Deep Learning-Based 3D Mapping

ANAIS 2024

January 2024

Prof. Francois Rameau
francois.rameau@sunykorea.ac.kr

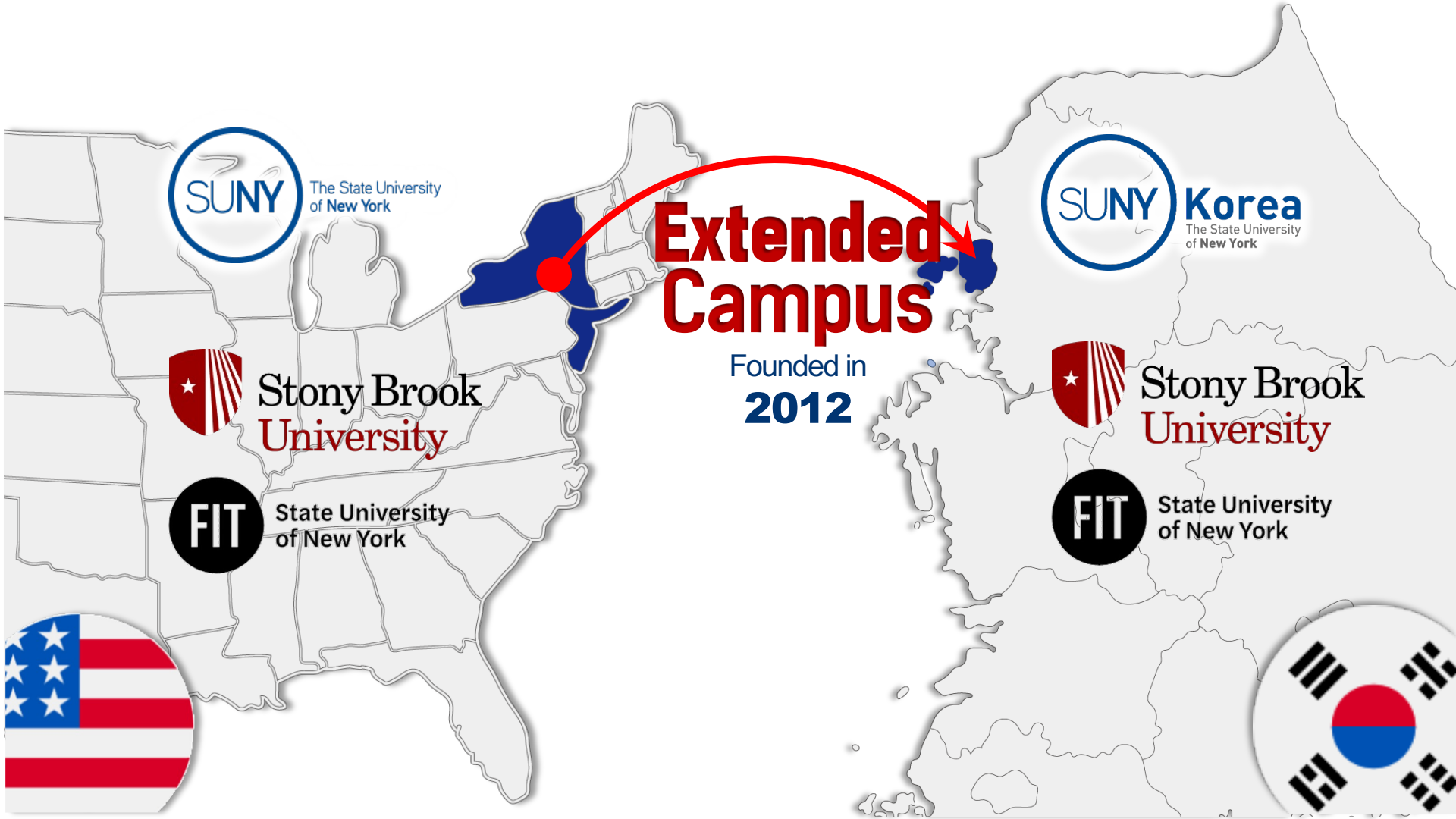
SUNY Korea
The State University of New York

SUNY Korea
The State University of New York

Stony Brook University

FIT State University of New York

SUNY Korea & SBU



3D Reconstruction

What is 3D reconstruction?

Creating a **3D representation** of an **object or scene** from raw data from **sensors**

What Sensors can we use?

Many different kinds of sensors can be used for 3D reconstruction. Can you cite some of them?

Cameras

Pros:
- Compact
- Energy efficient
- Large range
- Affordable

Cons:
- Unknown scale
- Computationally intensive for 3D reconstruction

Camera Rig

Pros:
- 3D at metric scale
- Additional constraints

Cons:
- Calibration
- Synchronization
- More expensive

LiDAR

Pros:
- Direct metric 3D
- Highly accurate

Cons:
- Cost
- Energy consumption
- Bulky
- No color information
- Limited resolution

Depth Sensors

Pros:
- 3D at metric scale
- Color information

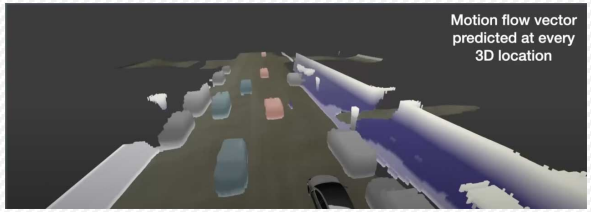
Cons:
- Range
- Ineffective outdoor

Do you know more sensors?

Why do we need 3D reconstruction?

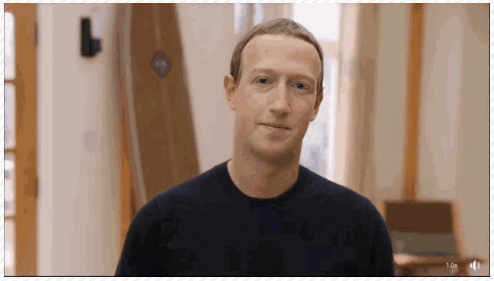
A growing number of fields take advantage of 3D reconstruction

Autonomous Driving

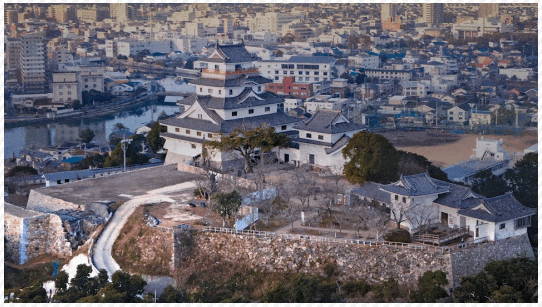


Civil Engineering

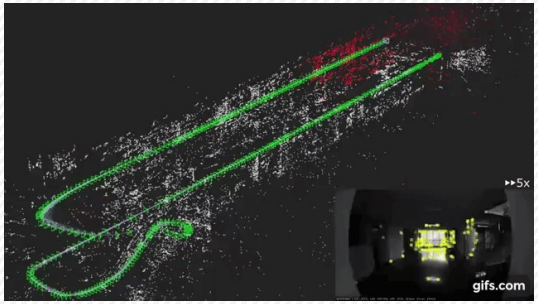
AR/VR



Cultural Heritage



Robotics

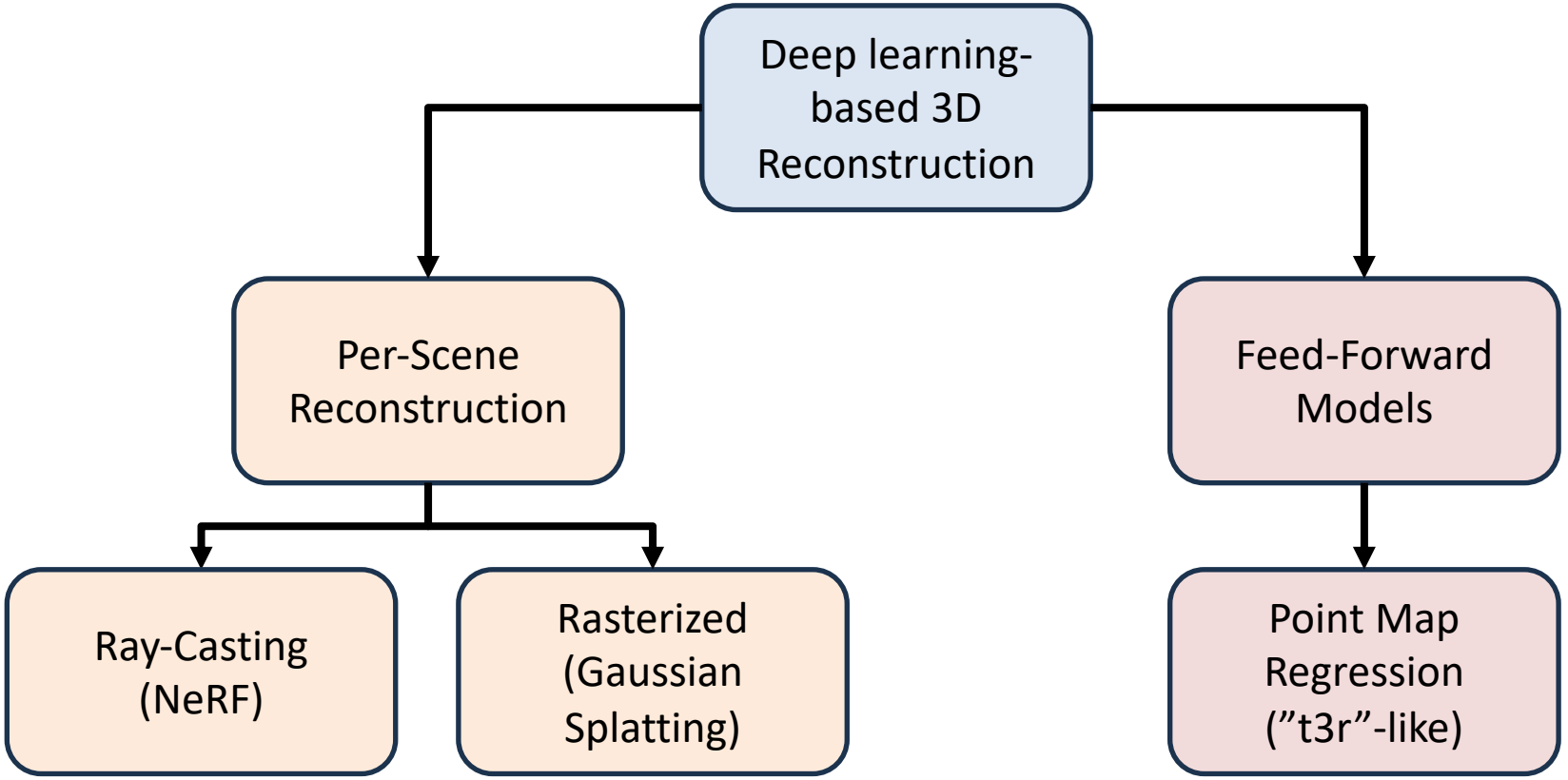


E-Commerce

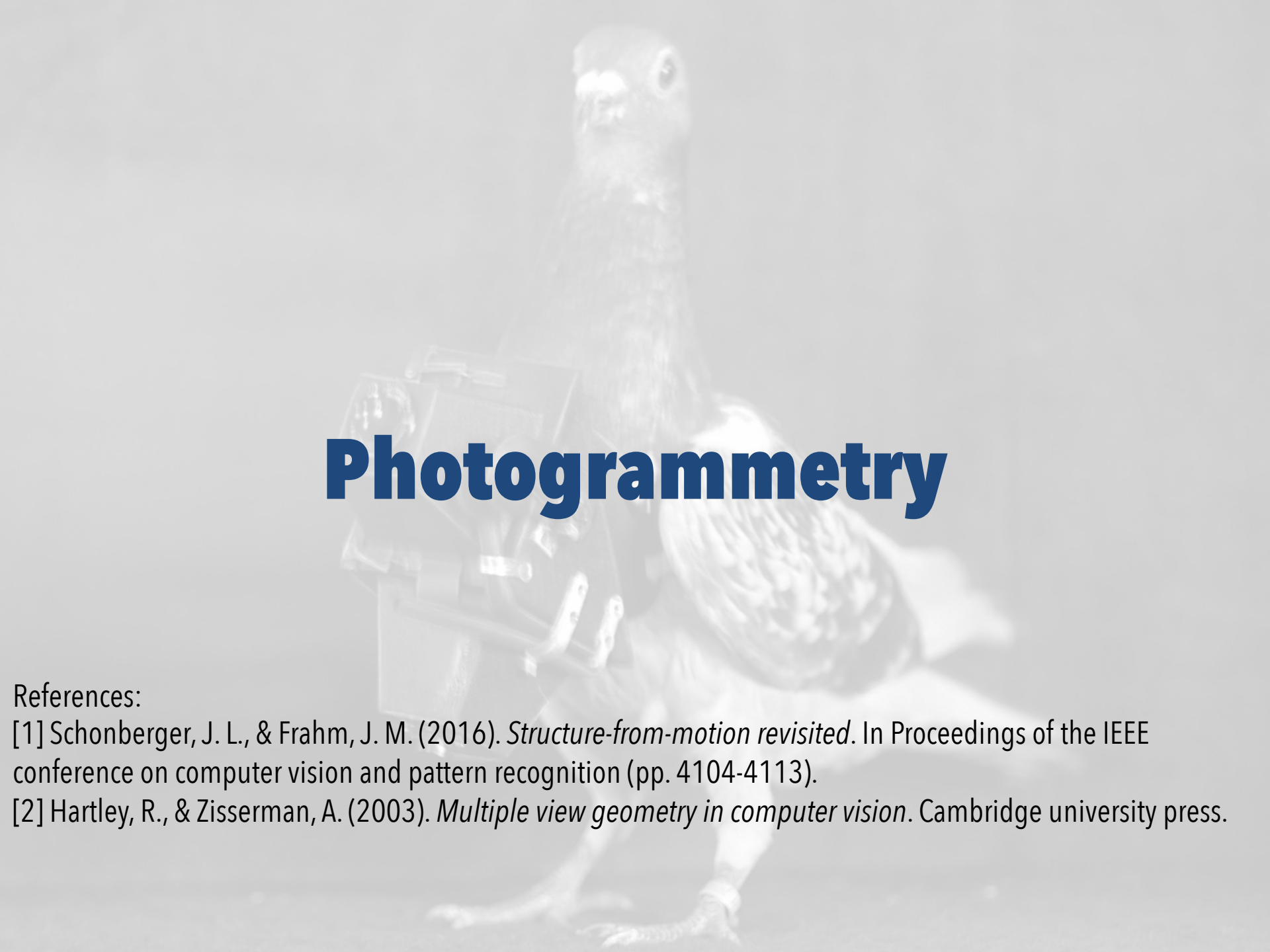
Any ideas where else it can applied?

The Current Trends

For modern 3D reconstruction, two major trends are emerging



But before digging into what is trendy let's see the foundation of it



Photogrammetry

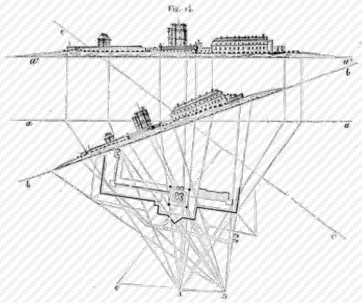
References:
[1] Schonberger, J. L., & Frahm, J. M. (2016). *Structure-from-motion revisited*. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4104-4113).
[2] Hartley, R., & Zisserman, A. (2003). *Multiple view geometry in computer vision*. Cambridge university press.

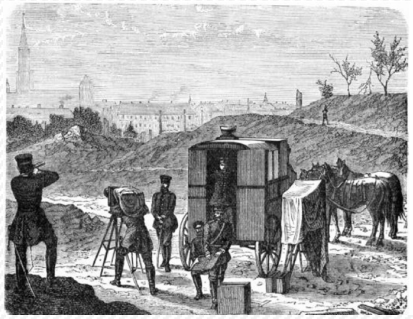
Where it all started?

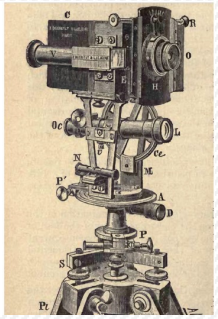
Photogrammetry is the science of using overlapping photographs to measure and reconstruct accurate 3D models of objects, scenes, or terrains.

Aimé Laussedat

1849: He experimented with using photographs to create topographic maps and measure the dimensions of structures.



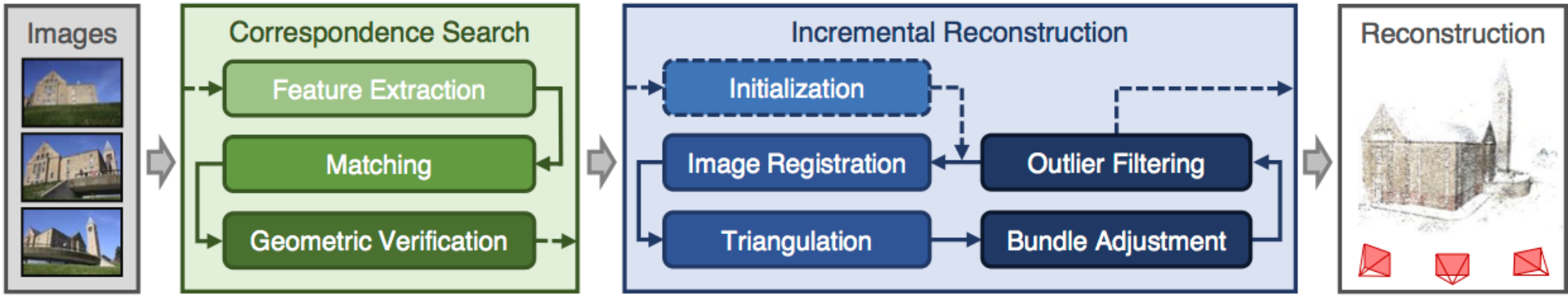




We have not invented much since then . . . we just made it automatic!

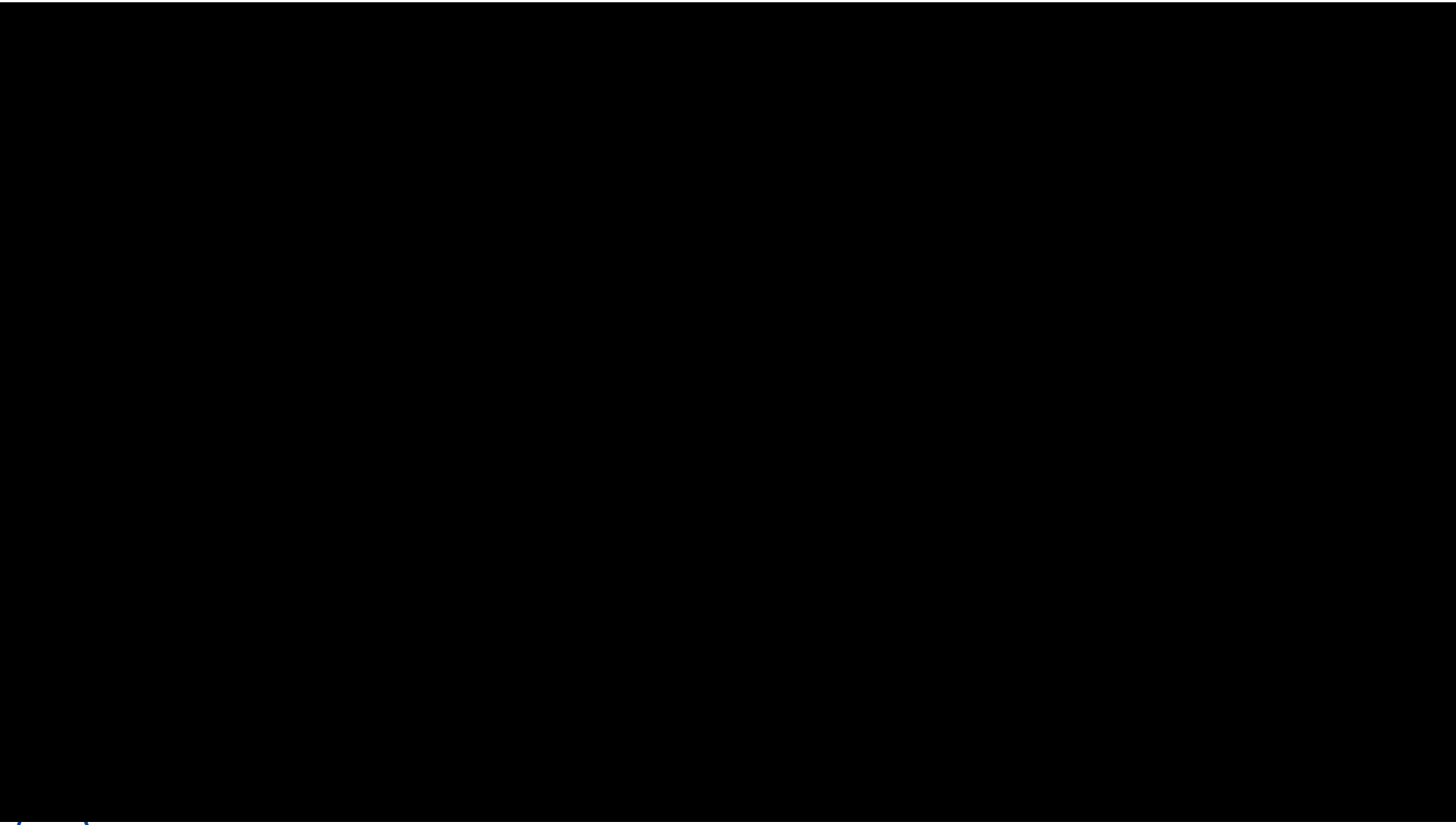
Structure from Motion Pipeline

How did we automatize it? Well, by matching keypoints across images



Colmap pipeline

Modern Results



Everything Starts with Images

Collect your data the way you like!

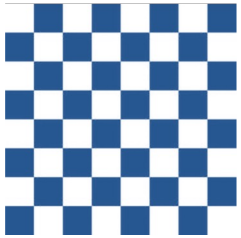
UAV

handheld

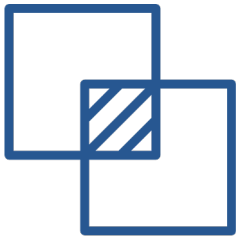
PhD Student

But be careful!

