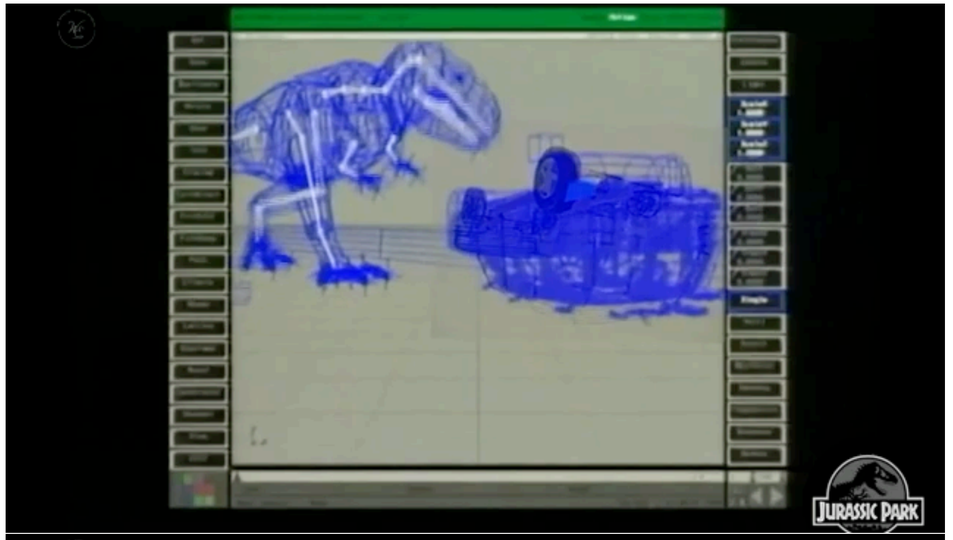
3D -> 2D : doing great, games, movies, etc.

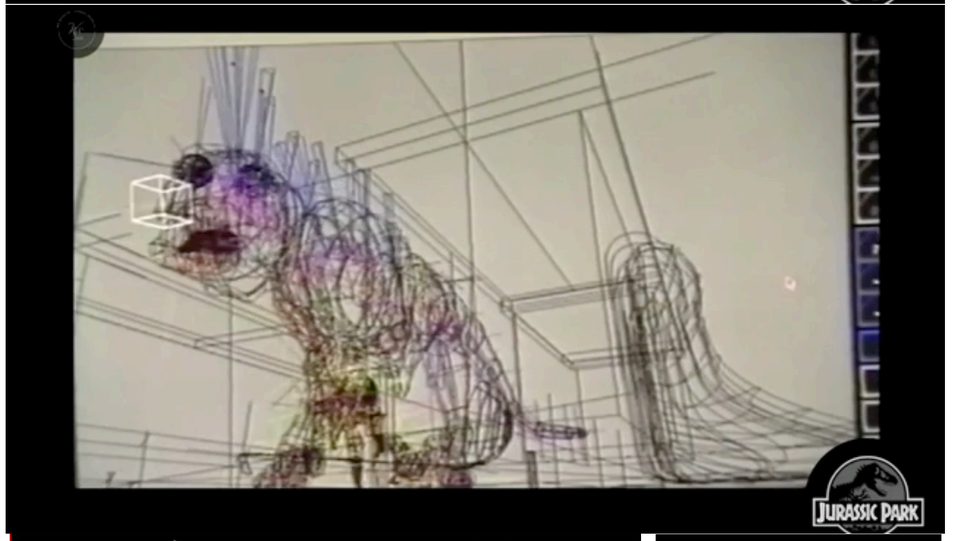
But not differentiable

=> Harder to use for the neural AI, inverse problems

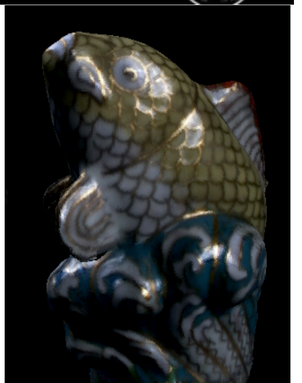
2D -> 3D : starts working but limited

2D -> 2D NVS : meh beyond 3D reconstruction + Lambertian (maybe view-dependent) texture









Quick teaser of modern NVS

Motivation Eye Candy: NERF

NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis

Ben Mildenhall[1⋆] Pratul P. Srinivasan[1⋆] Matthew Tancik[1⋆]
Jonathan T. Barron[2] Ravi Ramamoorthi[3] Ren Ng[1]

[1]UC Berkeley [2]Google Research [3]UC San Diego

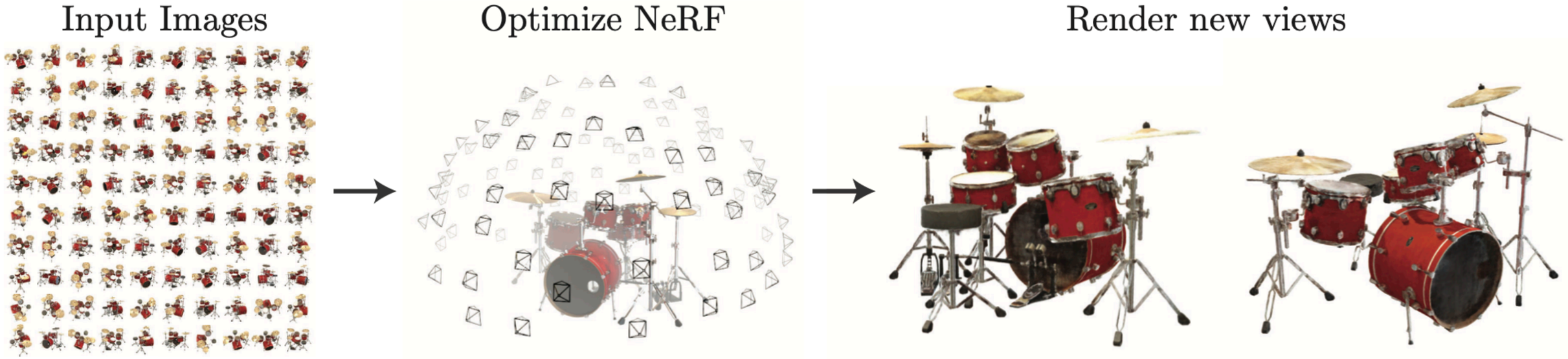
https://www.matthewtancik.com/nerf

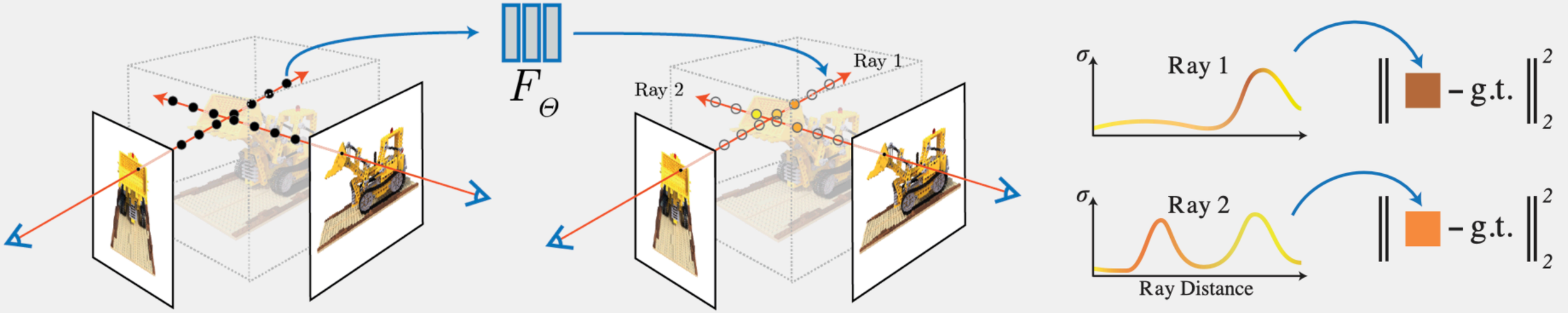
Scenario : 100s of images as input (known cameras), synthesize views for novel 3D viewpoints