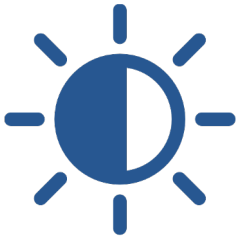
Avoid motion blur



Calibrate your camera



Ensure 60 to 80%
Overlap btw images



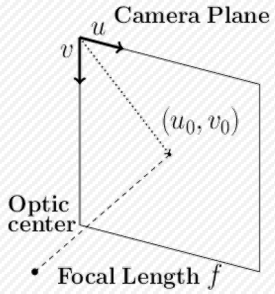
Maintain consistent
lighting

Understand the Camera Geometry

To reconstruct the scene, you first need to model the projection

Camera's internal geometry

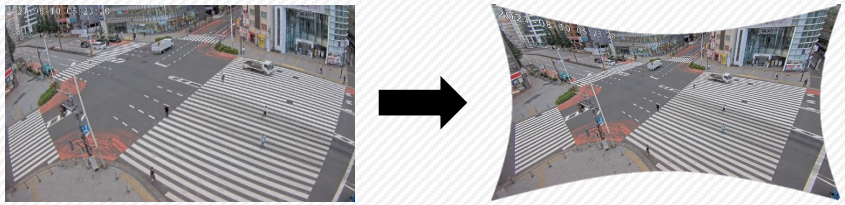
Intrinsic parameters



$$\tilde{\mathbf{p}} \sim \mathbf{KP},$$

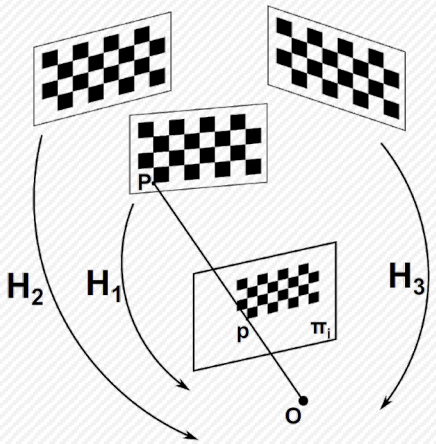
$$\tilde{\mathbf{p}} \sim \begin{bmatrix} f & skew & u_0 \\ 0 & \lambda f & v_0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{P}$$

Distortion



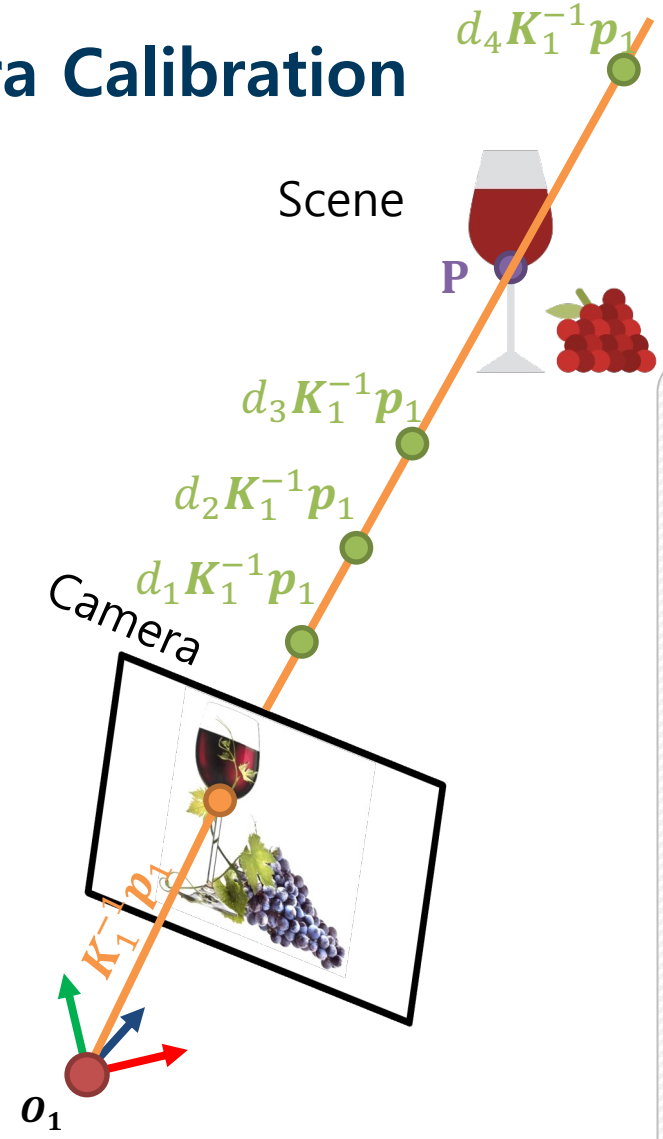
Camera calibration

Often, the camera is calibrated offline using a checkboard pattern



The output is this process are the intrinsic parameters

Camera Calibration



Why is camera calibration needed

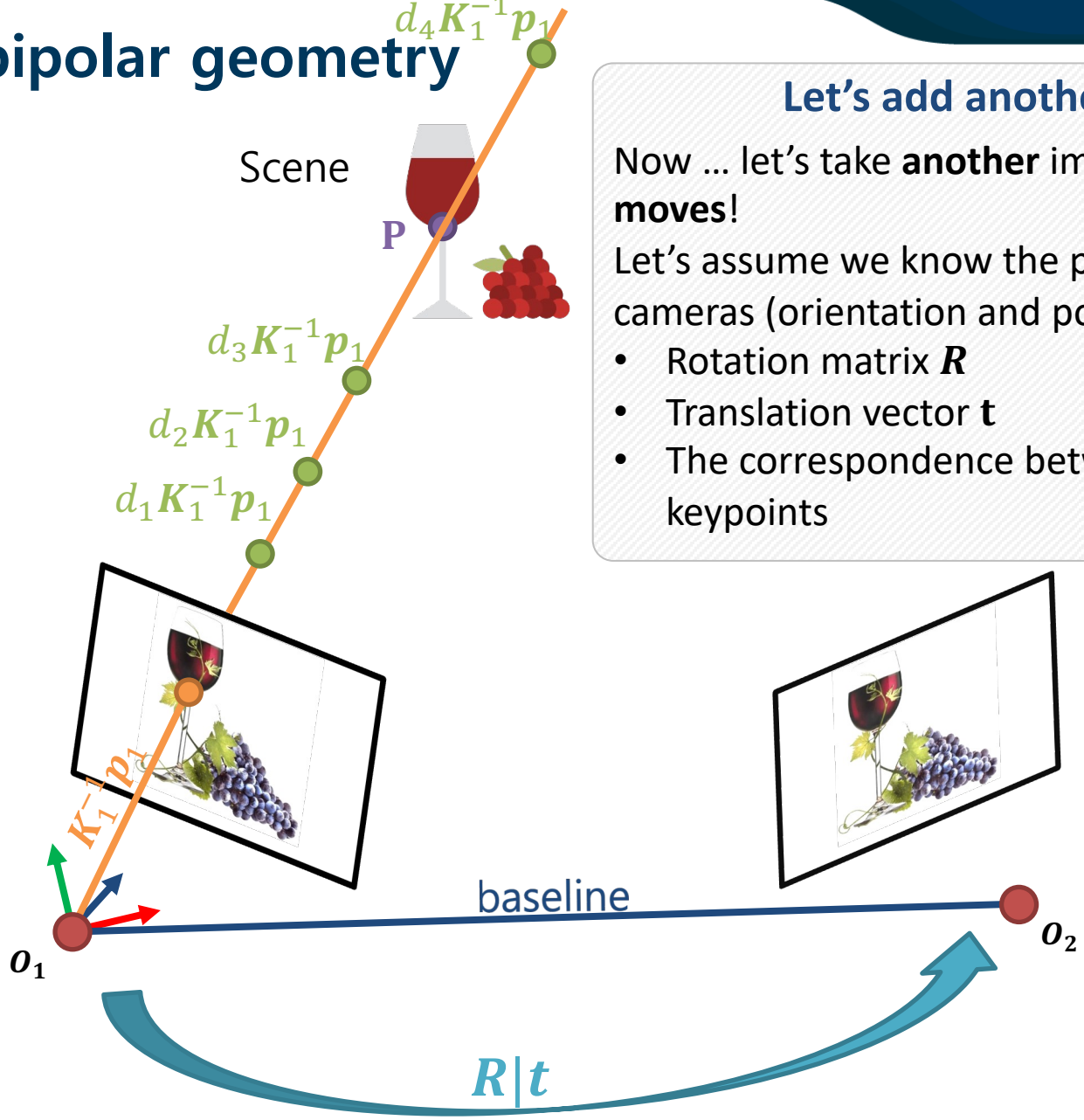
From a single view cannot reconstruct the scene. But … with the calibration, we know the light ray vector forming each point in the image.

For a image point $\boldsymbol{p}_1 = \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$ its vector is $\boldsymbol{K}_1^{-1}\boldsymbol{p}_1 \sim \mathbf{P}$ therefore $d\boldsymbol{K}_1^{-1}\boldsymbol{p}_1 = \mathbf{P}$

Only one unknown scalar d, the depth

But what is the depth? From a single view, we do not know

The epipolar geometry



Let's add another view

Now … let's take **another** image. Our camera **moves**!

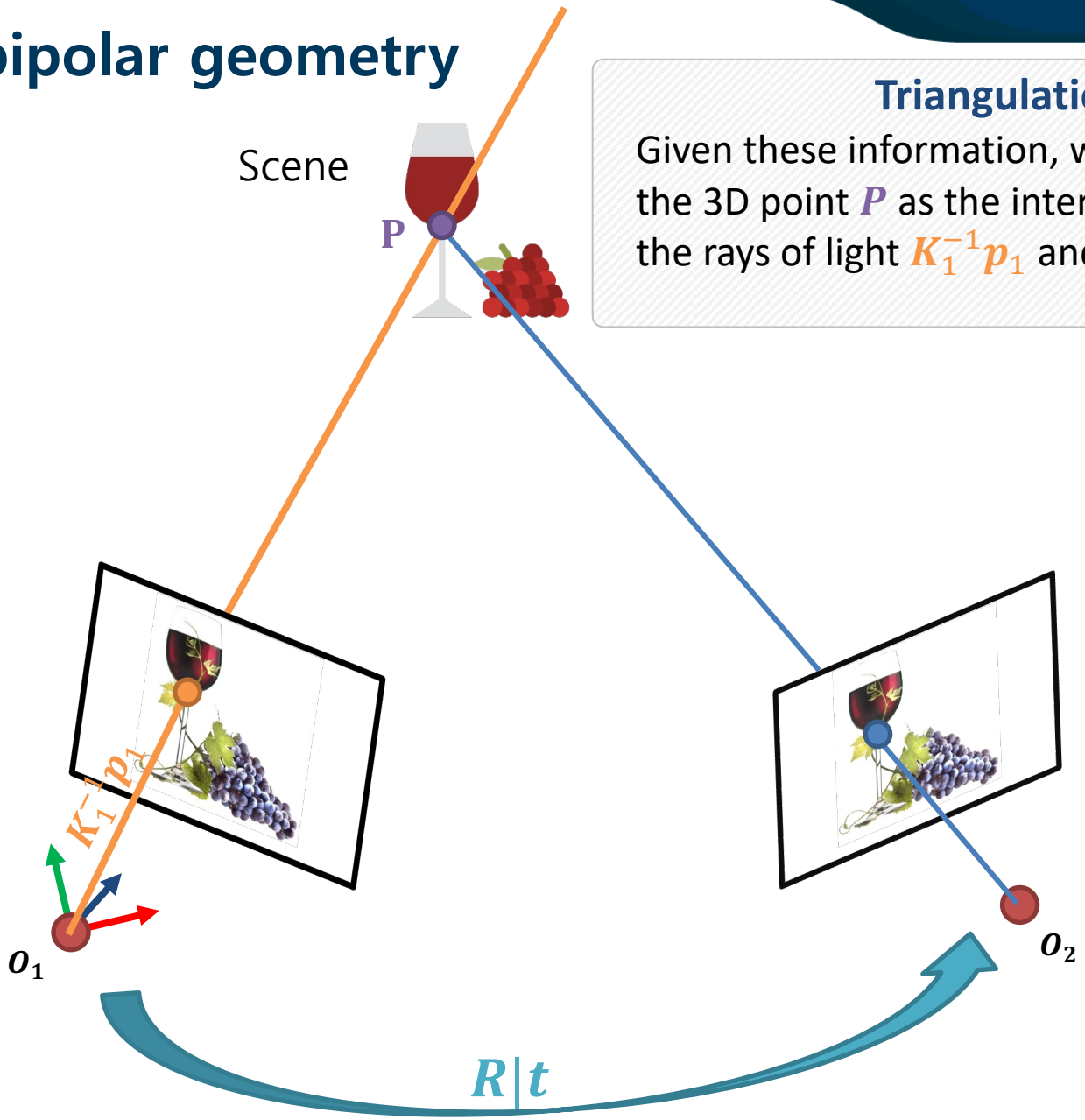
Let's assume we know the pose between the cameras (orientation and position):

- Rotation matrix $\boldsymbol{R}$
- Translation vector $\mathbf{t}$
- The correspondence between both images keypoints

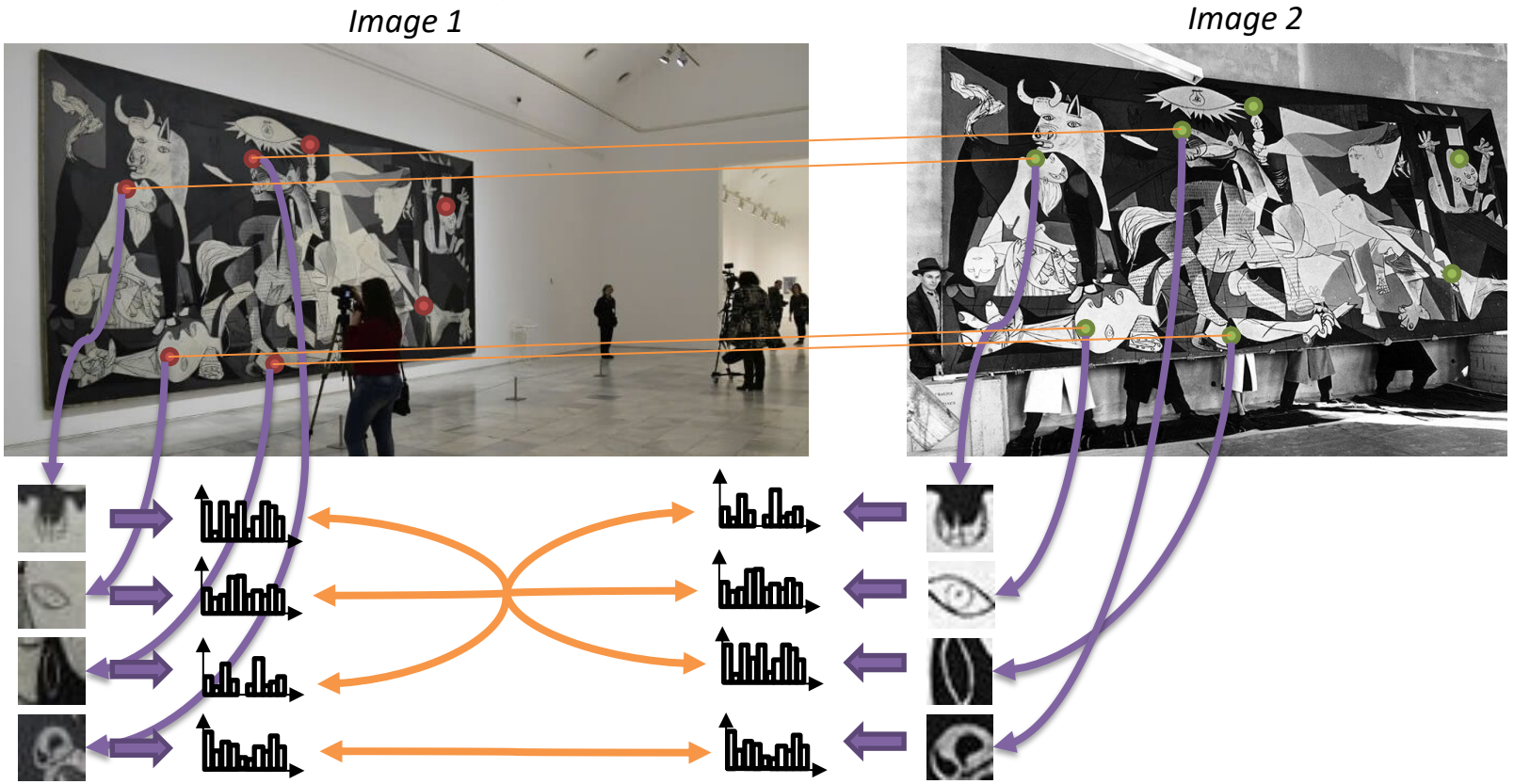
The epipolar geometry

Triangulation

Given these information, we can triangulate the 3D point P as the intersection between the rays of light $K_1^{-1}p_1$ and $RK_W^{-1}p_w + t$



Features Matching

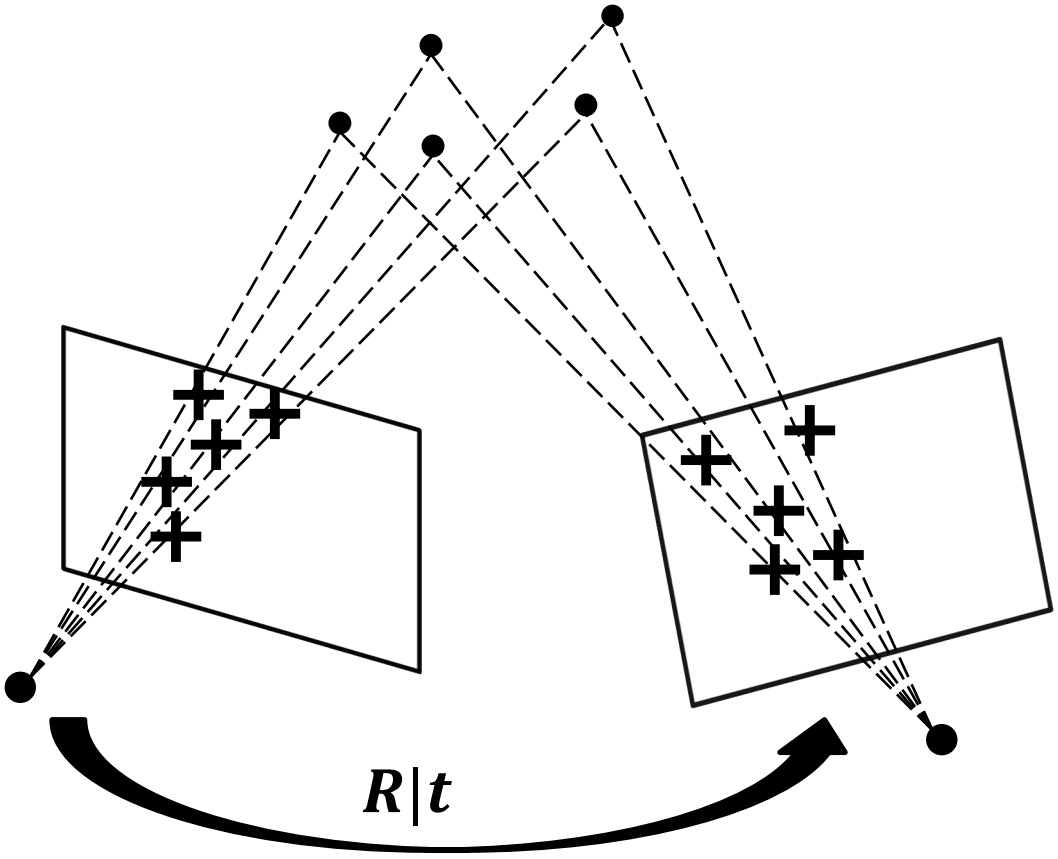


Given two images how to find a set of corresponding points?

1. detection of distinctive keypoints in the image 1
2. detection of distinctive keypoints in the image 2
3. Extraction of meaningful descriptor for each patch around the keypoints
4. Find the best pair of correspondence (keypoints with closest descriptors)

Features Matching

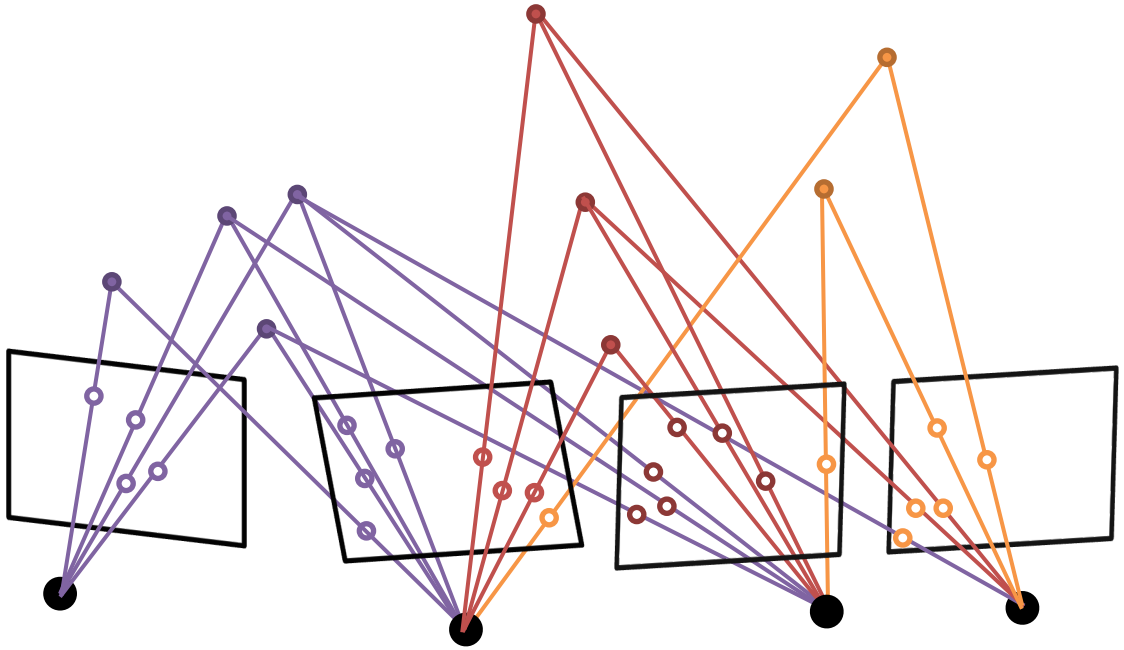
Given at least five correct correspondences, you can estimate the pose between two images via the Essential Matrix and initialize a 3D point cloud!