Neural network (x, y, z) => RGB+opacity

Rendered with ray marching

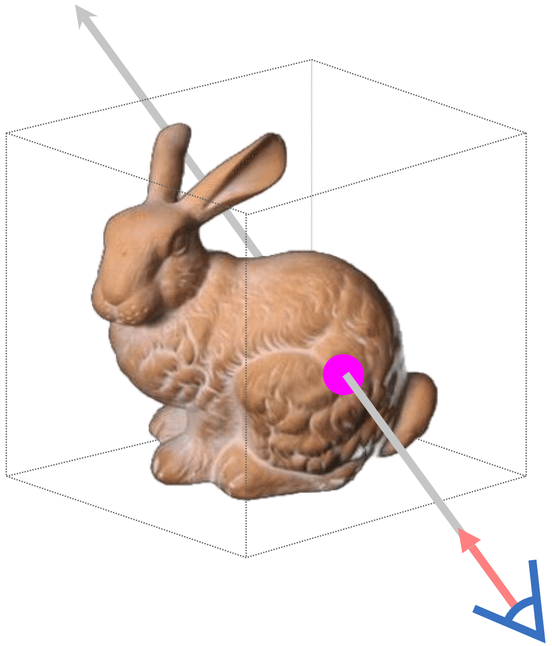
Network overfit to single scene

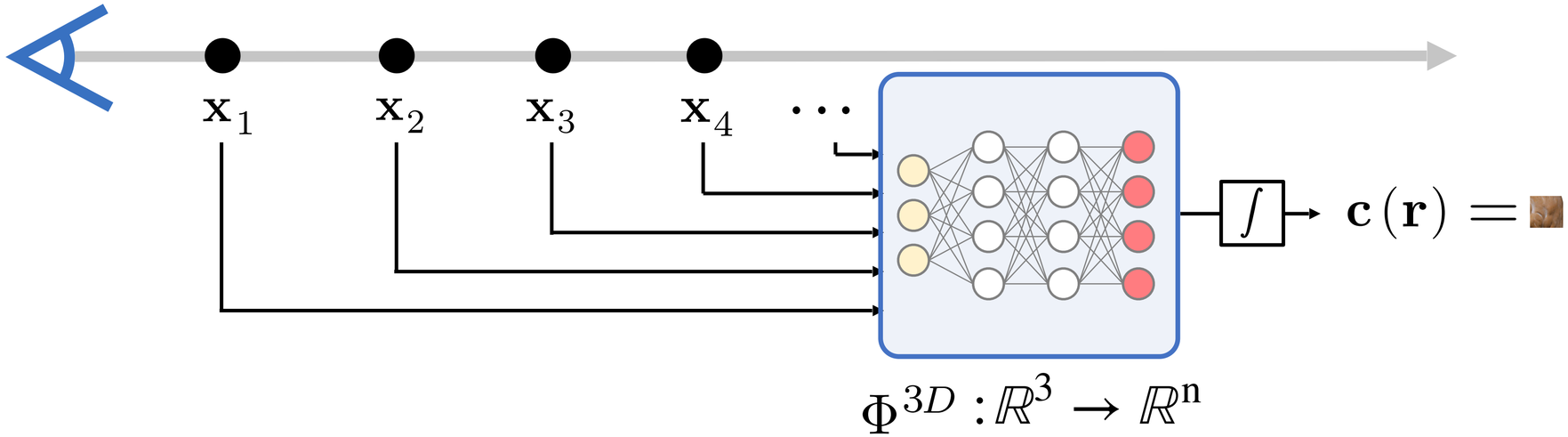




We synthesize views by querying 5D coordinates along camera rays and use classic volume rendering techniques to project the output colors and densities into an image. Because volume rendering is naturally differentiable, the only input required to optimize our representation is a set of images with known camera poses. We describe how to effectively optimize neural radiance fields to render photorealistic novel views of scenes with complicated geometry and appearance, and demonstrate results that outperform prior work on neural rendering and view synthesis.