View Aggregation

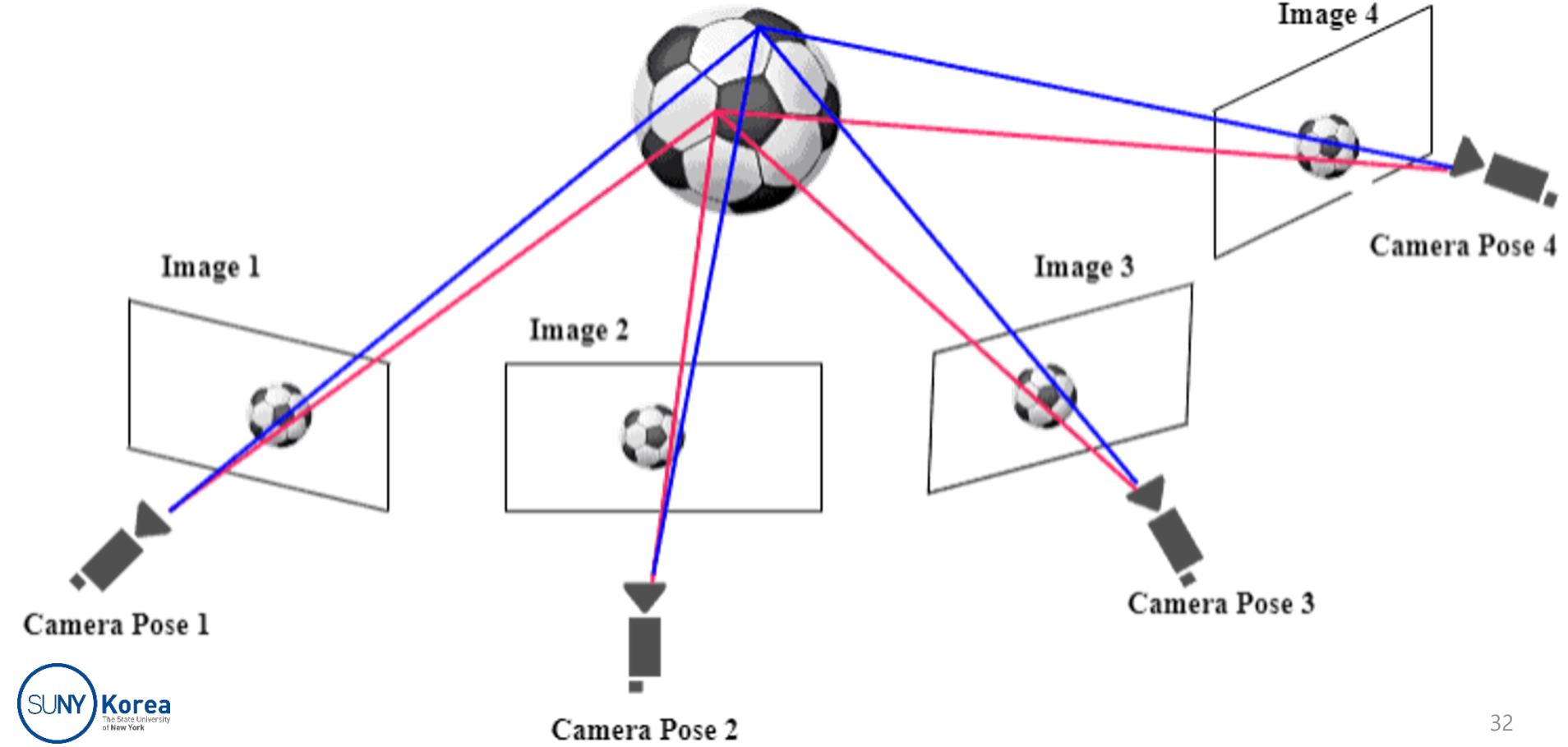
Now that you know the basics, you can initialize a 3D map and aggregate new views to it!

1. Triangulate 3D points
2. Find the pose of another image using matches
3. Triangulate new points
4. Repeat

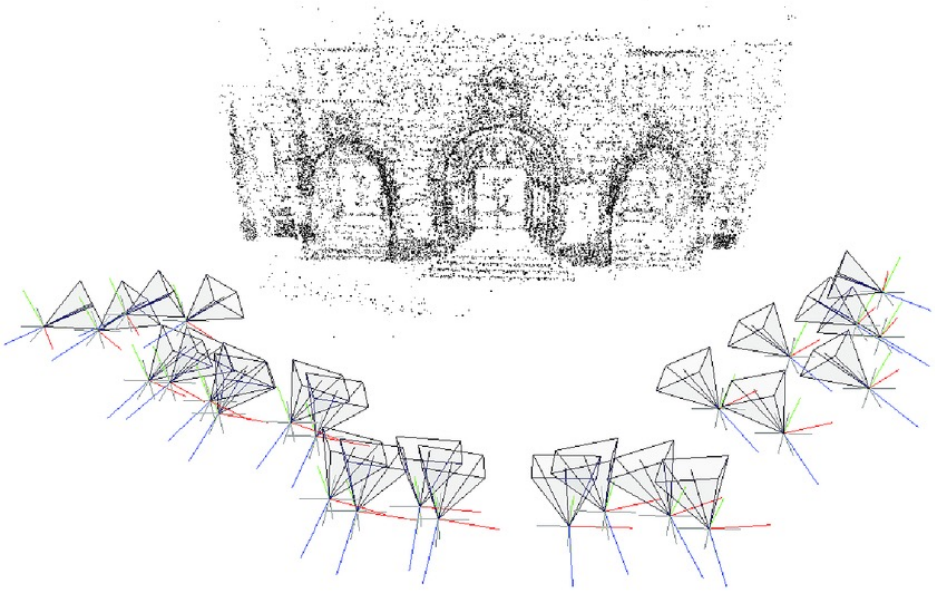


Bundle Adjustment

Bundle adjustment optimizes camera parameters and 3D points to enforce geometric consistency across all views by minimizing reprojection error.



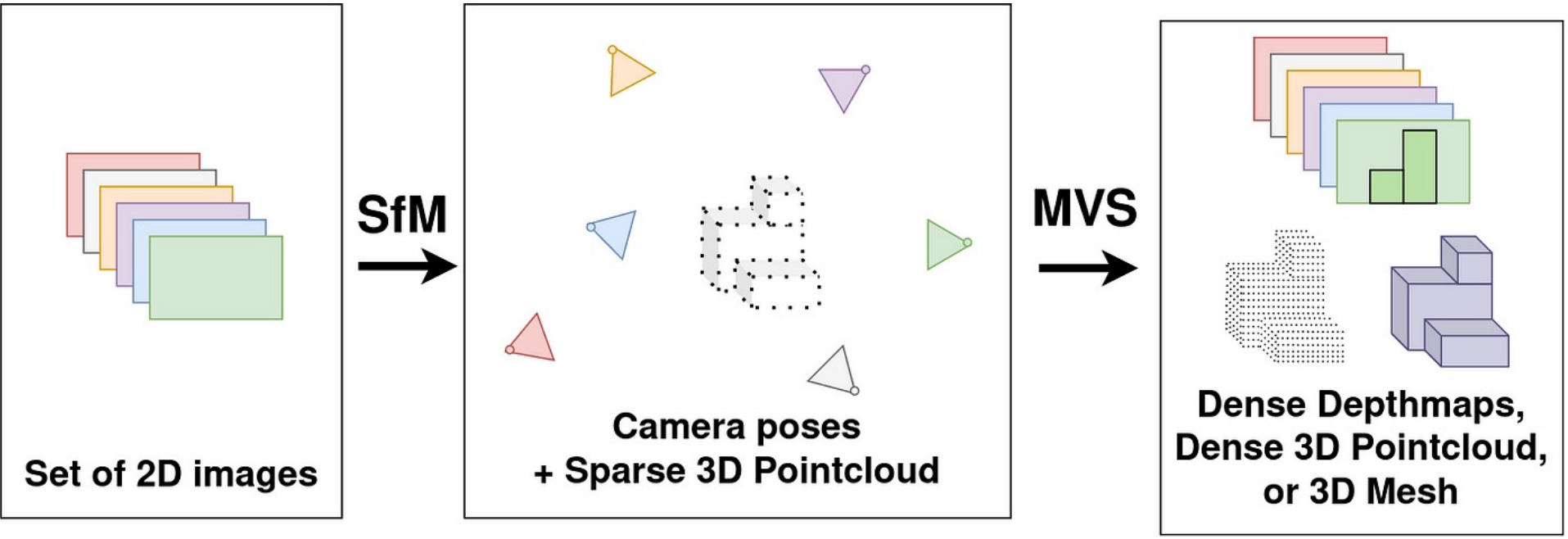
The Output of SfM



What can you say about this output? Is it useful?

This dense reconstruction is better, how can we get this?

Densification via MVS



SfM: Structure from Motion
MVS: Multi-View Stereo

A Look on the Final Reconstruction

What can we say about this result?

Deep Learning Per-Scene Fitting

Neural Radiance Fields

References:
[1] Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., & Ng, R. (2021). *Nerf: Representing scenes as neural radiance fields for view synthesis*. Communications of the ACM, 65(1), 99-106.

Note: Some of the slides used in this presentation are inspired by the talk of Torsten Sattler entitled "A Brief Introduction to Neural Radiance Fields" at CESCG Academy 2023.

NeRF

NeRF stands for Neural Radiance Fields

Representing Scenes as Neural Radiance Fields for View Synthesis

ECCV'20

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

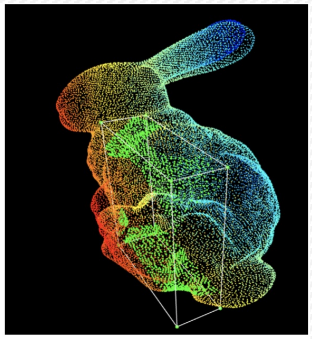
*Denotes Equal Contribution

SUNY Korea
The State University of New York

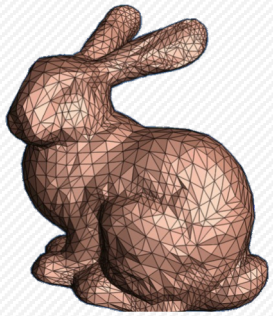
Explicit Representations

The previous reconstructions we have seen where achieved using explicit representations

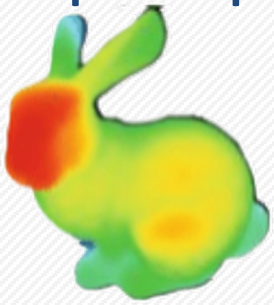
Point Cloud



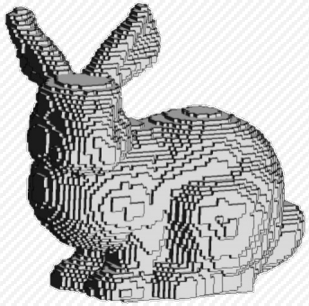
Mesh



Depth Map



Voxels

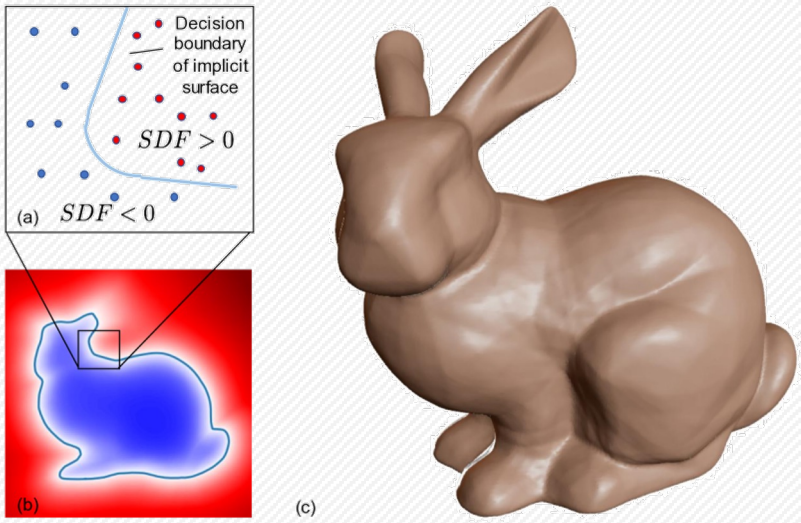


Explicit representation directly defines the geometry or structure of a scene using concrete elements like point clouds, meshes, or voxels.

Implicit Representation

In 3D geometry, we also commonly employ implicit representations

A Good Example: The Signed Distance Function



An SDF encodes the signed distance of a point to the nearest surface.

Why?

Implicit representations often have very desirable properties:

- **Continuous**
- **Differentiable**

But SDF, as well as explicit representations, assume Lambertian Surfaces!

NeRF - Motivation

One way to perform implicit representation is to use an NN to store the geometry of a scene

Better Capture Fine Details

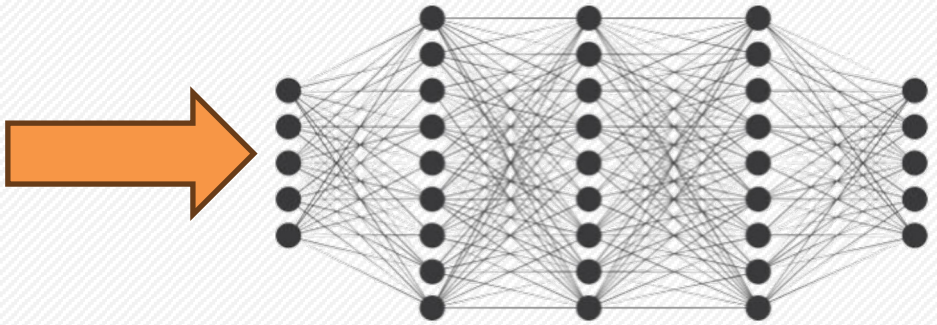
Reflection and Illumination Variations

And it is VERY compact!

Implicit Representation - Image

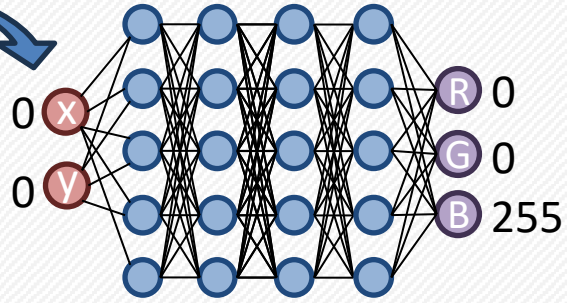
For instance, can we have an implicit representation of an image?

What if you Wanted to Store this Image into a Neural Network?



What about something like that?

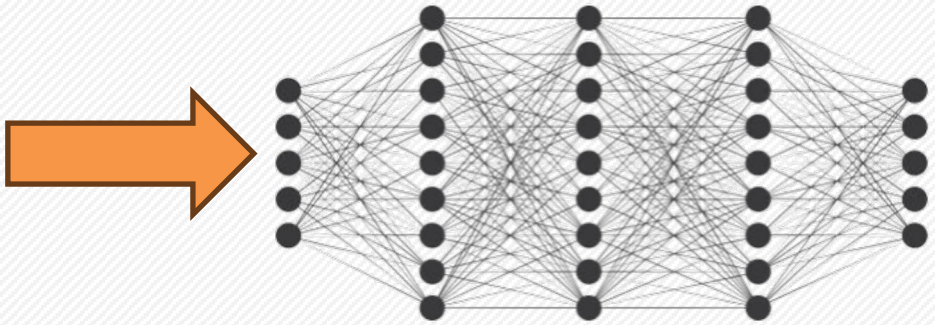
0,0	0,1	0,2	0,3	0,4	0,5
1,0	1,1	1,2	1,3	1,4	1,5
2,0	2,1	2,2	2,3	2,4	2,5
3,0	3,1	3,2	3,3	3,4	3,5



Implicit Representation - Image

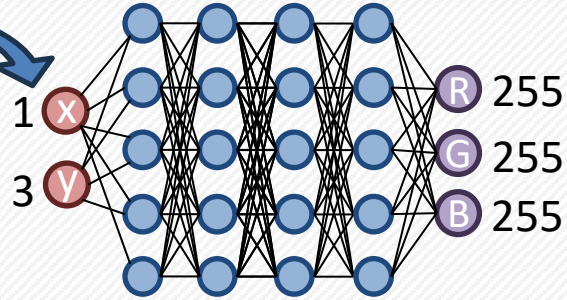
For instance, can we have an implicit representation of an image?

What if you Wanted to Store this Image into a Neural Network?



What about something like that?

0,0	0,1	0,2	0,3	0,4	0,5
1,0	1,1	1,2	1,3	1,4	1,5
2,0	2,1	2,2	2,3	2,4	2,5
3,0	3,1	3,2	3,3	3,4	3,5



Do you think it will work?

Implicit Representation - Image

Let's try with a real image

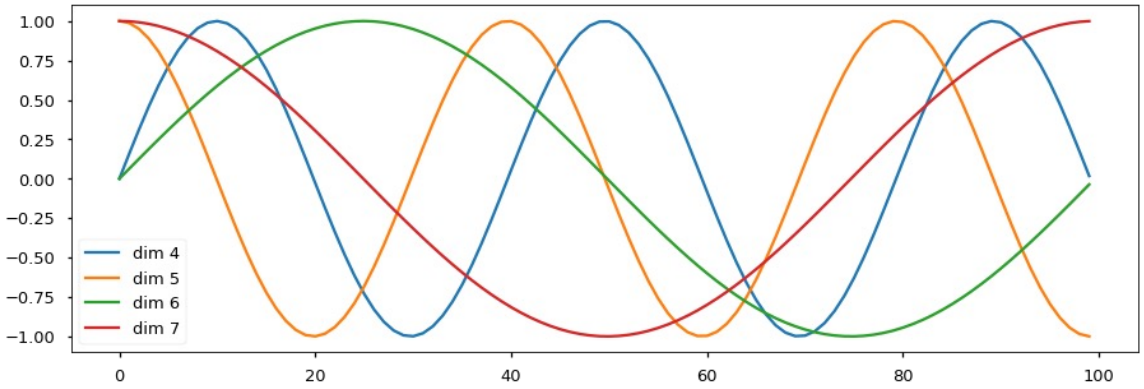
Source: [Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains]

MLP output

Supervision image

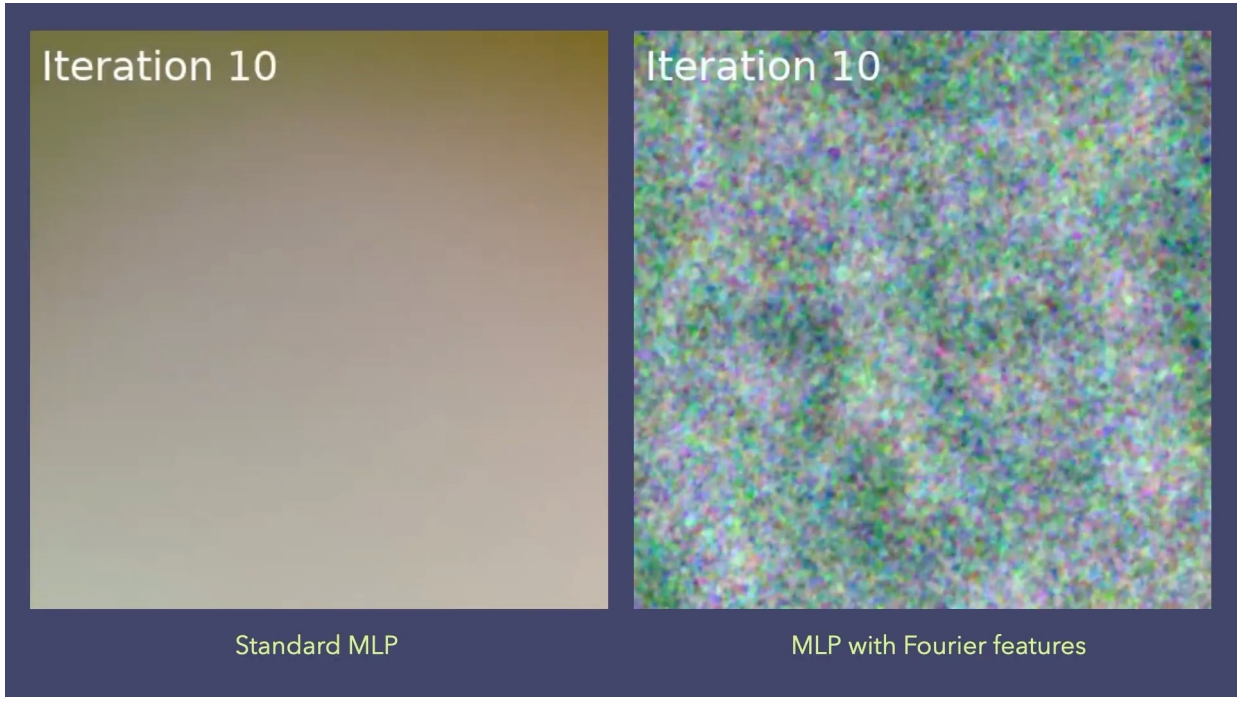
Implicit Representation - Image

- **Why doesn't it work?**
  - An MLP inherently favors smooth, low-frequency patterns missing high-frequency details
- **Solution:**
  - **positional encoding**, which maps input coordinates to a higher-dimensional space using sinusoidal transformations



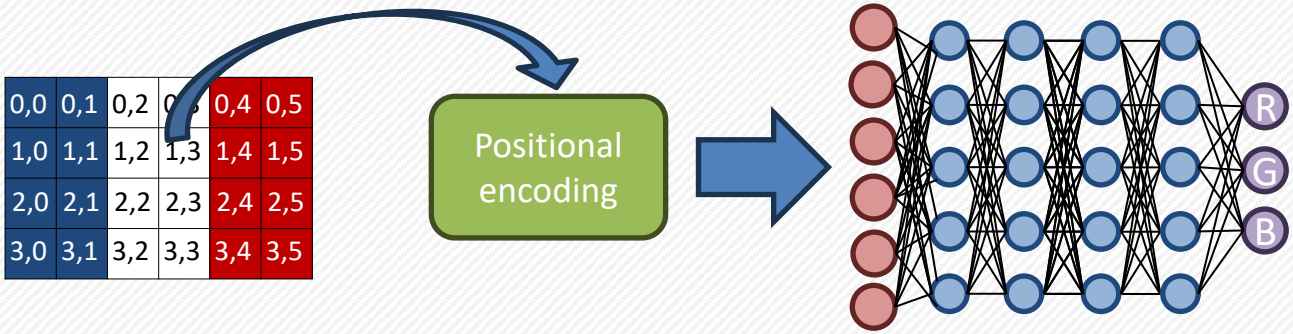
$$\gamma(p) = \left(\sin\left(2^0\pi p\right), \cos\left(2^0\pi p\right), \cdots, \sin\left(2^{L-1}\pi p\right), \cos\left(2^{L-1}\pi p\right)\right).$$

Implicit Representation - Image



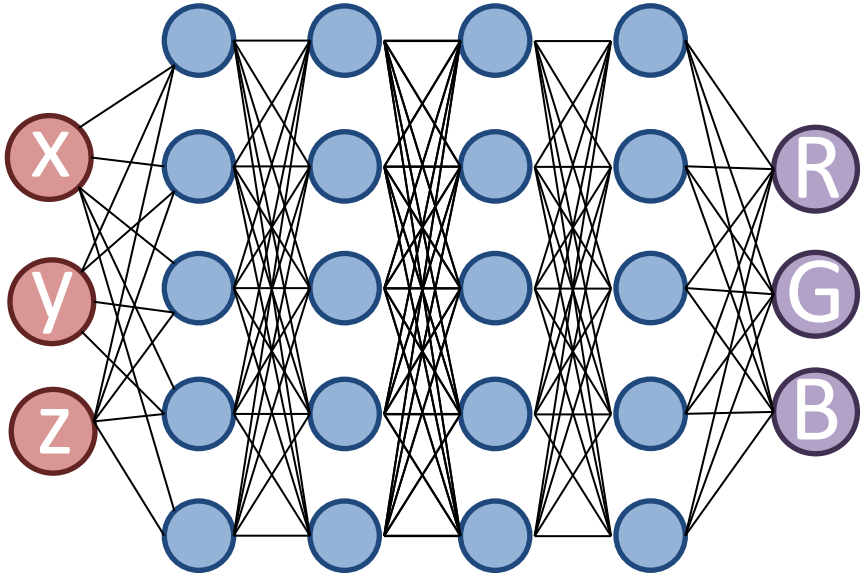
Source: Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains

With Positional encoding



Implicit Representation – 3D

Now that we know that it is possible with a 2D image, can we do the same with an entire 3D Scene?



By intuition, you might believe it can look like this?!

How would it be different?

- Additional inputs/outputs
- Need multiple images of the scene
- Viewpoint Dependence
- Everything is “empty” except for the surfaces
- Transparent or reflective surfaces

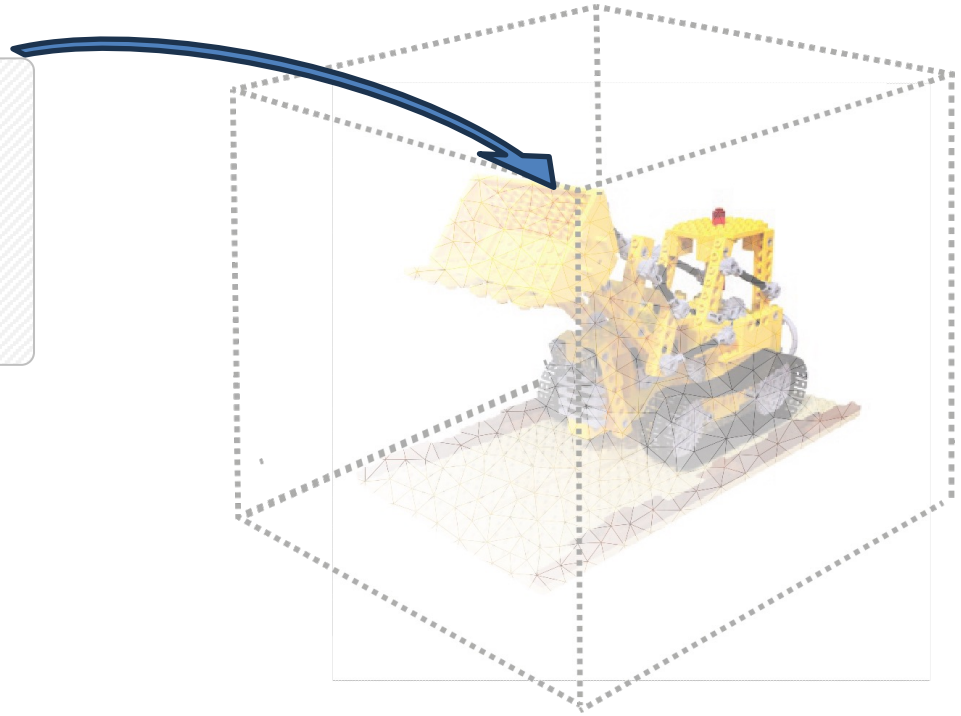
This is exactly what NeRF is addressing!

Basics – Volume Rendering

In a Sample

Each sample in the volume contains:

- The color $\boldsymbol{c}_i$
- The volume density σ_i

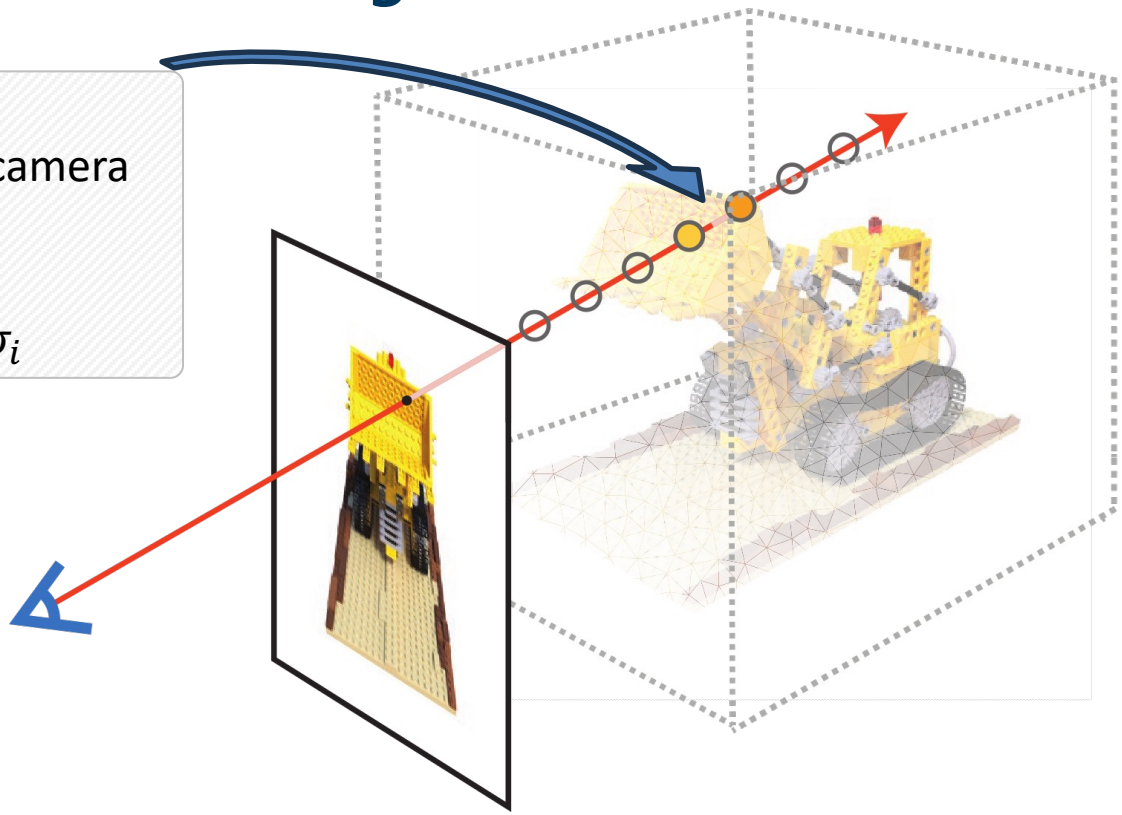


Basics – Volume Rendering

In a Sample

Each ith sample along a camera ray contains:

- The color $\boldsymbol{c}_i$
- The volume density σ_i

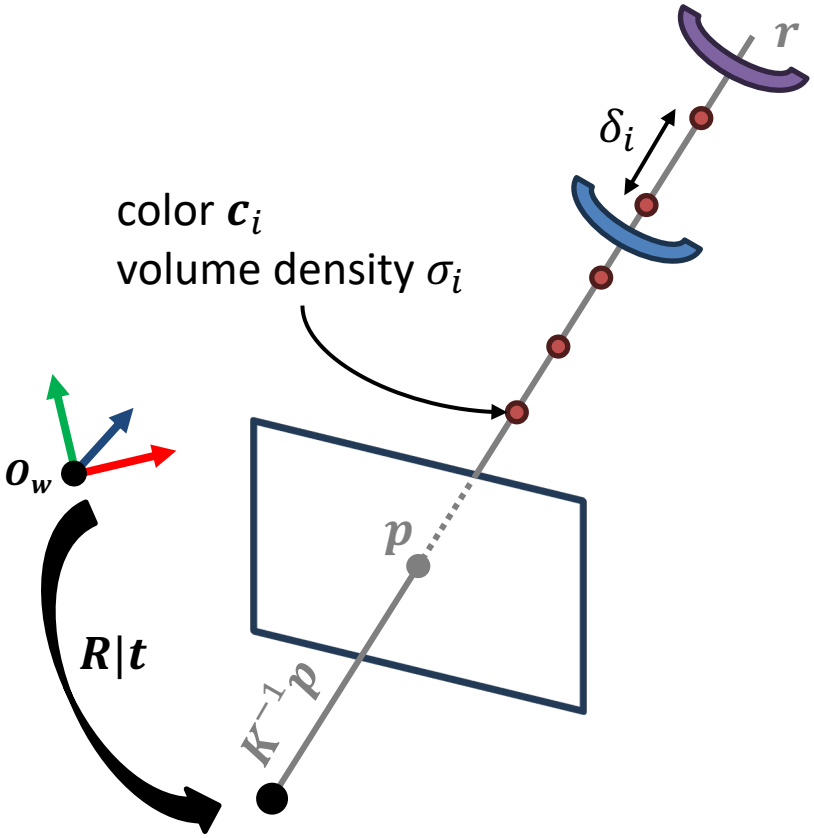


The color $\hat{C}(r)$ of a pixel, is the integration of all sampled points along the ray r

$$\hat{C}(r) = \sum_{i=1}^{N} \alpha_i \boldsymbol{c}_i$$

Torsten Sattler, "A Brief Introduction to Neural Radiance Fields", CESCG Academy 2023.

Basics – Volume Rendering



Detailed Equation

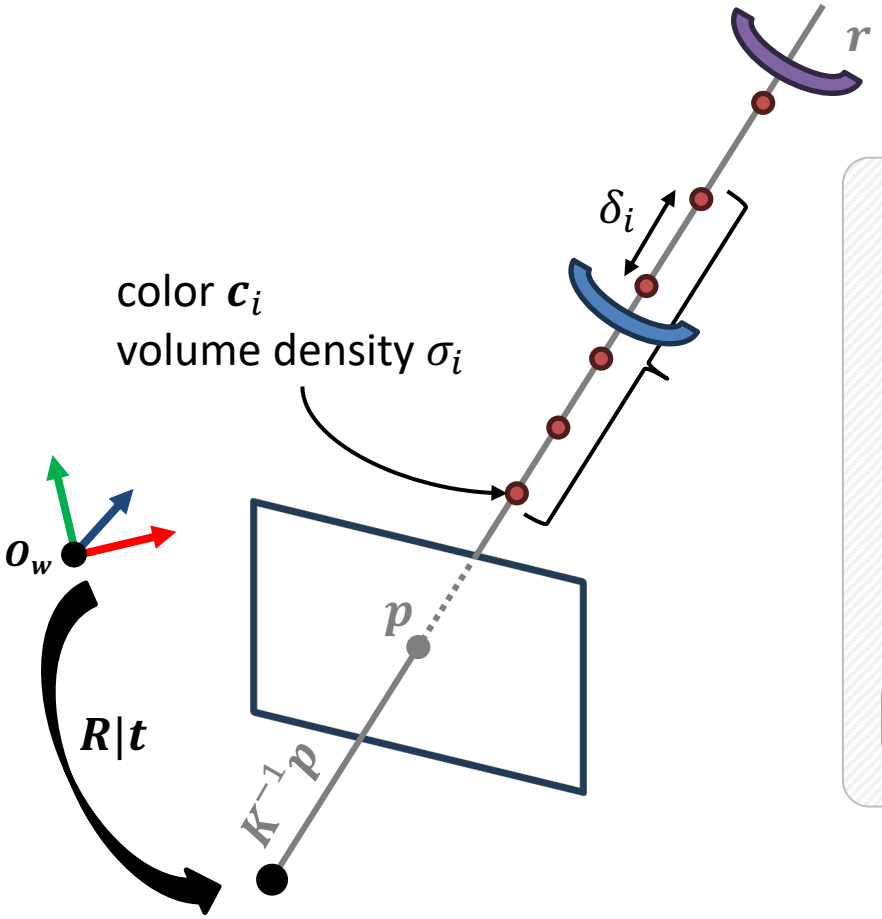
$$\hat{C}(r) = \sum_{i=1}^{N} \alpha_i \boldsymbol{c}_i$$

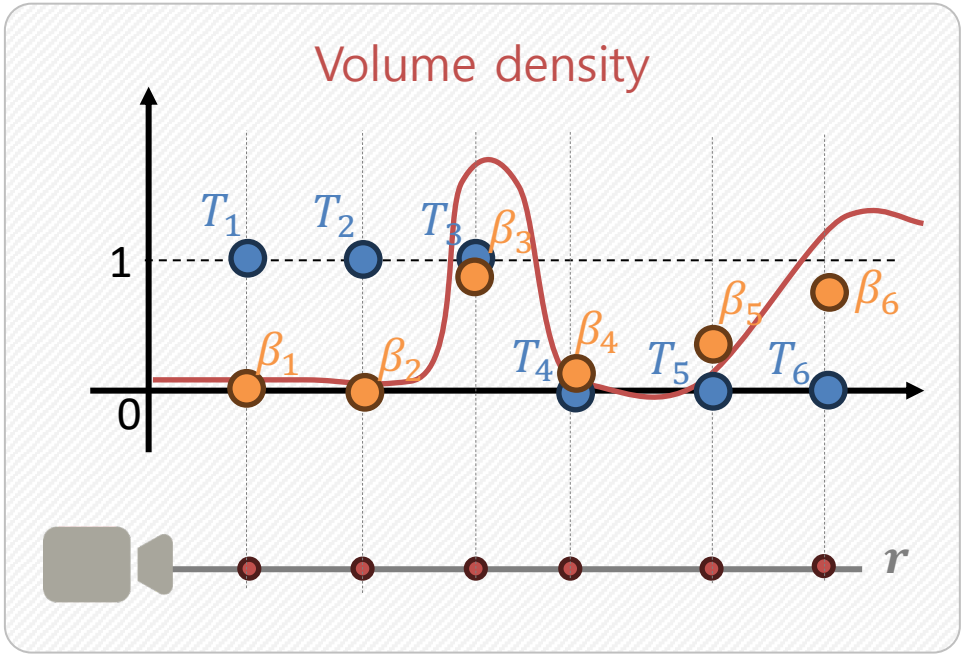
$$= \sum_{i=1}^{N} T_i \underbrace{(1 - e^{(-\sigma_i \delta_i)})}_{\text{visibility}} \boldsymbol{c}_i$$

with: $T_i = e^{-\sum_{j=1}^{i-1} \sigma_i \delta_i}$

Torsten Sattler, "A Brief Introduction to Neural Radiance Fields", CESCG Academy 2023.

Basics – Volume Rendering





$$\hat{C}(r) = \sum_{i=1}^{N} T_i \underbrace{(1 - e^{(-\sigma_i \delta_i)})}_{\beta_i} \boldsymbol{c}_i \qquad \text{with: } T_i = e^{-\sum_{j=1}^{i-1} \sigma_i \delta_i}$$

Torsten Sattler, "A Brief Introduction to Neural Radiance Fields", CESCG Academy 2023.

Neural Radiance Fields - NeRF

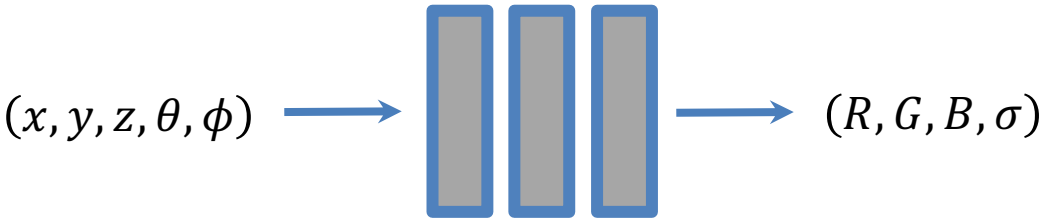
The Volume Rendering equation has some interesting properties

$$\hat{C}(r) = \sum_{i=1}^{N} T_i (1 - e^{(-\sigma_i \delta_i)}) c_i$$

✓Fully differentiable!

Neural Radiance Field: Learn the volumetric representation

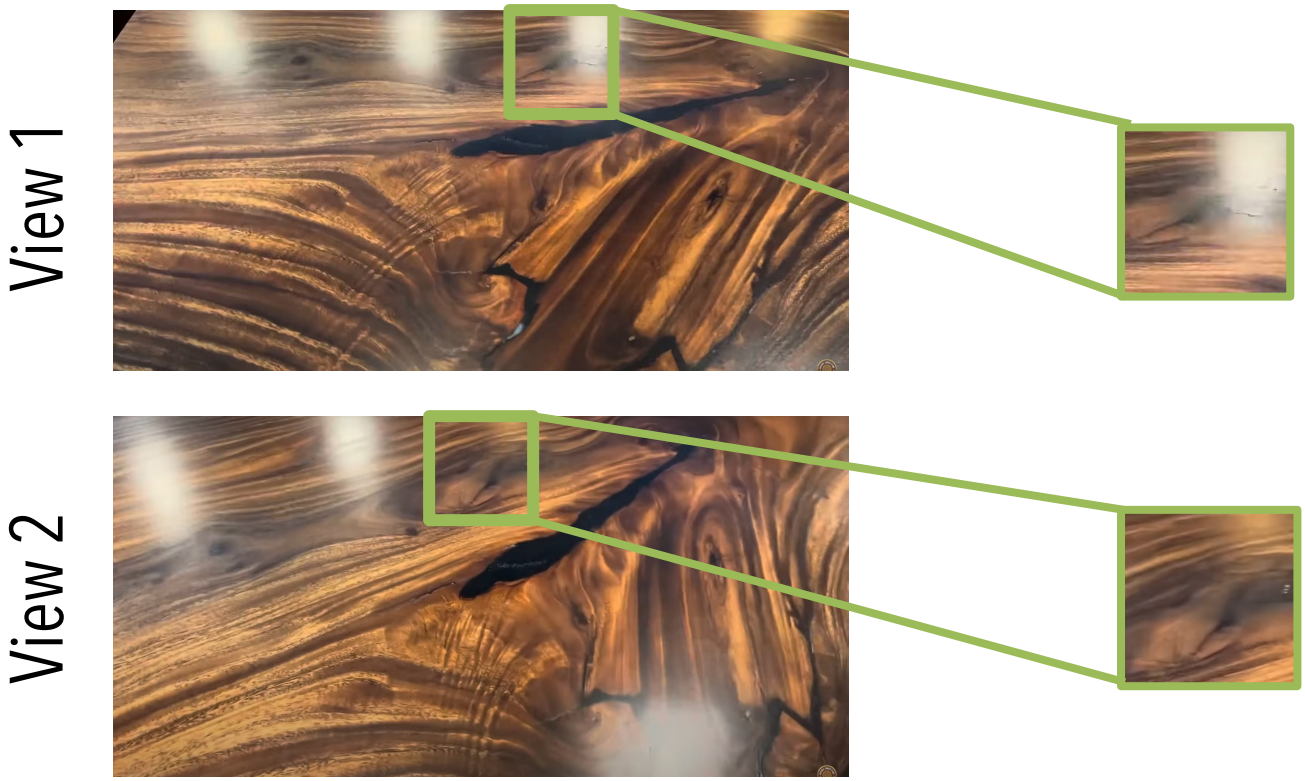
Continuous scene representation (vs. discrete voxel volumes)



Torsten Sattler, "A Brief Introduction to Neural Radiance Fields", CESCG Academy 2023.

Neural Radiance Fields - NeRF

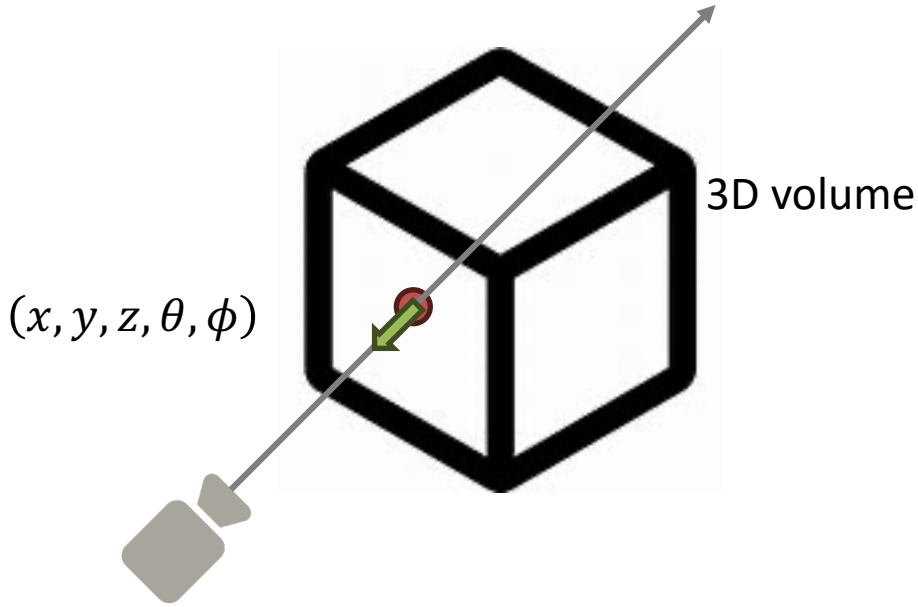
What is this "viewing direction" as input?
The same point in space can have different color depending on the material, lighting and the viewer's position



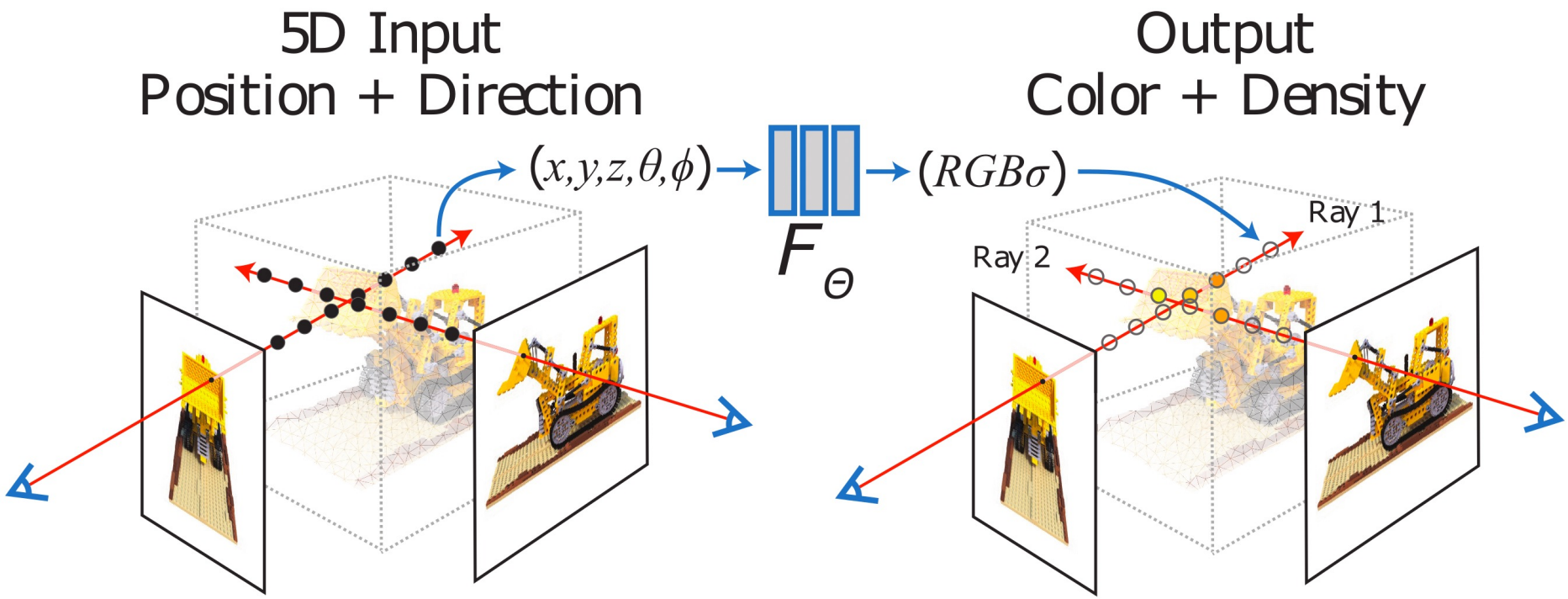
Neural Radiance Fields - NeRF

To deal with that, the color of any 3D point varies as a function of the viewing direction at a given 3D position

Change (θ, ϕ) to visualize view-dependent effects



Neural Radiance Fields - NeRF



Compute the color for each pixel via volume rendering

NeRF - Training

How can we train that??