Basic NERF idea





NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis

Ben Mildenhall* UC Berkeley
Pratul P. Srinivasan* UC Berkeley
Matthew Tancik* UC Berkeley
Jonathan T. Barron Google Research
Ravi Ramamoorthi UC San Diego
Ren Ng UC Berkeley

* Denotes Equal Contribution

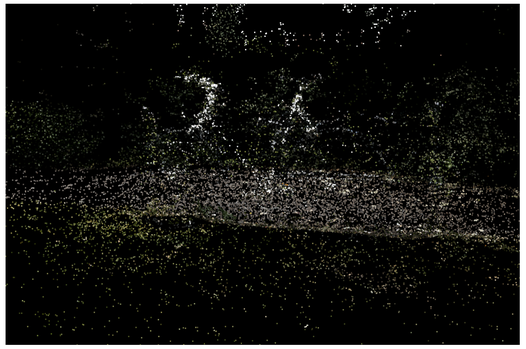






3D Gaussian Splatting

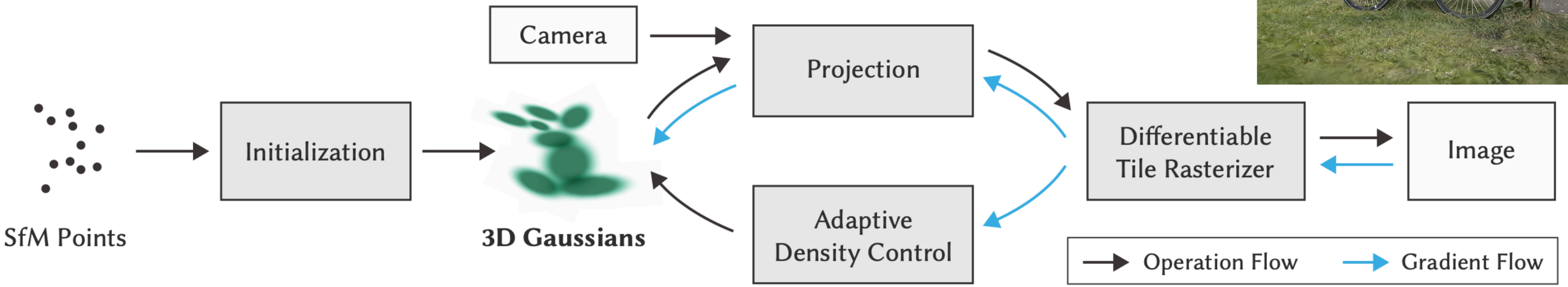
- Intuition: ray-casting through 3D is wasteful (mostly empty space)
- Represent the scene as a collection of points with size: 3D Gaussians (mean & covariance) + color & opacity
- Interestingly, builds on deep learning lessons, but removes deep learning