Posed Images for Training

We assumed calibrated cameras with known poses!

How to get that? Any guess?

Good old Structure from Motion (e.g. Colmap)

Rendering loss

To train a NeRF, we minimize the discrepancy between the rendered pixel and the real one in the training set.

$$\mathcal{L} = \sum_{r \in \mathcal{R}} \left\| \hat{C}_c(r) - C(r) \right\|_2^2$$

Color predicted by the network

Color in real image

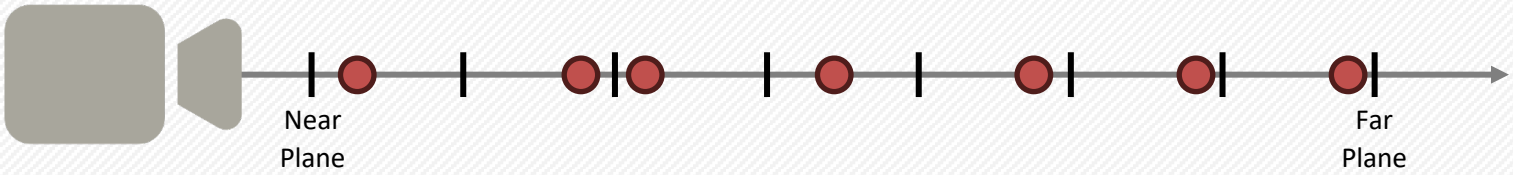
Training a NeRF is almost like overfitting the network to "remember" the scene

NeRF – Training: Volume Sampling

How to select sample in space?

Uniform Sampling

An easy way is to divide the space into equally sized intervals

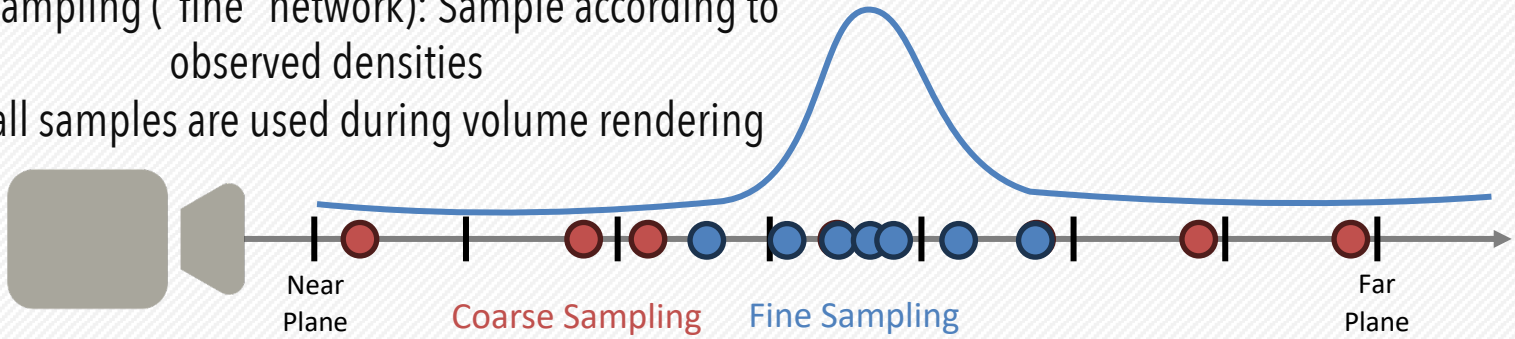


It is inefficient as it allocates equal computation to regions with little or no contribution to the final color, such as empty space.

Hierarchical Volume Sampling

"Fine" sampling ("fine" network): Sample according to observed densities

Note: all samples are used during volume rendering



Torsten Sattler, "A Brief Introduction to Neural Radiance Fields", CESCG Academy 2023.

NeRF – Training: Loss

As you might have understood, we use two networks:

- A coarse network to estimate the overall density along the ray
- A fine network that will mostly focus on samples near the surface

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

Ground truth color

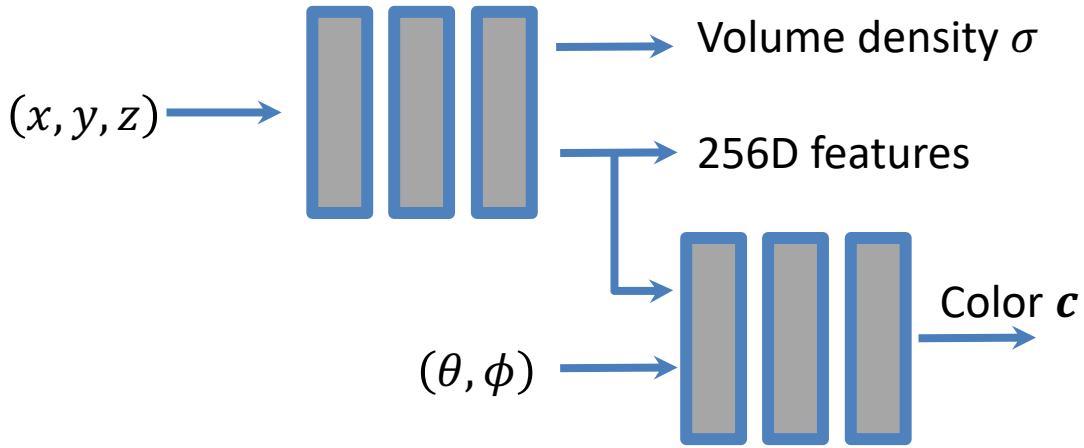
Color predicted by the coarse network

Color predicted by the fine network

NeRF: Important details

A NeRF network is actually divided into two parts

Why?



Because the density does not depend on the viewing direction!

Torsten Sattler, "A Brief Introduction to Neural Radiance Fields", CESCG Academy 2023.

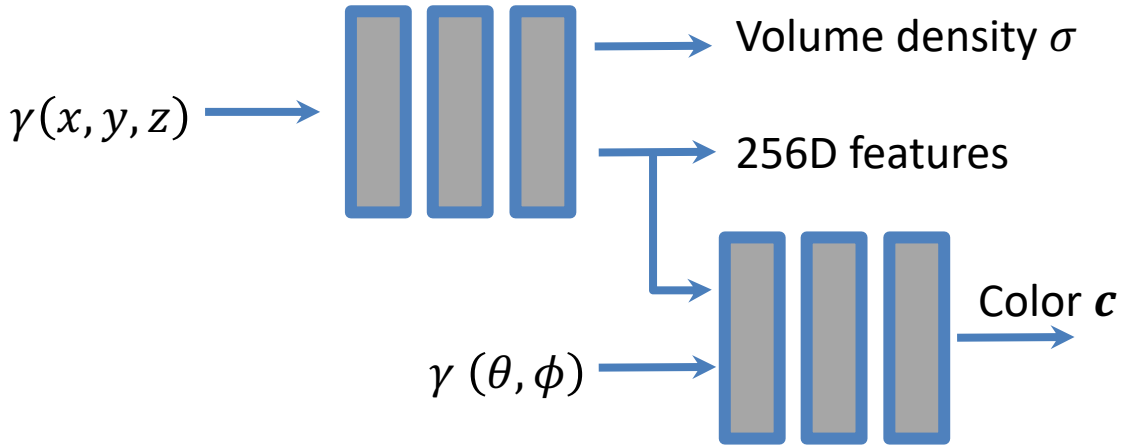
NeRF – Positional Encoding

NeRF (Naive)

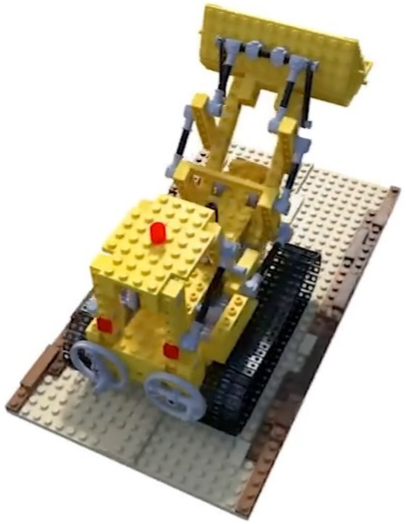
NeRF (with positional encoding)

Positional Encoding

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



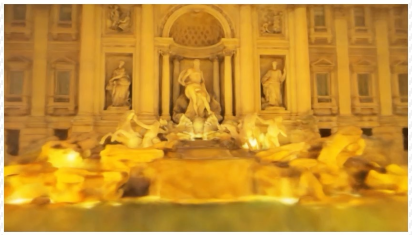
NeRF – Results



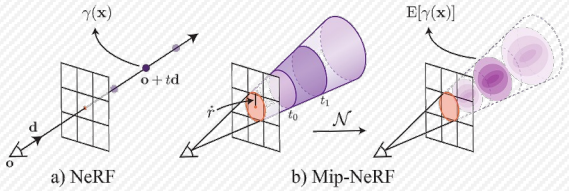
NeRF – Follow-up works

NeRF was the first of many follow-ups and paved the way for numerous novel applications.

"In the Wild" [1]

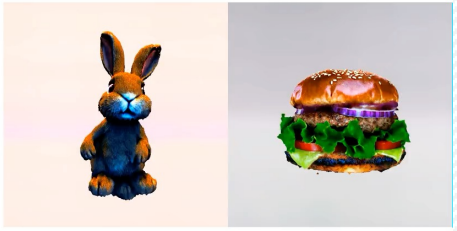


Multi-Scale Representat.



[2]

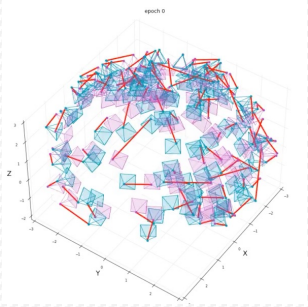
Generative Models



a rabbit, animated movie character, high detail 3d model

a DSLR photo of a delicious hamburger

Uncalibrated [3]



Real-Time [4]

Re-Lighting

[1] Martin-Brualla, Ricardo, et al. "Nerf in the wild: Neural radiance fields for unconstrained photo collections." Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2021.
[2] Barron, Jonathan T., et al. "Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields." Proceedings of the IEEE/CVF international conference on computer vision. 2021.
[3] Lin, Chen-Hsuan, et al. "Barf: Bundle-adjusting neural radiance fields." *Proceedings of the IEEE/CVF international conference on computer vision*. 2021.
[4] Müller, Thomas, et al. "Instant neural graphics primitives with a multiresolution hash encoding." ACM transactions on graphics (TOG) 41.4 (2022): 1-15.

NeRF - Limitations

Despite impressive results, NeRF suffers a few limitations. Among them here are probably the most challenging

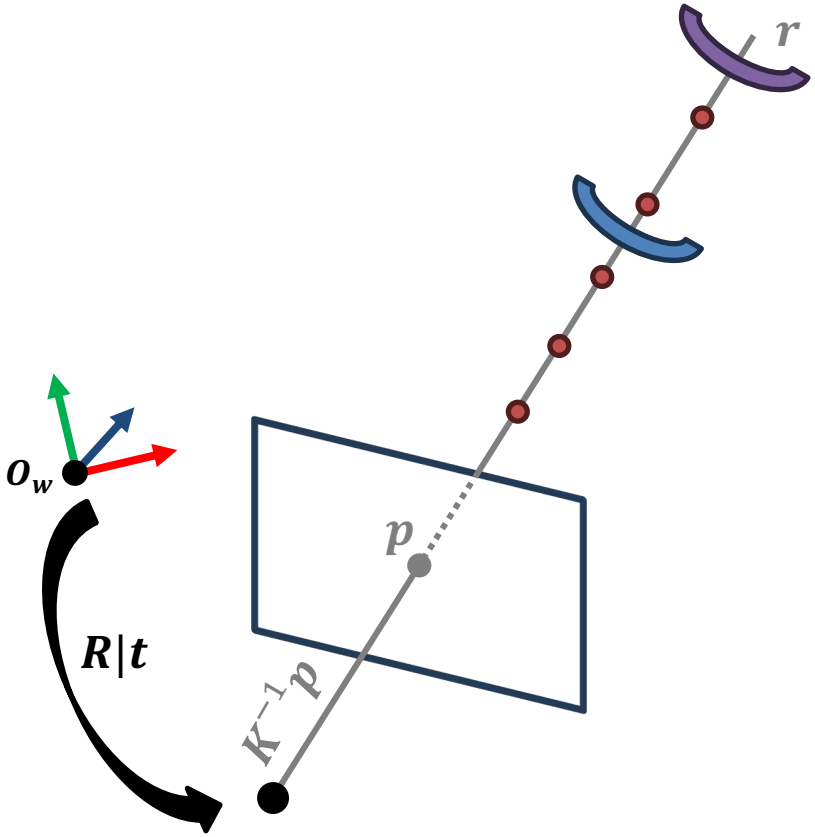
Unsuitable for Large Scale Scenes

Very Slow To Train & Render!

The original NeRFs are very slow to train and very slow to render!

Why are NeRFs so Slow?

For each pixel to render, it needs to evaluate the network … multiple times!



NeRF, Parameterize Radiance Field densely, at every point in space … But we just care about points with density?!?

This is exactly the problem tackled by Gaussian Splatting!

Gaussian Splatting

References:

[1] Kerbl, B., Kopanas, G., Leimkühler, T., & Drettakis, G. (2023). *3d gaussian splatting for real-time radiance field rendering*. ACM Trans. Graph., 42(4), 139-1.

Note: Some of the slides used in this presentation are reproduced from *Stanford CS231A* *Computer Vision: From 3D Reconstruction to Recognition*, Silvio Savarese & Jeannette Bohg, 2024.

Another relevant reference: "A comprehensive Overview of Gaussian Splatting"

Gaussian Splatting

3D Gaussian Splatting for Real-Time Radiance Field Rendering

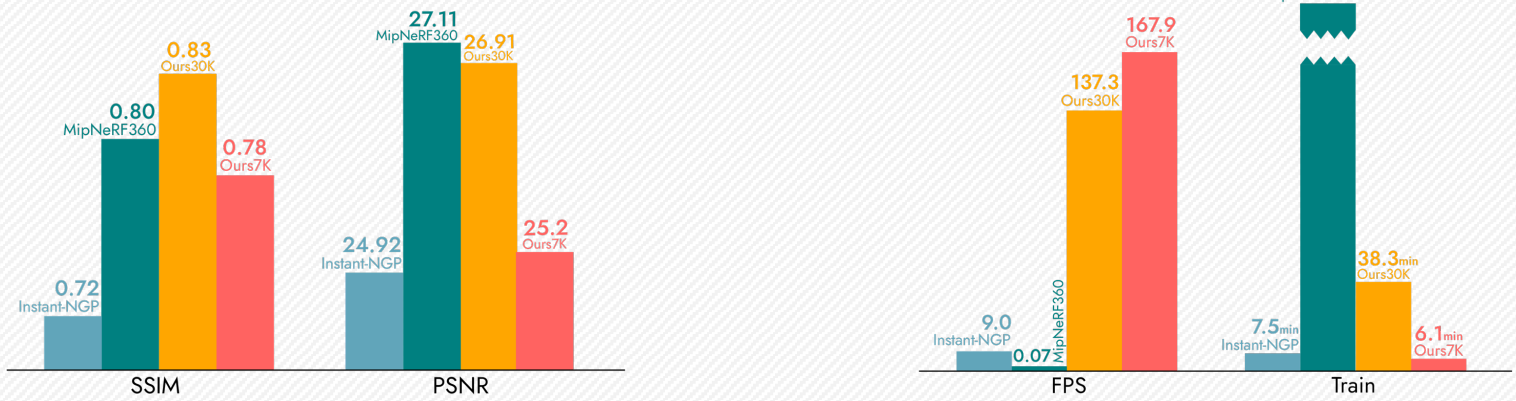
BERNHARD KERBL*, Inria, Université Côte d'Azur, France
GEORGIOS KOPANAS*, Inria, Université Côte d'Azur, France
THOMAS LEIMKÜHLER, Max-Planck-Institut für Informatik, Germany
GEORGE DRETTAKIS, Inria, Université Côte d'Azur, France

SIGGRAPH 2023
(ACM Transactions on Graphics)

SUNY Korea
Inria
UNIVERSITÉ CÔTE D'AZUR
max planck institut informatik

Gaussian Splatting – Speed up

Quality vs Speed

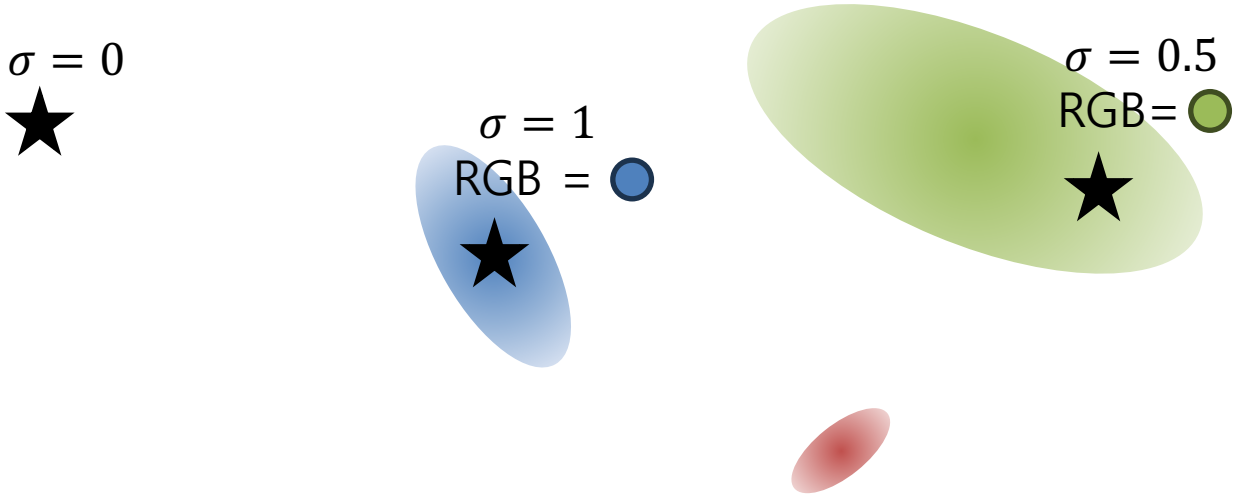


Qualitative Results

What is the magic behind it??

Gaussian Splatting – Main Concept

Key idea: Parameterize the Radiance Field sparsely **only** where density is nonzero



3D Gaussian Blobs floating in Space

GS– Elliptical Gaussian Kernels

Each volume splat is defined has an elliptical Gaussian Kernel defined by the following attributes:

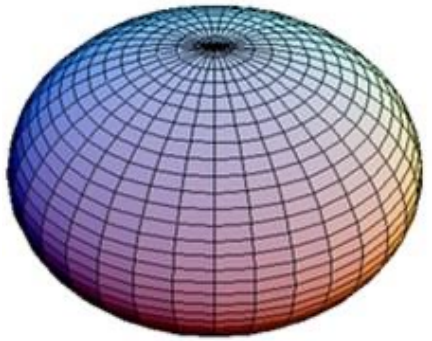
Opacity

σ

Color

$\boldsymbol{c}$

Or Spherical Harmonic

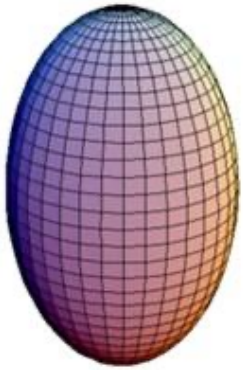


Center Point

$\boldsymbol{\mu} \in \mathbb{R}^3$

Covariance Matrix

$\sum \in \mathbb{R}^{3\times3}$



Each Gaussian kernel G is defined as follows:

$$G(\boldsymbol{p};\boldsymbol{\mu},\sum) = \frac{1}{(2\pi)^{\frac{3}{2}}|\sum|^{\frac{1}{2}}} e^{-\frac{1}{2}(\boldsymbol{\mu}-\boldsymbol{p})^T\sum^{-1}(\boldsymbol{\mu}-\boldsymbol{p})}$$

Gaussian Splatting

Gaussians floating around!

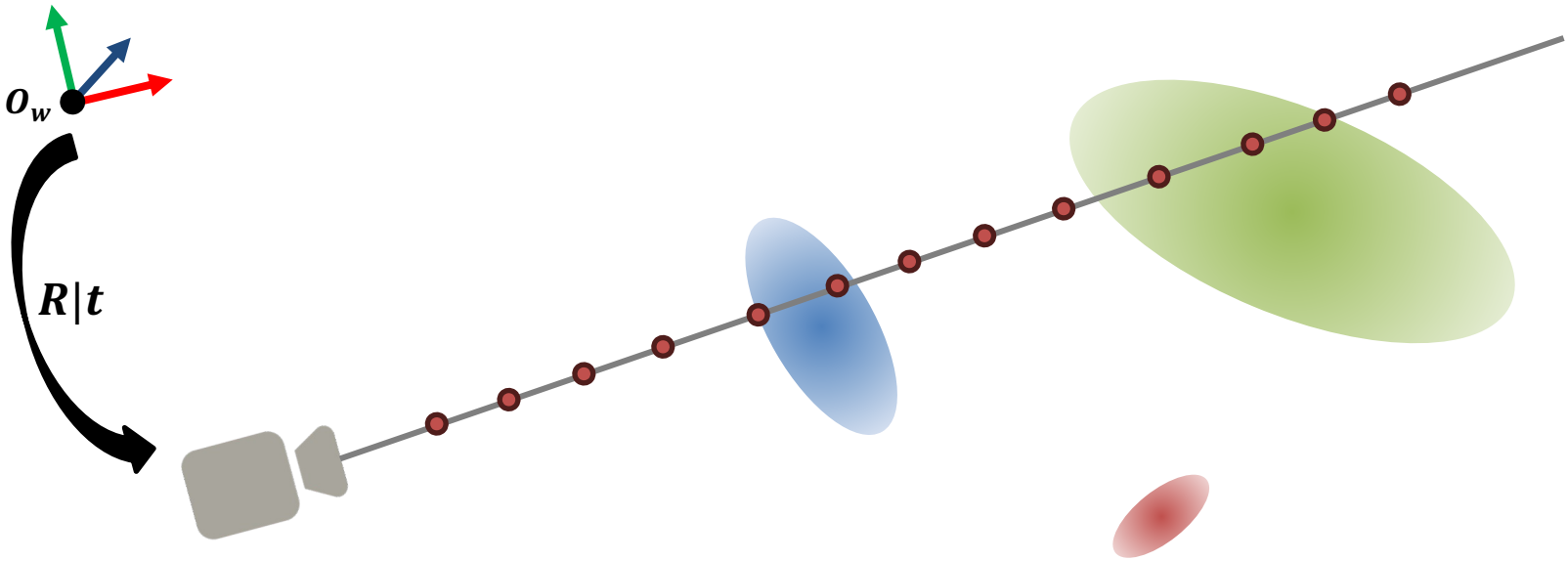
Anisotropic Volumetric 3D Gaussians

Final Rendering

3D Gaussian Visualization

Gaussian Splatting – Rendering

How to render that? Any ideas?



Why not volume integral like NeRFs?

Then we will still sample in many zero-density places …

But we know the locations where the density is non-zero!

Gaussian Splatting – The Splat!

Instead of sampling, what if we could directly project our 3D Gaussians on our image?

Projecting the mean

$$\begin{bmatrix} u \\ v \\ z \end{bmatrix} = \boldsymbol{K}[\boldsymbol{R}|\boldsymbol{t}] \begin{bmatrix} \mu_x \\ \mu_y \\ \mu_z \\ 1 \end{bmatrix} = \boldsymbol{KW} \begin{bmatrix} \mu_x \\ \mu_y \\ \mu_z \\ 1 \end{bmatrix}$$

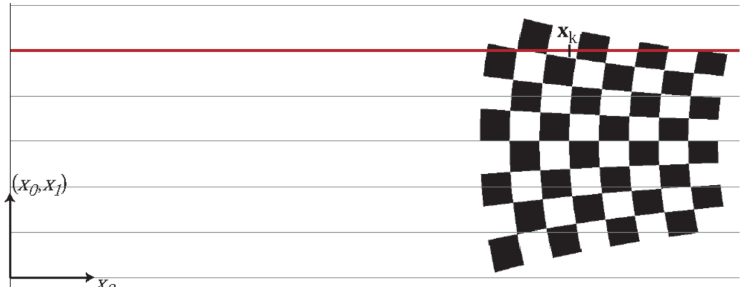
$$\mu^{2D} = \begin{bmatrix} \mu_x^{2D} \\ \mu_y^{2D} \end{bmatrix} = \begin{bmatrix} u/z \\ v/z \end{bmatrix}$$

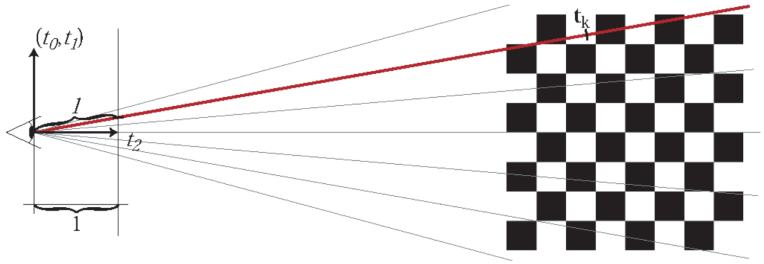
Projecting the covariance

The covariance requires a first-order approximation due to projective geometry's non-linearity.

$$\boldsymbol{\Sigma}^{2D} = \boldsymbol{JW\Sigma W^T J^T}$$

$$J = \frac{\delta \mu^{2D}(\mu)}{\delta \mu}$$





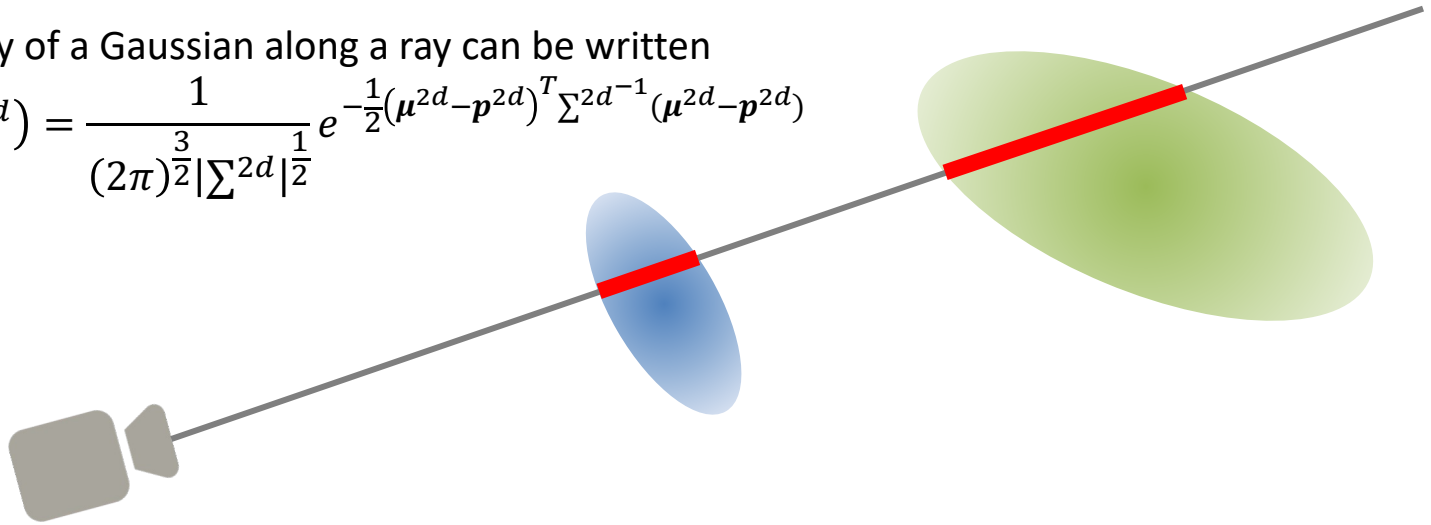
Zwicker, Matthias, et al. "EWA splatting." *IEEE Transactions on Visualization and Computer Graphics* 8.3 (2002): 223-238.

Gaussian Splatting – Integral

We can compute the integral without sampling!

The density of a Gaussian along a ray can be written

$$G^{2d}(\boldsymbol{p}^{2d}) = \frac{1}{(2\pi)^{\frac{3}{2}}|\sum^{2d}|^{\frac{1}{2}}} e^{-\frac{1}{2}(\boldsymbol{\mu}^{2d}-\boldsymbol{p}^{2d})^T \sum^{2d\,-1}(\boldsymbol{\mu}^{2d}-\boldsymbol{p}^{2d})}$$



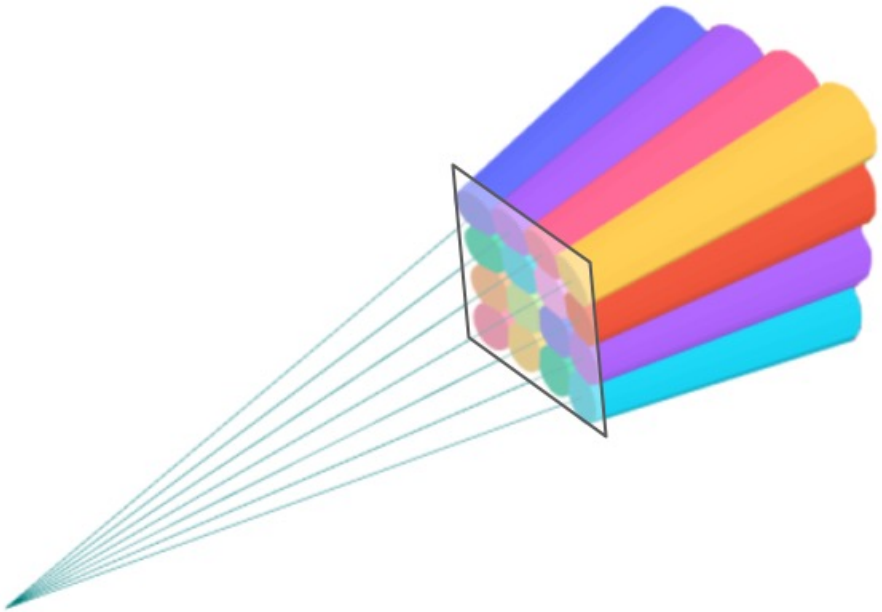
We just need to sort the 3D Gaussian per distance and to integrate until saturation!

$$C = \sum_{i \in N} c_i \alpha_i \prod_{j=1}^{i-1}(1-\alpha_j)$$

$$= \sum_{i \in N} c_i G_i^{2d}(\boldsymbol{p}^{2d}) \prod_{j=1}^{i-1}\left(1 - G_i^{2d}(\boldsymbol{p}^{2d})\right)$$

Gaussian Splatting – Integral

To speed up sorting and blending, the image is divided into tiles, restricting computations to relevant Gaussians within each tile.

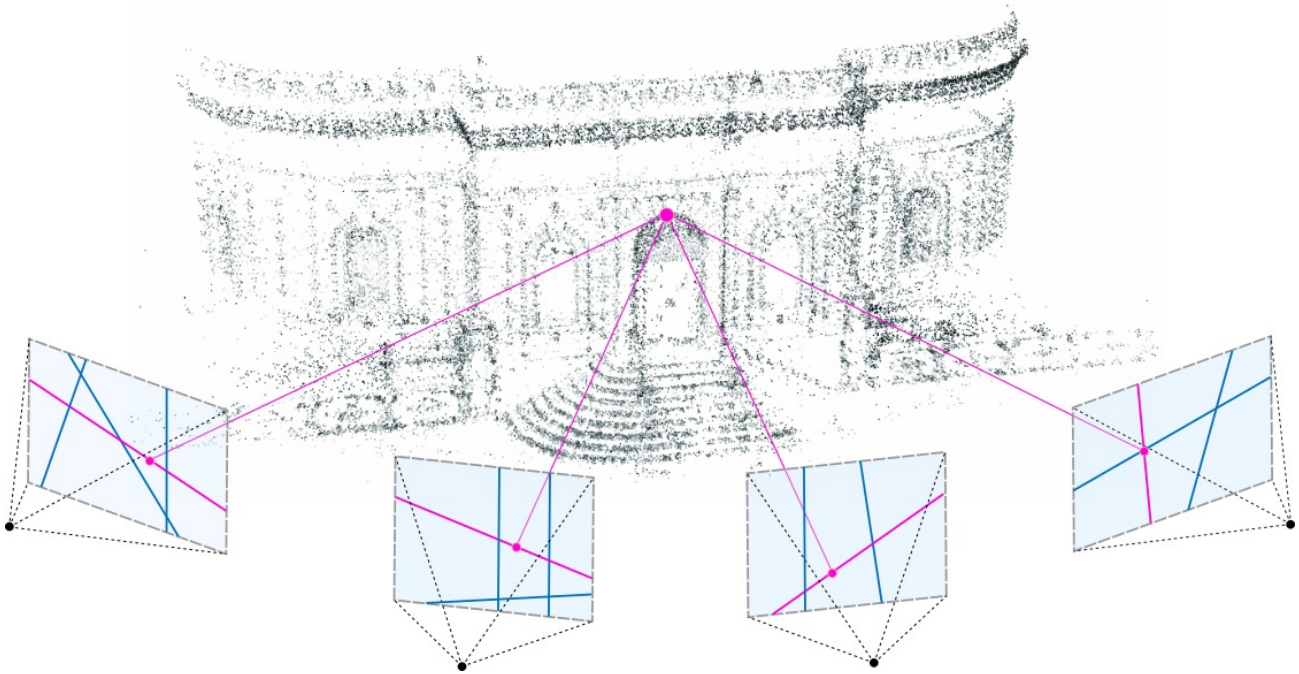


Source: "A comprehensive Overview of Gaussian Splatting"

Gaussian Splatting – Initialization

How to initialize that? Any ideas?

Randomly? It seems really unlikely to converge!



We use the 3D points from SfM!

Gaussian Splatting – Training pipeline

After initialization, Stochastic Gradient Descent optimizes the scene using a loss combining L1 and D-SSIM between the ground truth and current render.

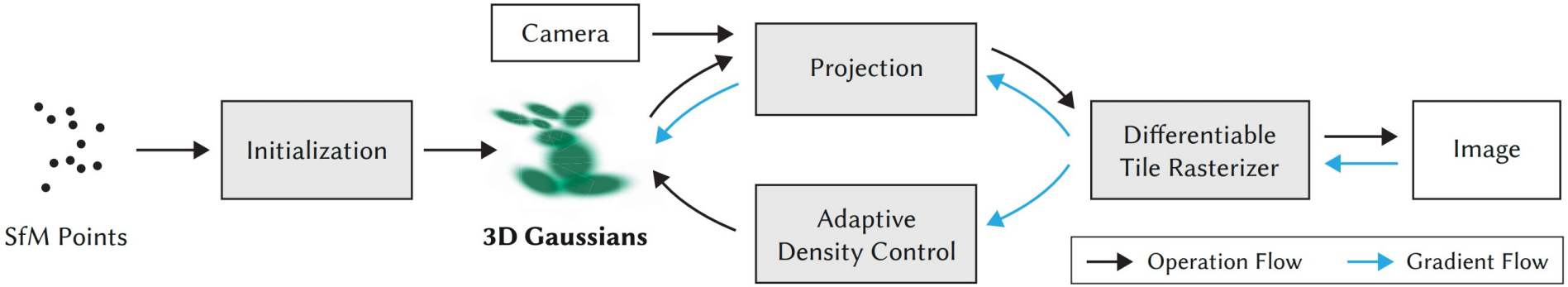
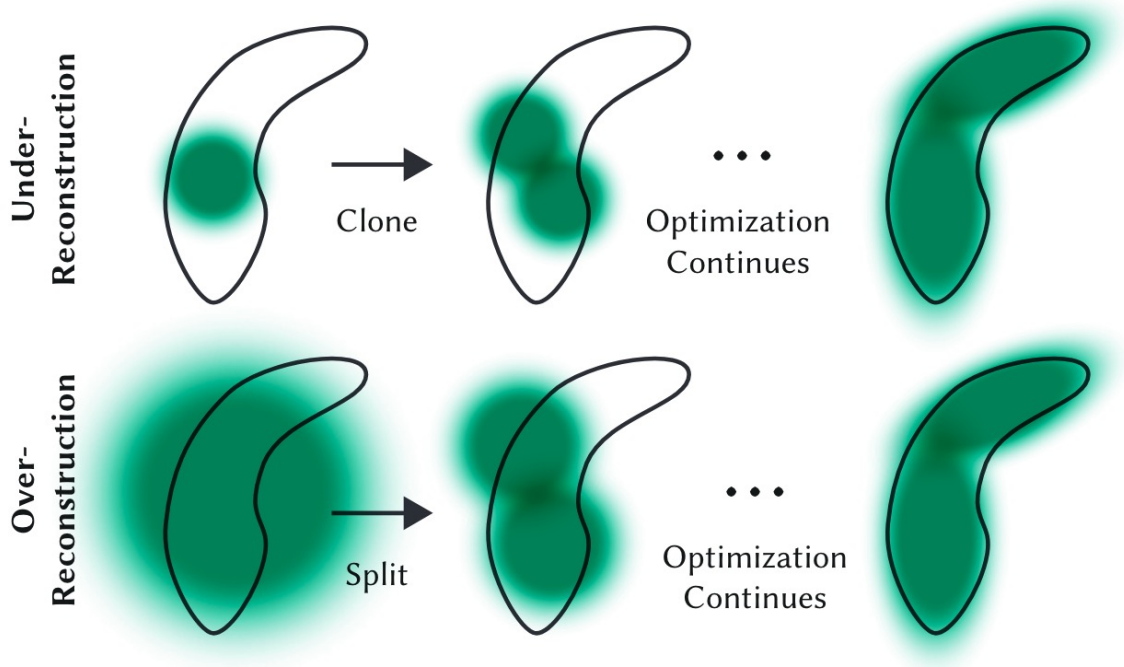


Fig. 2. Optimization starts with the sparse SfM point cloud and creates a set of 3D Gaussians. We then optimize and adaptively control the density of this set of Gaussians. During optimization we use our fast tile-based renderer, allowing competitive training times compared to SOTA fast radiance field methods. Once trained, our renderer allows real-time navigation for a wide variety of scenes.

GS– Pruning and Spanning

Every 100 iterations, adaptive densification refines the point distribution by splitting points with large gradients to address under-reconstruction and removing points with low transparency to prevent over-reconstruction.



Gaussian Splatting – Results

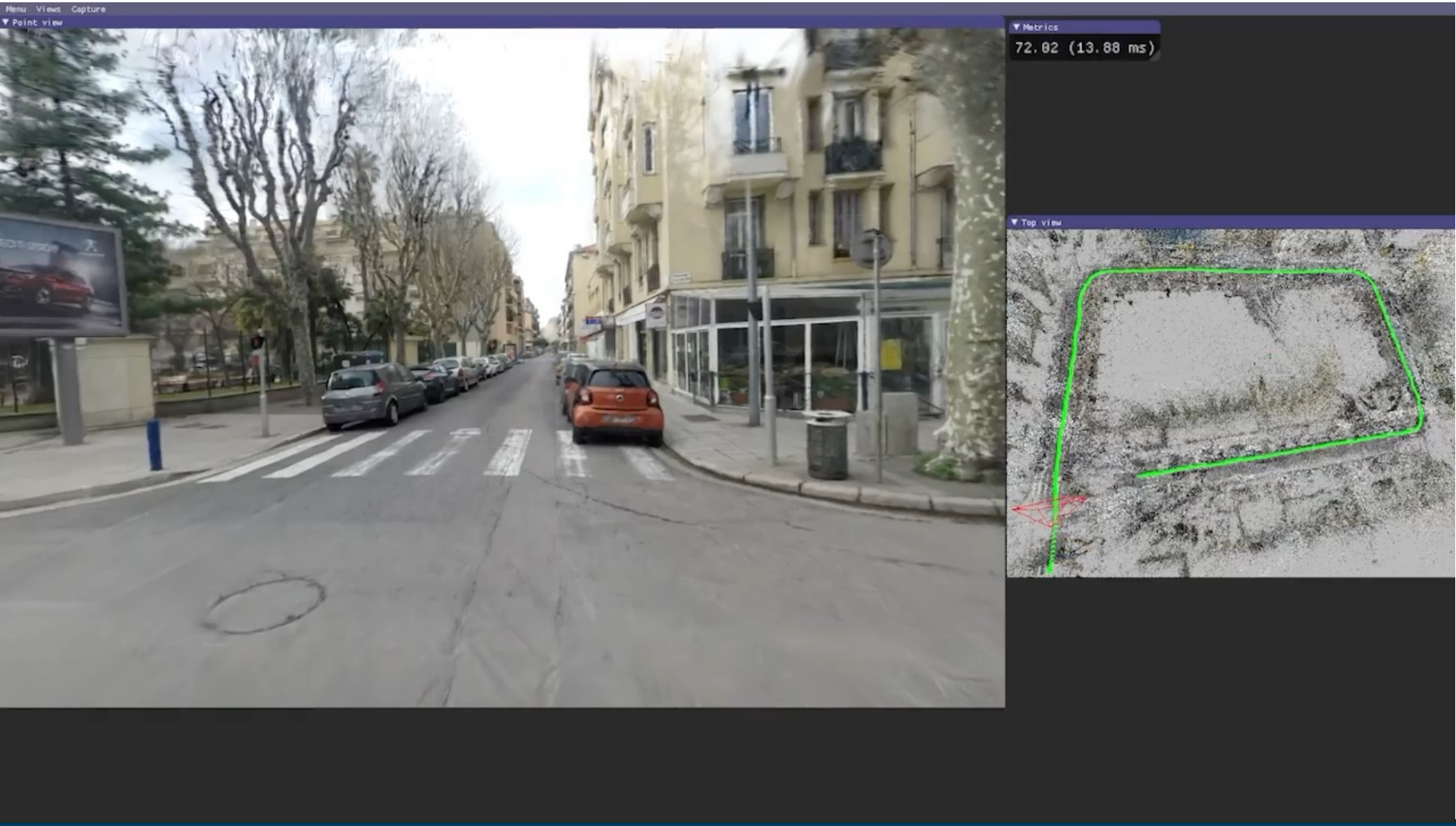
Visual Comparisons

MipNeRF360 [Barron '22]

Ours

Gaussian Splatting – Results

Recent works tackle the problem of large scale GS reconstruction!



Gaussian Splatting – Conclusion

Gaussian Splatting leads to fantastic results:

- photorealistic results
- real time rendering
- Fast Training
- Compact Representation

But it requires many assumptions:

- Known sparse 3D point cloud
- Known camera poses from SfM
- And it is a "per-scene" solution
- It requires many images of the scene

One shot Reconstruction

References:
[1] Weinzaepfel, P., Leroy, V., Lucas, T., Brégier, R., Cabon, Y., Arora, V., ... & Revaud, J. (2022). Croco: Self-supervised pre-training for 3d vision tasks by cross-view completion. Advances in Neural Information Processing Systems, 35, 3502-3516.
[2] Wang, S., Leroy, V., Cabon, Y., Chidlovskii, B., & Revaud, J. (2024). *Dust3r: Geometric 3d vision made easy*. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 20697-20709).
[3] Leroy, V., Cabon, Y., & Revaud, J. (2025). *Grounding image matching in 3d with mast3r*. In European Conference on Computer Vision (pp. 71-91). Springer, Cham.

Note: Many slides used in this presentation are reproduced from the presentation *From CroCo to MASt3R*, Jerome Revaud, Naver Labs Europe, 2024

Direct Feed-Forward Network

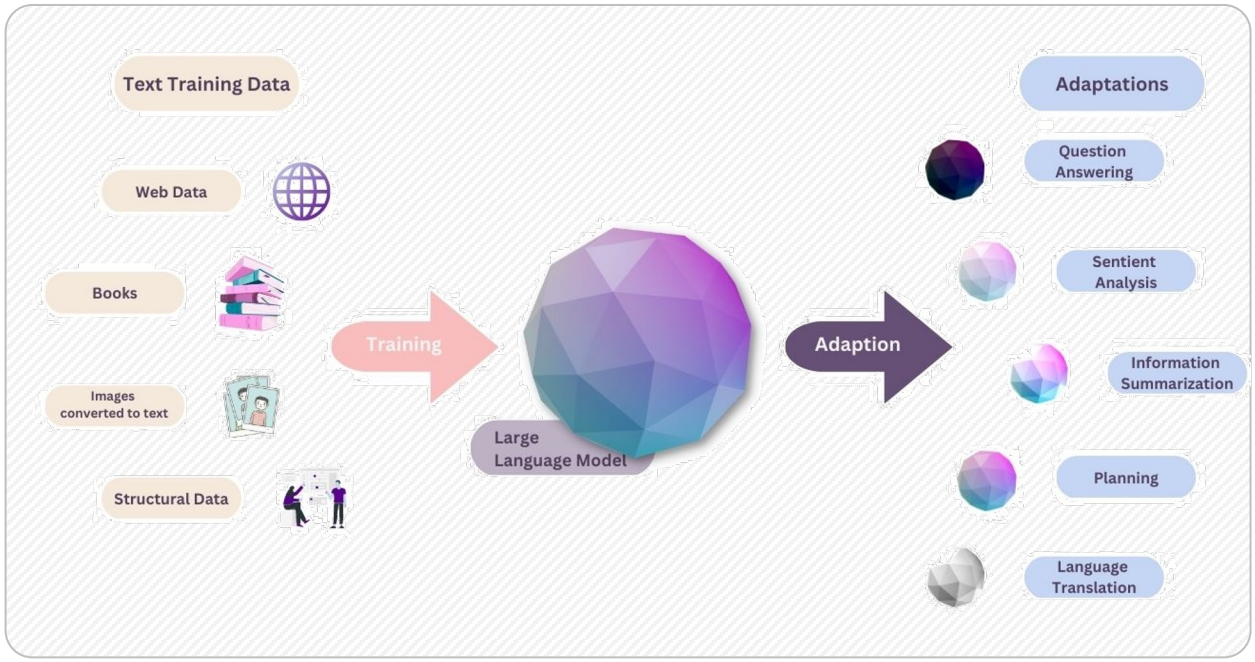
Unlike per-scene fitting such as NeRF or GS, a new trend is emerging: The end-to-end 3D reconstruction via Scene Coordinate Regression

In the past couple of years, the vision of Naver Labs Europe has been to create 3D Fundamental Models.