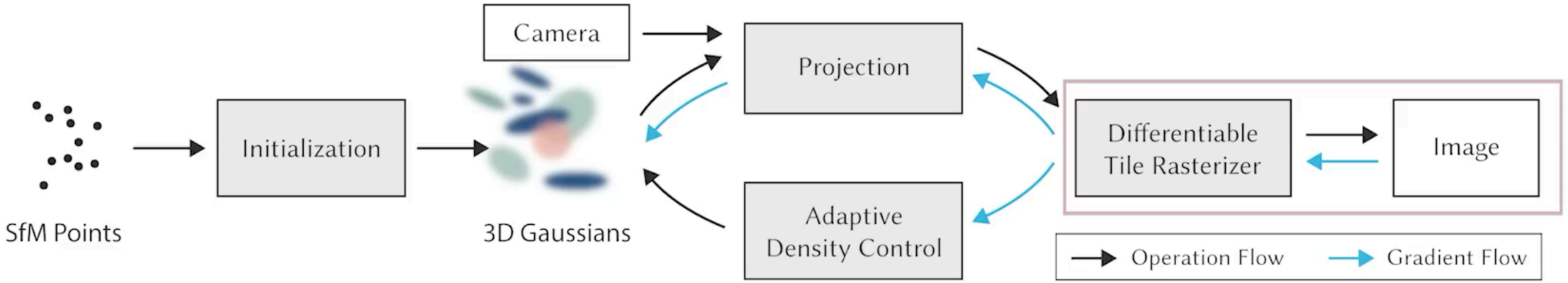




Kerbl, B., Kopanas, G., Leimkühler, T., & Drettakis, G. (2023). 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (ToG)*



Overview of our Method

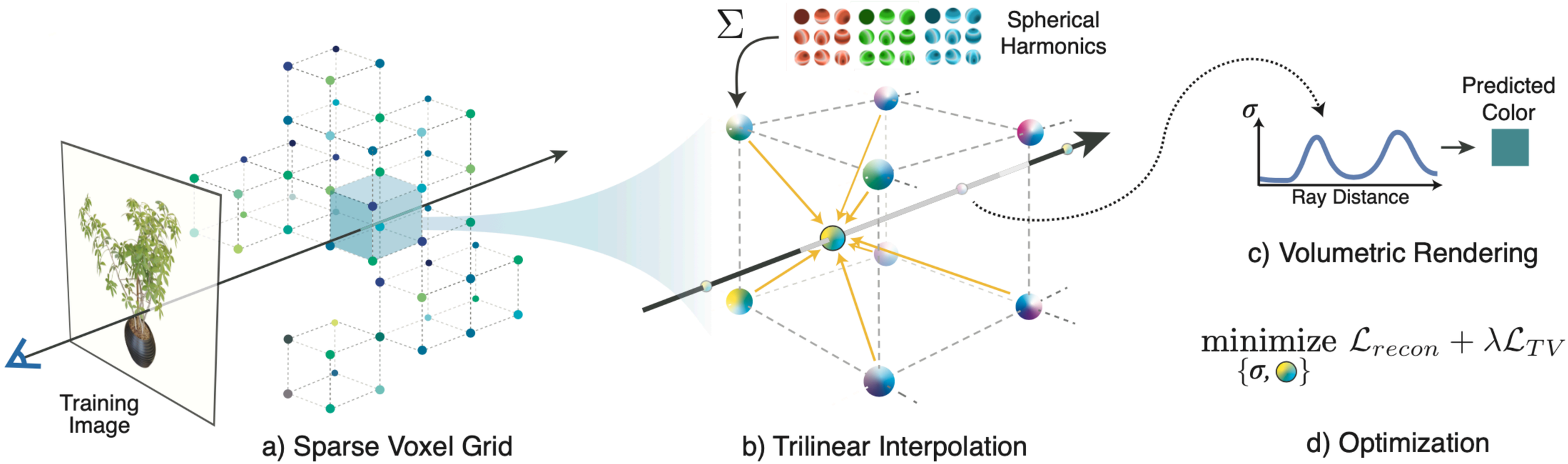


Finally, we render transparent anisotropic Gaussians and backpropagate the gradients to their properties.