Fundamental 3D Model

Before Large Language Models (LLMs), NLP tasks were achieved with specialized networks (OCR, Translation, etc..)

The use of unified, fundamental models has enhanced performances on all tasks!

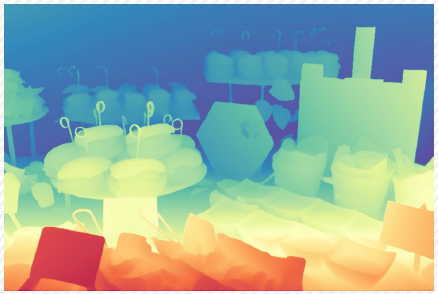


Can we envision something similar for 3D vision?

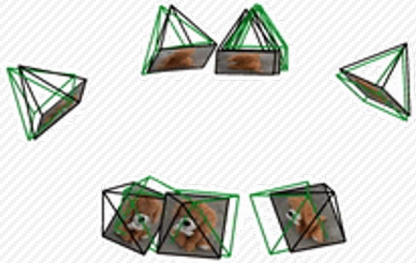
Fundamental 3D Model

What about 3D? What tasks are we addressing?

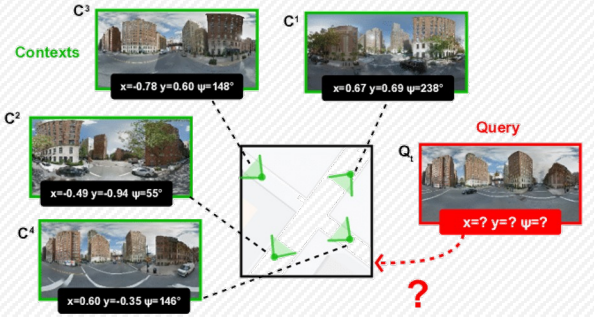
Depth Estimation



Relative Pose Estimation

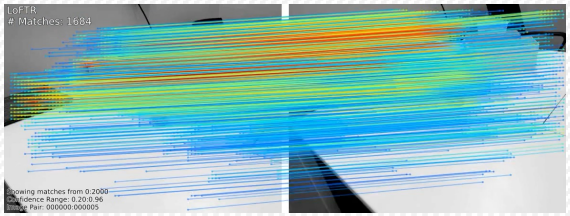


Visual Localization



3D Reconstruction

Point Matching

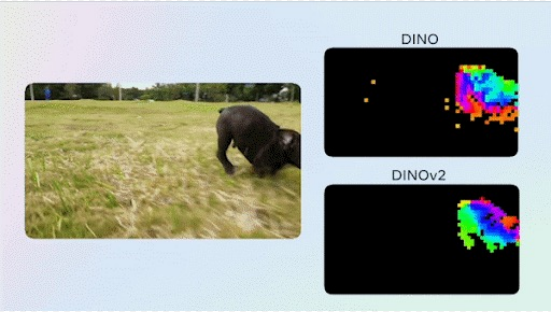


Anything else?
Can we do all of these with a single model??

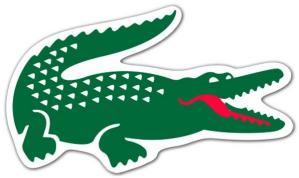
Fundamental 3D Model

What about Self-Supervised Learning?

In 2D SSL did great!



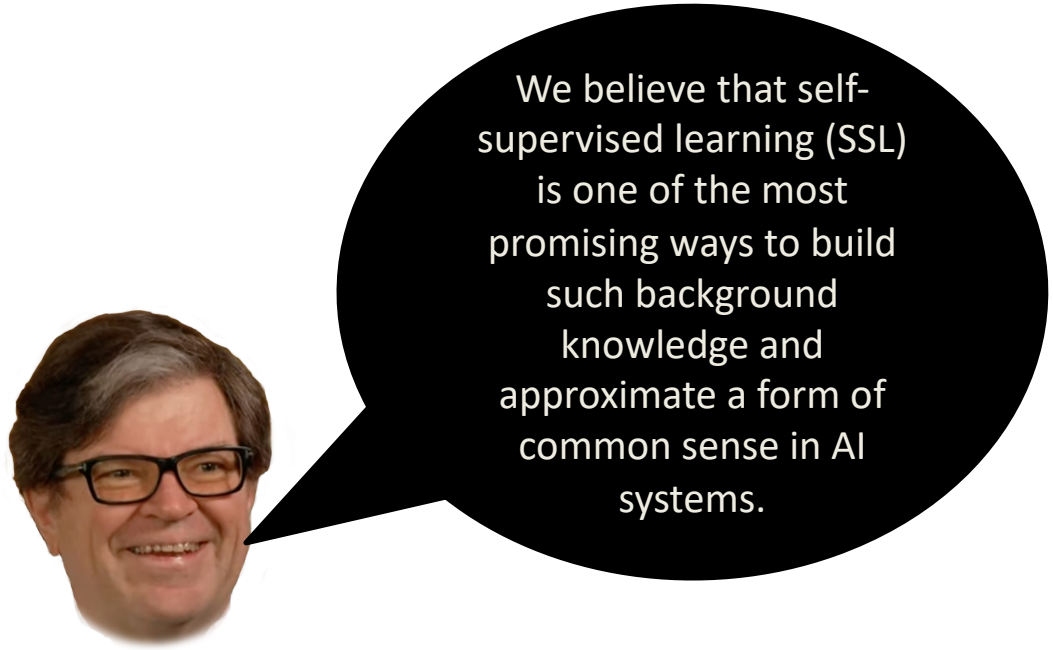
In 3D?

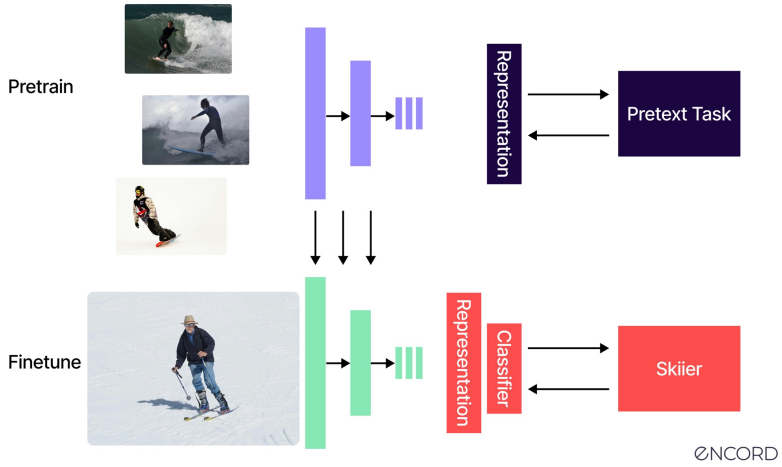



Croco

Self-Supervised Learning

Self-supervised learning (SSL) is a paradigm in machine learning where a model is trained on a pretext task using the data itself to generate supervisory signals, rather than relying on externally-provided labels.

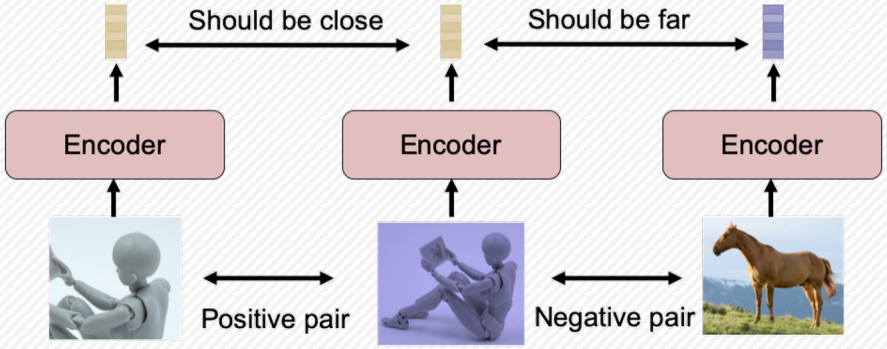




Self-Supervised Learning

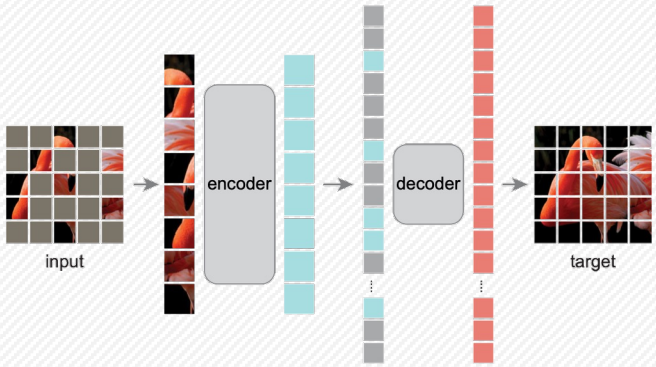
Some representative SSL pipelines

Contrastive Approaches



trains a model by distinguishing augmented views of the same data sample (positives) from other samples (negatives) in a learned feature space.

Generative Approaches



Generative self-supervised learning trains a model to reconstruct or predict missing parts of data, enabling it to learn meaningful representations.

CroCO

Great, but such SSLs do not account for 3D. What pretext task can we use such that the network learns some "common sense" about 3D perception??

CroCo: Self-Supervised Pre-training for 3D Vision Tasks by Cross-View Completion

---

Philippe Weinzaepfel Vincent Leroy Thomas Lucas

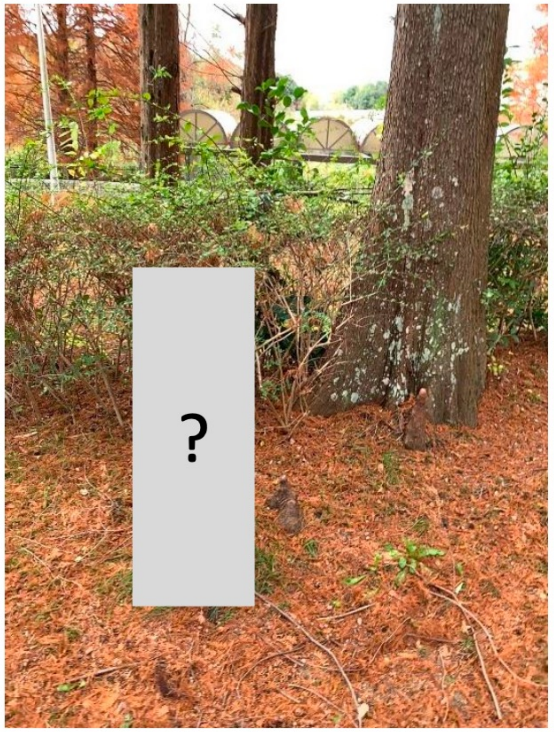
Romain Brégier Yohann Cabon Vaibhav Arora Leonid Antsfeld

Boris Chidlovskii Gabriela Csurka Jérôme Revaud

CroCO: The concept

Given a pair of images, the pretext task is to complete the query image where some areas have been arbitrarily masked.
Is it a generative or ~~contrastive~~ SSL approach?

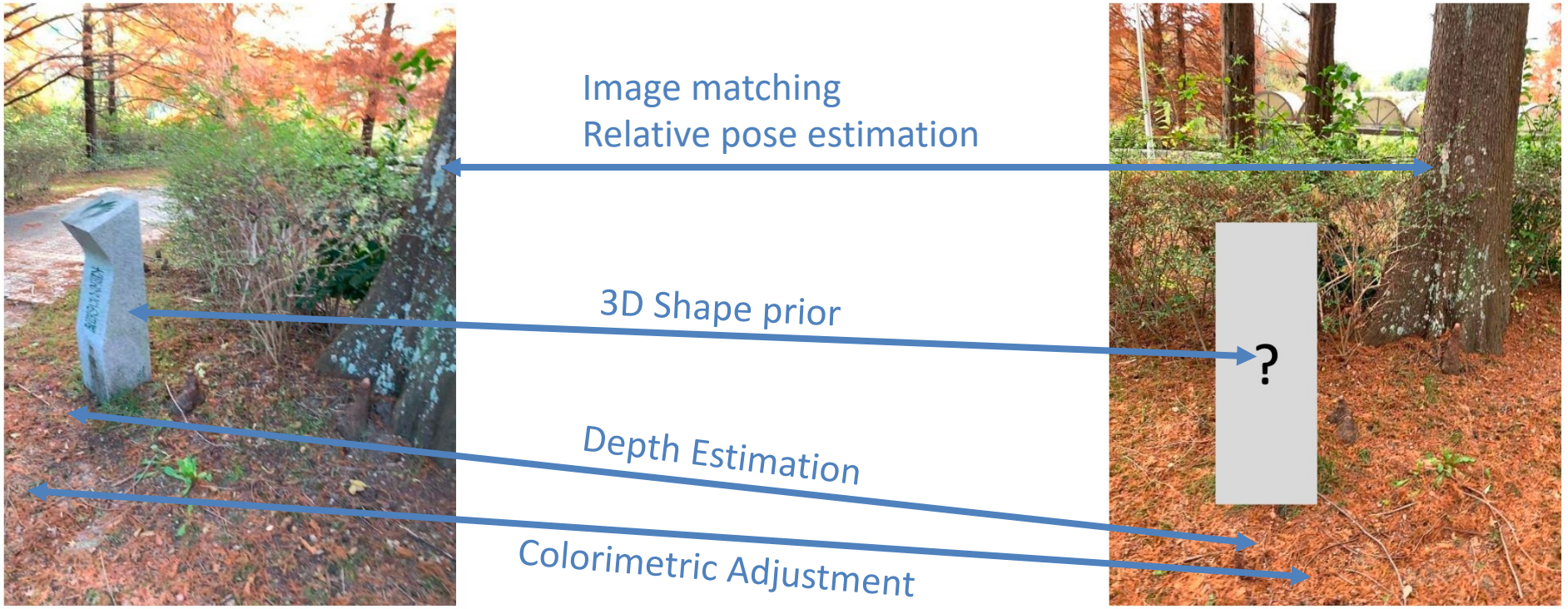
Reference view



Query view

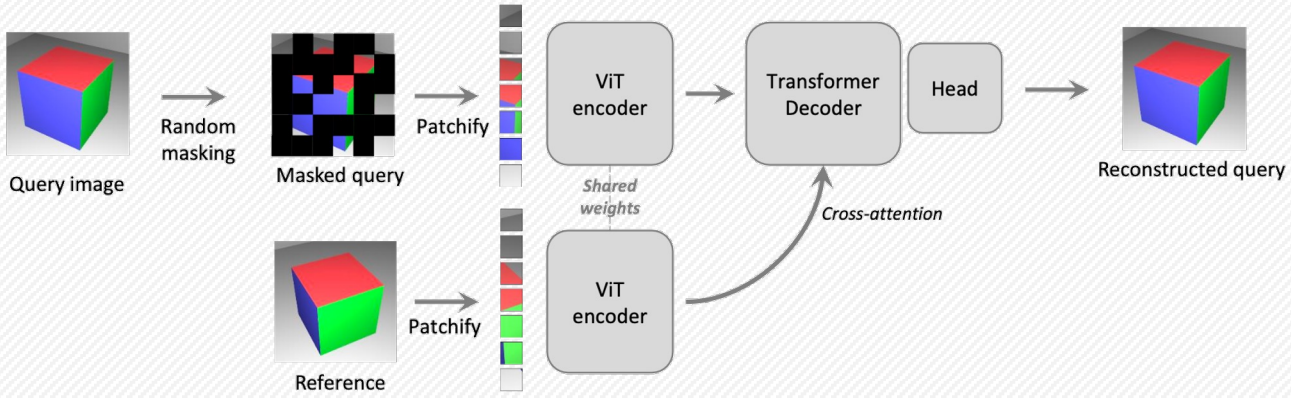
CroCO: The concept

How is this task related to 3D vision??

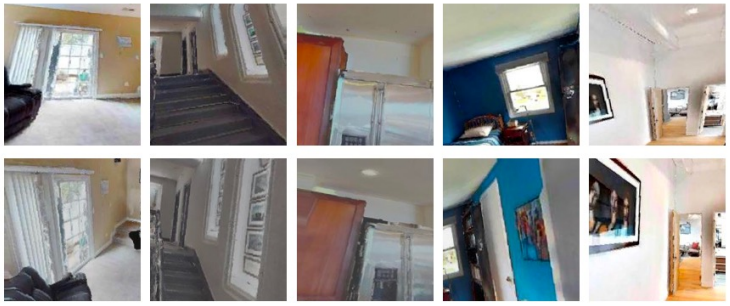


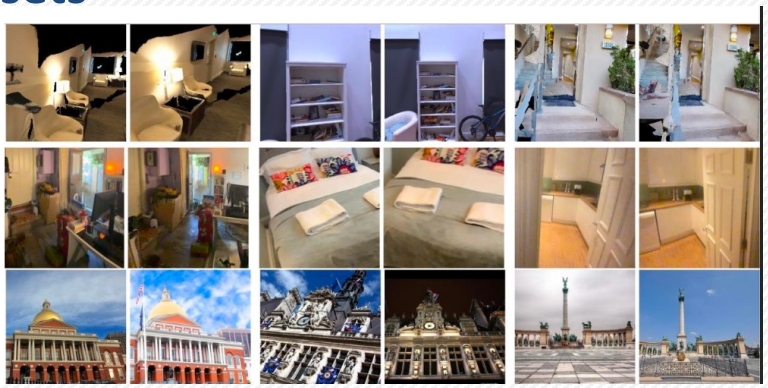
CroCO: Model and Data

CroCo Training Pipeline



Datasets





2M image pairs from the Habitat simulator + 5M real images

CroCO: Datasets

Pretraining a network using CroCo leads to SoTa results for many tasks, such as:

- Monocular Depth
- Absolute pose regression
- Optical Flow
- Stereo Matching, etc….

Table 8: **Absolute pose regression results** as averaged median errors over the 7 scenes for different backbones, pre-training methods and datasets. Each column corresponds to different ratios of the training set. Best result per column on bold and second best underlined.

Architecture	pre-training	100%	20%	10%	5%
ResNet34	Supervised (ImageNet)	**27.1cm**, 9.0°	34.0cm, 10.9°	43.7cm, 130.°	62.3cm, 17.3°
ViT-Base/16	MAE (ImageNet)	27.9cm, 9.0°	28.0cm, 8.5°	30.6cm, 9.2°	34.4cm, **9.4°**
ViT-Base/16	MAE (Habitat)	28.3cm, 9.1°	28.0cm, 8.3°	30.7cm, 9.1°	35.3cm, 10.1°
ViT-Base/16	MultiMAE (ImageNet)	33.1cm, 9.8°	32.6cm, 9.7°	36.8cm, 10.8°	44.1cm, 12.2°
ViT-Base/16	CroCo CrossBlock (Habitat)	27.7cm, **8.6°**	**26.3cm, 7.3°**	**29.0cm, 8.3°**	**32.7cm**, 9.5°
ResNet34	Random	28.1cm, 8.9°	36.6cm, 11.4°	51.3cm, 14.0°	69.3cm, 17.6°
ViT-Base/16	Random	29.1cm, 9.3°	38.3cm, 11.6°	43.6cm, 12.5°	52.7cm, 14.1°

Table 5: **Relative pose estimation results** with the median camera position and orientation errors on 7-scenes. Finetuning a model pre-trained with CroCo achieves competitive results compared to existing methods directly regressing relative camera pose, without applying any fusion technique.

Method / pre-training	chess	fire	heads	office	pumpkin	redkitchen	stairs	Average
RelocNet* [6]	12cm, 4.14°	26cm, 10.44°	14cm, 10.5°	18cm, 5.32°	26cm, 4.17°	23cm, 5.08°	28cm, 7.53°	21cm, 6.74°
NC-EssNet* [96]	12cm, 5.63°	26cm, 9.64°	14cm, 10.66°	20cm, 6.68°	22cm, 5.72°	22cm, 6.31°	31cm, 7.88°	21cm, 7.50°
CamNet*† [25]	4cm, **1.73°**	**3cm, 1.74°**	5cm, **1.98°**	4cm, **1.62°**	**4cm, 1.64°**	**4cm, 1.63°**	**4cm, 1.51°**	**4cm, 1.69°**
top1 AP-GeM-18	27.9cm, 12.81°	40.4cm, 16.06°	21.6cm, 16.46°	37.5cm, 12.79°	44.4cm, 12.58°	46.7cm, 13.92°	32.2cm, 14.59°	36cm, 14.2°
MAE (Habitat)	13.2cm, 9.44°	32.0cm, 15.10°	16.0cm, 16.75°	24.8cm, 11.54°	25.4cm, 10.62°	29.4cm, 13.32°	32.8cm, 14.88°	24.8cm, 13.09°
CroCo (Habitat)	**2.4cm**, 2.81°	4.0cm, 3.86°	**3.1cm**, 4.00°	**3.4cm**, 2.53°	4.9cm, 2.79°	5.5cm, 3.72°	11.7cm, 4.53°	5.0cm, 3.46°

*: fuse multiple pose predictions †: exploit temporal information and multi-step retrieval

Table 4: **Optical flow results** on the training set of the MPI-Sintel dataset for various pre-training methods when finetuned on AutoFlow.

encoder init.	decoder init.	MPI-Sintel clean	MPI-Sintel final
random	random	18.81	18.97
MAE (IN1K)	random	4.68	5.16
MAE (Habitat)	random	4.63	5.24
CroCo (Habitat)	CroCo (Habitat)	**3.00**	**3.60**

CroCo

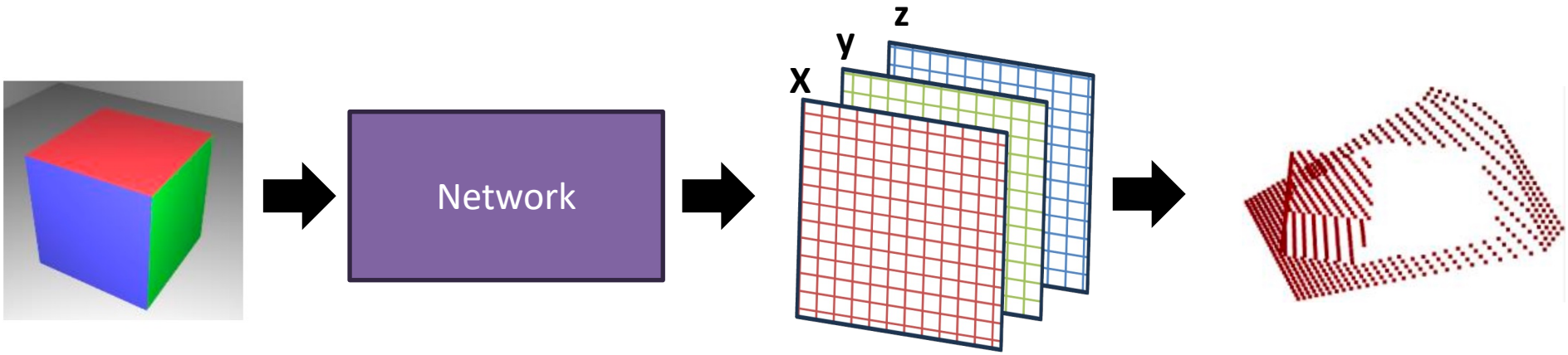
CroCo is a great pre-training pipeline that allows to fine-tune the model for many downstream tasks, but it is NOT a unified model!