Plenoxels: No neural networks

Plenoxels: Radiance Fields without Neural Networks

Alex Yu* Sara Fridovich-Keil* Matthew Tancik Qinhong Chen
Benjamin Recht Angjoo Kanazawa



Neuralangelo: High-Fidelity Neural Surface Reconstruction

Zhaoshuo Li[1,2] Thomas Müller[1] Alex Evans[1] Russell H. Taylor[2] Mathias Unberath[2]
Ming-Yu Liu[1] Chen-Hsuan Lin[1]

DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation

Jeong Joon Park[1,3] Peter Florence [2,3] Julian Straub[3] Richard Newcombe[3] Steven Lovegrove[3]

- Reconstruction as isosurface of neural field (xyz -> distance to surface)



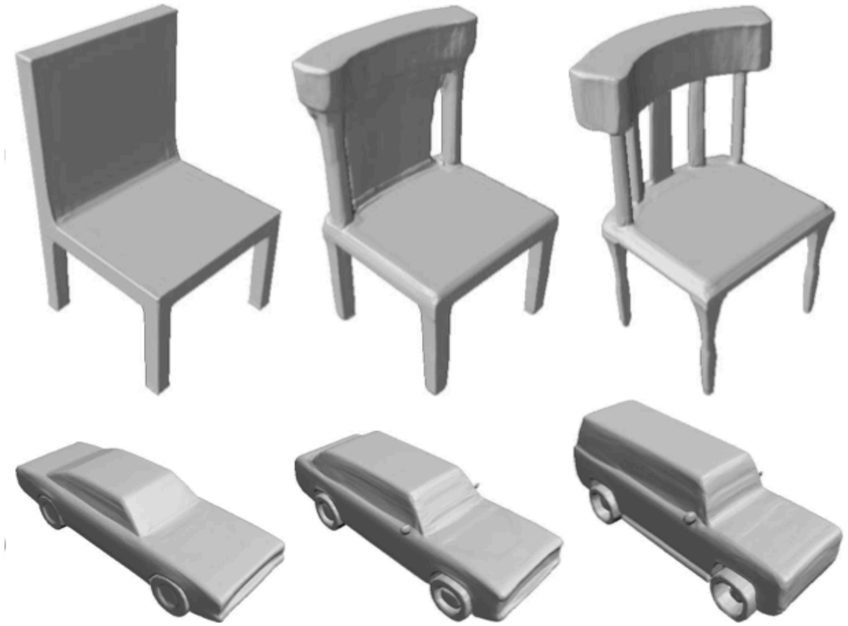


Figure 1. We present **Neuralangelo**, a framework for high-fidelity 3D surface reconstruction from RGB images using neural volume rendering, even without auxiliary data such as segmentation or depth. Shown in the figure is an extracted 3D mesh of a courthouse.

Warning: A complex landscape

- **Various forms of tasks :**
  - Reconstruction (output is 3D)
  - Novel View Synthesis (output is 2D)
- **Various scenarios**
  - Overfit to a single scene (NERF, Gaussian splatting) vs amortized training (DUST3R, VGGT)
  - General vs class specific
  - How much prior is learned (training from a dataset of many scenes, not just the one to be reconstructed)
  - Baked-in illumination or full relightable model
  - Number of input views: from 1 to 100s
- **Various 3D representations** (when used)
  - Volumetric/fuzzy vs hard surface vs 4D line/ray space
  - Various degrees and combinations of neural vs traditional data structure,
  - Various levels of localization/colocation vs global
  - World space or camera space
  - Various levels of coverage, from single view to full 360 all the way to infinity
  - Or skip middleman, no explicit representation

Quick Philosophy

Bitter Lesson

the problem **only needs learning**, no inductive bias/intelligent manually-coded solution (**if you have enough compute & data**)

http://www.incompleteideas.net/IncIdeas/BitterLesson.html

https://www.vincentsitzmann.com/blog/bitter_lesson_of_cv/

Parable of the Parser

the problem is not even relevant (often because bitter lesson allows us to skip it.) It was just an **intermediate fake tasks** for an intelligent manually coded solution

https://drive.google.com/file/d/1VodGljuEhBKwZIXQwN-ApH6g2wBAVAdK/view

Is 3D still needed ? Is it a fake task?