How can we unify 3D tasks through a single model??
The trick is to find a representation that incorporates all the 3D vision task implicitly:

Point Map Regression

Point Map Regression

Point Map Regression is only the computation of the 3D points in a given referential!



We can understand it as a point map regression: 1-to-1 mapping between pixels and their corresponding 3D points

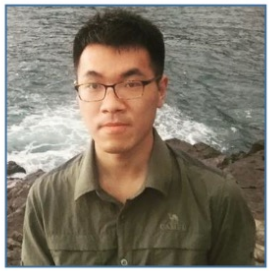
Most geometric tasks are based on 2D-3D correspondence!


DUst3R

DUst3r

Dust3r stands for "Dense Unconstrained Stereo 3D Reconstruction"
But the paper is entitled

DUSt3R: Geometric 3D Vision Made Easy



Shuzhe Wang
Aalto University

Vincent Leroy
Naverlabs Europe

Yohann Cabon
Naverlabs Europe

Boris Chidlovskii
Naverlabs Europe

Jérome Revaud
Naverlabs Europe

Aalto University

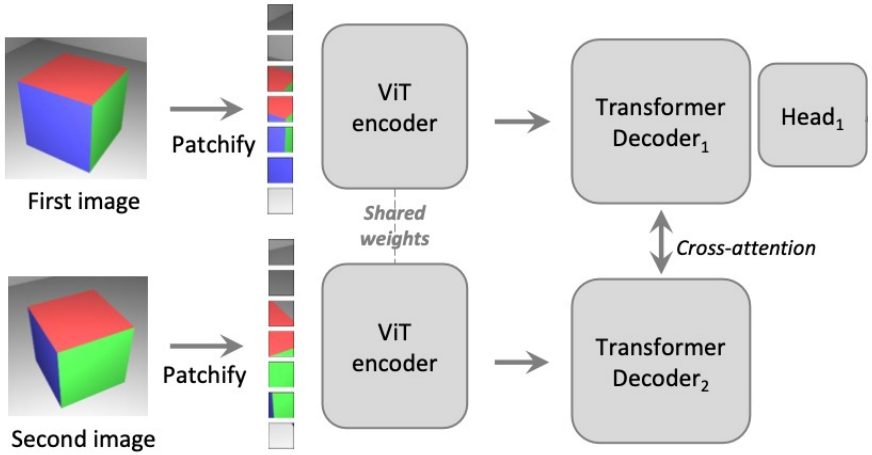
NAVER LABS Europe

CVPR 2024 • SEATTLE

SUNY Korea

DUst3r

DUst3r is a transformer based Point map regression network which can estimate the 3D coordinates of each pixel

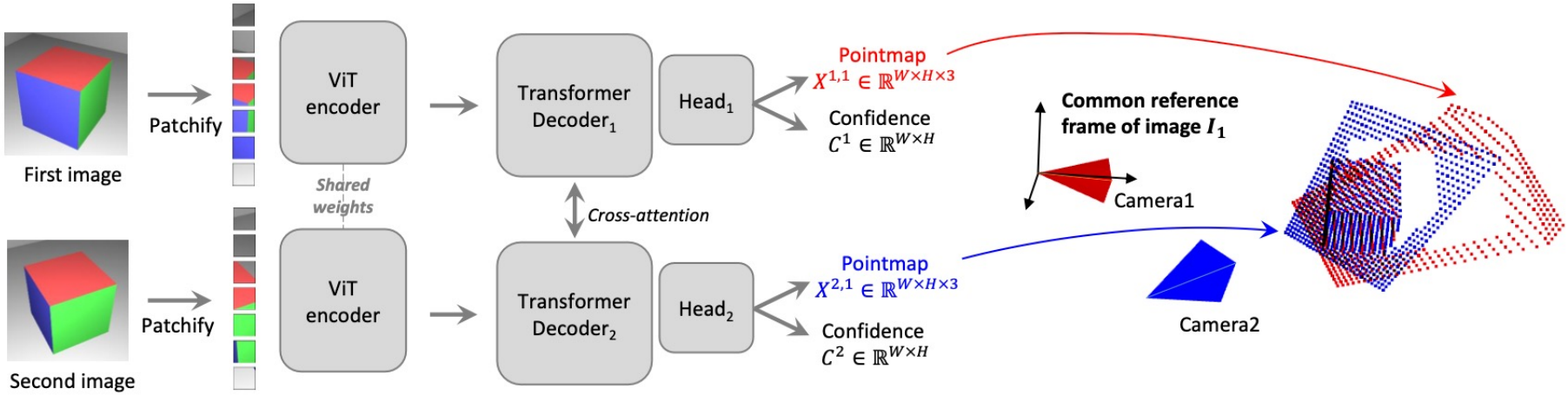


It reuses the CroCo backbone to benefit from the pre-training

BUT …

DUst3r

DUst3r is a transformer based Point map regression network which can estimate the 3D coordinates of each pixel

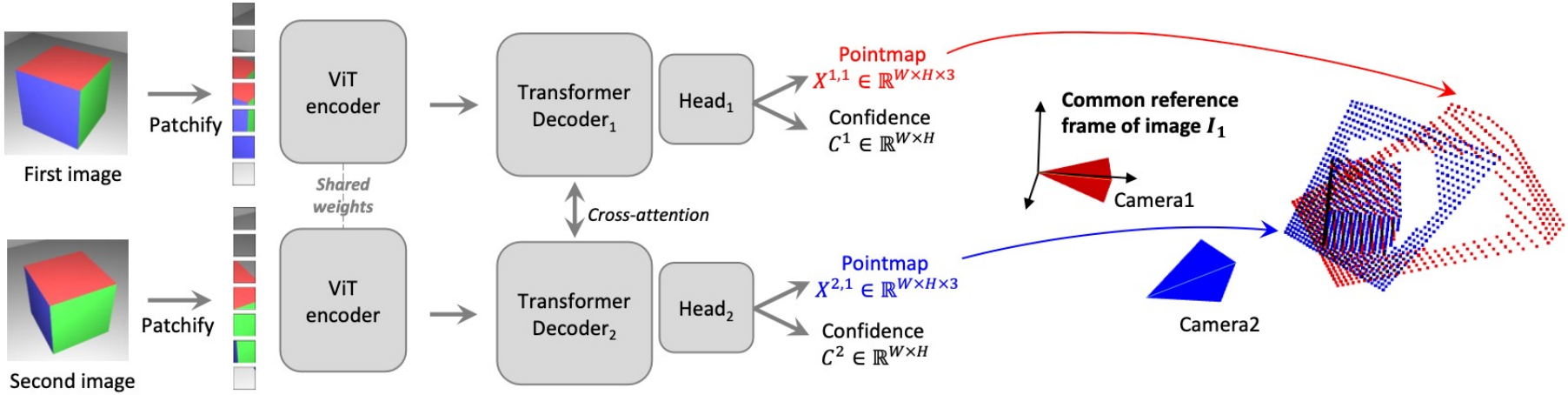


It reuses the CroCo backbone to benefit from the pre-training

BUT with another decoder to regress, the 3D point coordinates for each image in the pair!

DUst3r

DUst3r is a transformer based Point map regression network which can estimate the 3D coordinates of each pixel



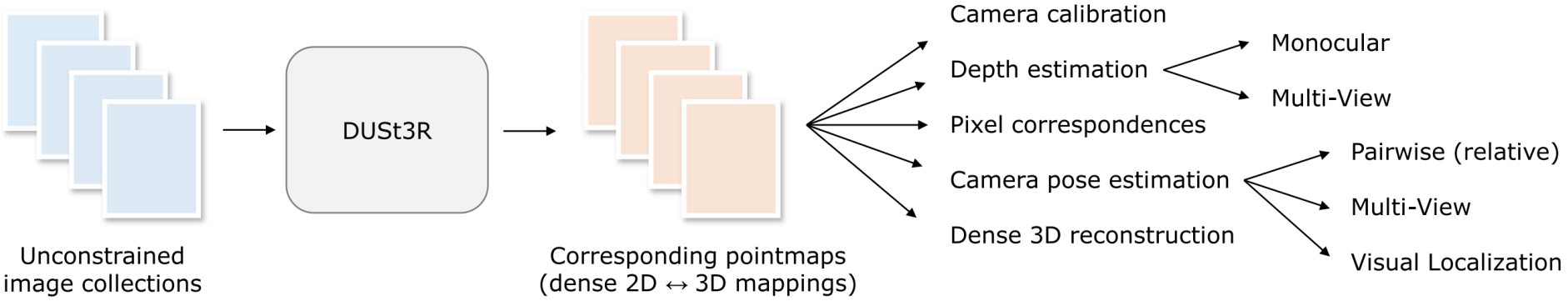
Trained in a fully supervised manner using known depth maps and poses

Datasets	Type	N Pairs
Habitat [103]	Indoor / Synthetic	1000k
CO3Dv2 [93]	Object-centric	941k
ScanNet++ [165]	Indoor / Real	224k
ArkitScenes [25]	Indoor / Real	2040k
Static Thing 3D [68]	Object / Synthetic	337k
MegaDepth [55]	Outdoor / Real	1761k
BlendedMVS [161]	Outdoor / Synthetic	1062k
Waymo [121]	Outdoor / Real	1100k

DUst3r

How is this model a unified models?

What tasks can it resolves??



Solving the map point regression problem is equivalent to resolving all these tasks!

DUst3r

DUSt3R: Geometric 3D Vision Made Easy

S. Wang [1], V. Leroy [2], Y. Cabon [2], B. Chidlovskii [2] and J. Revaud [2]

[1] Aalto University

2 Naver Labs Europe

DUst3r

DUst3R is working very well on many tasks, it is extremely robust!

It works under extreme conditions, with uncalibrated images

But …

It lacks accuracy for certain tasks

	Methods	GT cams	Acc.↓	Comp.↓	Overall↓
Handcrafted (a)	Camp [11]	✓	0.835	0.554	0.695
	Furu [32]	✓	0.613	0.941	0.777
	Tola [100]	✓	0.342	1.190	0.766
	Gipuma [33]	✓	**0.283**	0.873	0.578
Learning Based (b)	MVSNet [121]	✓	0.396	0.527	0.462
	CVP-MVSNet [119]	✓	0.296	0.406	0.351
	UCS-Net [16]	✓	0.338	0.349	0.344
	CER-MVS [55]	✓	0.359	0.305	0.332
	CIDER [118]	✓	0.417	0.437	0.427
	PatchmatchNet [103]	✓	0.427	0.277	0.352
	GeoMVSNet [136]	✓	0.331	**0.259**	**0.295**
	DUSt3R 512	×	2.677	0.805	1.741

MVS benchmark on DTU


Mast3R

MASt3R

MASt3R stand for "Matching And Stereo 3D Reconstruction"
But the paper is entitled

Grounding Image Matching in 3D with MASt3R

Vincent Leroy
Naverlabs Europe

Yohann Cabon
Naverlabs Europe

Jérome Revaud
Naverlabs Europe

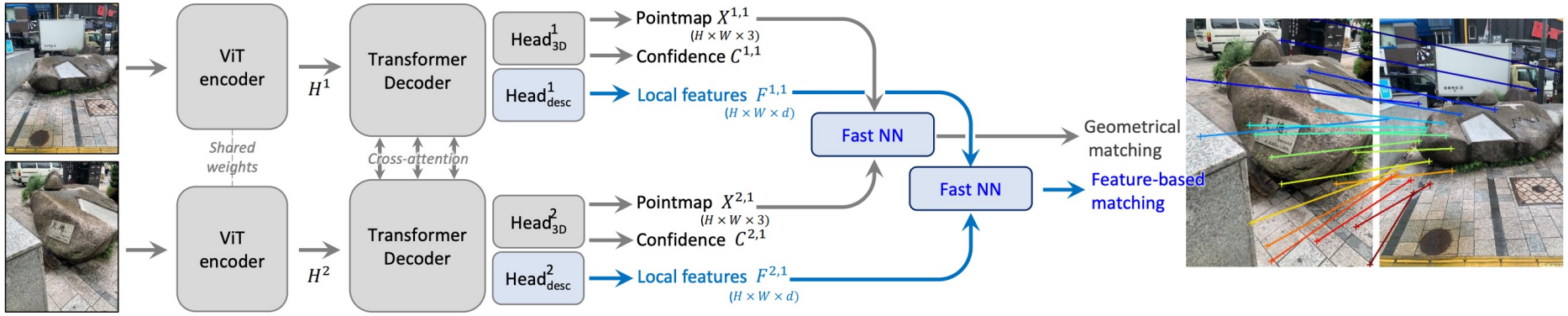
NAVER LABS Europe

ECCV Milan 2024

MASt3R

MASt3R is very similar to Dust3r but with an additional loss specifically designed to enforce proper correspondences

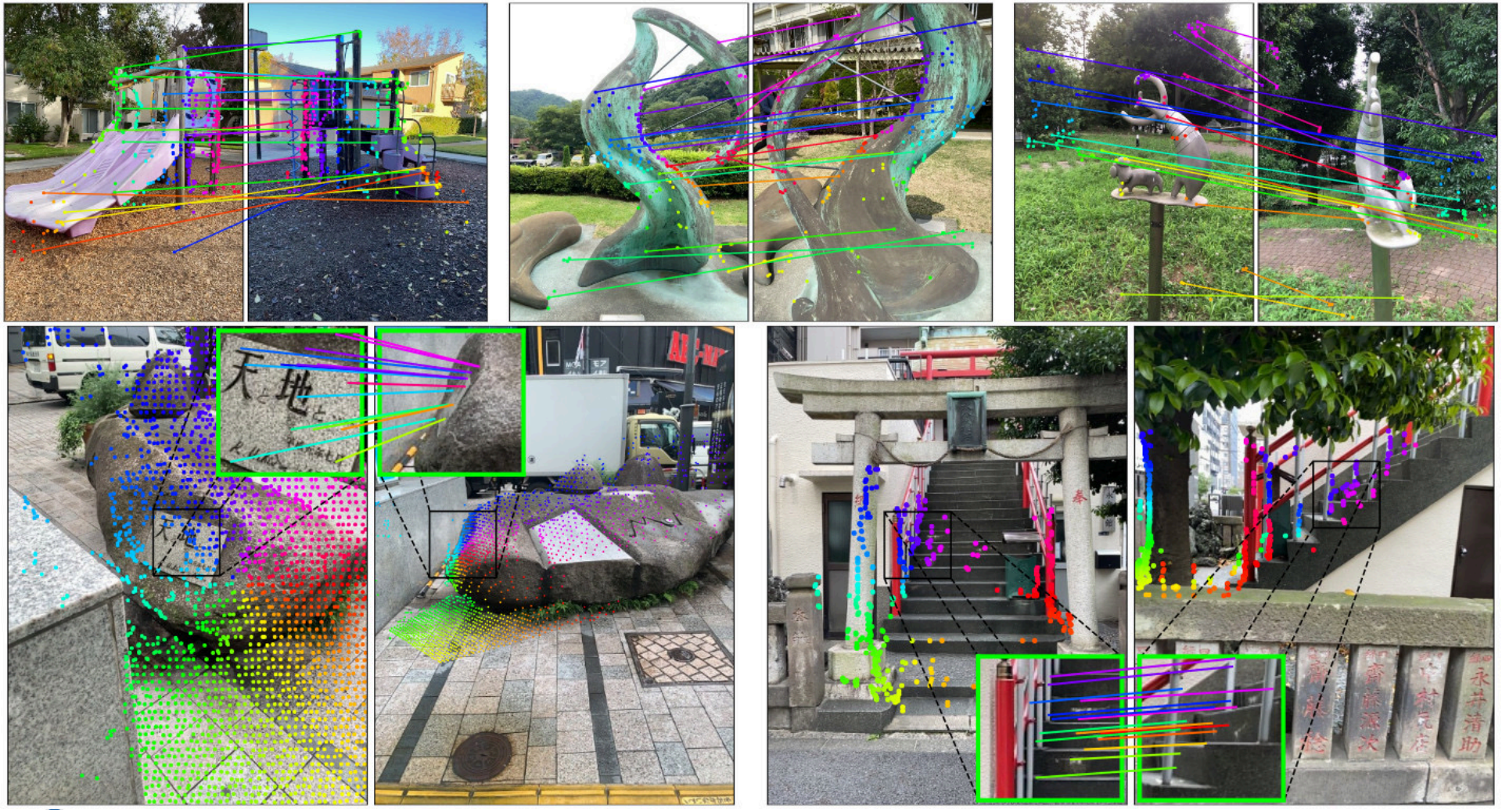
(And also, it is now metric!)



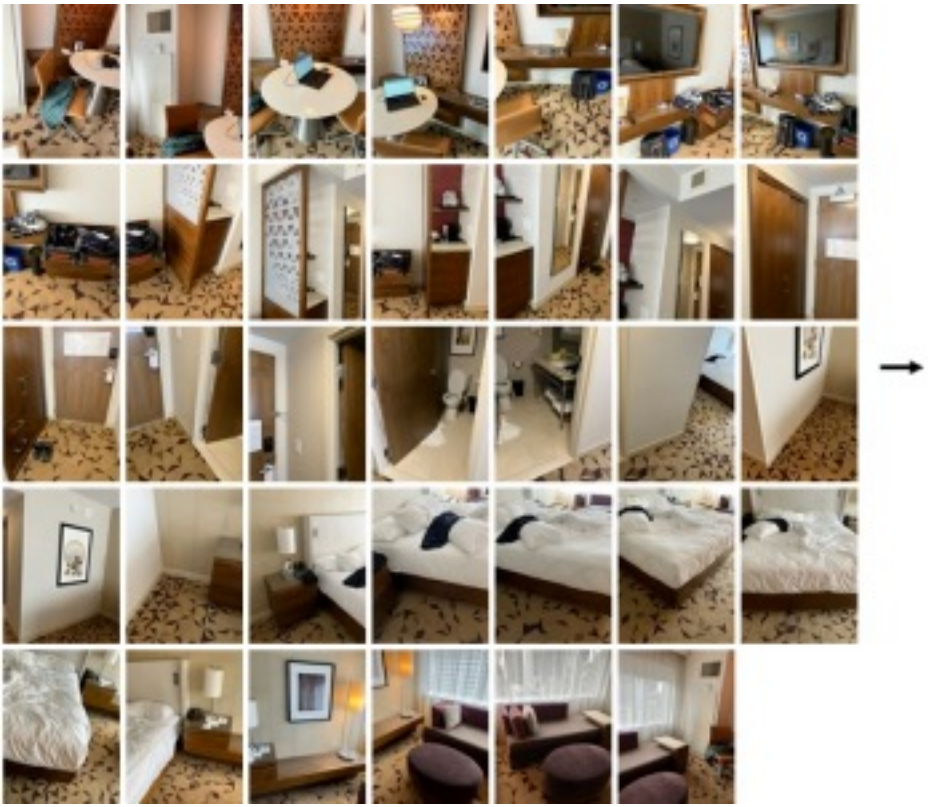
The local features are trained with Contrastive Learning

MASt3R: Results

Some qualitative Results



MASt3R: Results



MASt3R: Results

Mast3r is top-ranked in many benchmarks!

Table 2: Comparison with the state of the art on the *test* set of the Map-free dataset.

	depth	VCRE (<90px) Reproj. ↓	Prec. ↑	AUC ↑	Pose Error Med. Err. (m,°) ↓		Precision ↑	AUC ↑
RPR [5]	DPT	147.1 px	40.2%	0.402	1.68m	22.5°	6.0%	0.060
SIFT [52]	DPT	222.8 px	25.0%	0.504	2.93m	61.4°	10.3%	0.252
SP+SG [72]	DPT	160.3 px	36.1%	0.602	1.88m	25.4°	16.8%	0.346
LoFTR [82]	KBR	165.0 px	34.3%	0.634	2.23m	37.8°	11.0%	0.295
DUSt3R [102]	DPT	116.0 px	50.3%	0.697	0.97m	7.1°	21.6%	0.394
MASt3R	DPT	104.0 px	54.2%	0.726	0.80m	**2.2°**	27.0%	0.456
MASt3R	(auto)	**48.7** px	**79.3%**	0.933	**0.36**m	**2.2°**	**54.7%**	0.740
MASt3R (direct reg.)		53.2 px	79.1%	**0.941**	0.42m	3.1°	53.0%	**0.777**

	Methods	Co3Dv2 RRA@15	RTA@15	mAA(30)	RealEstate10K mAA(30)
(a)	Colmap+SG [21,72]	36.1	27.3	25.3	45.2
	PixSfM [50]	33.7	32.9	30.1	49.4
	RelPose [115]	57.1	-	-	-
	PosReg [100]	53.2	49.1	45.0	-
	PoseDiff [100]	80.5	79.8	66.5	48.0
	RelPose++ [49]	(85.5)	-	-	-
	RayDiff [116]	(93.3)	-	-	-
	DUSt3R-GA [102]	**96.2**	86.8	76.7	67.7
(b)	DUSt3R [102]	94.3	88.4	77.2	61.2
	MASt3R	94.6	**91.9**	**81.8**	**76.4**

	Methods	Acc.↓	Comp.↓	Overall↓
(c)	Camp [13]	0.835	0.554	0.695
	Furu [31]	0.613	0.941	0.777
	Tola [90]	0.342	1.190	0.766
	Gipuma [32]	**0.283**	0.873	0.578
(d)	MVSNet [110]	0.396	0.527	0.462
	CVP-MVSNet [109]	0.296	0.406	0.351
	UCS-Net [17]	0.338	0.349	0.344
	CER-MVS [55]	0.359	0.305	0.332
	CIDER [107]	0.417	0.437	0.427
	PatchmatchNet [99]	0.427	0.277	0.352
	GeoMVSNet [119]	0.331	**0.259**	**0.295**
(e)	DUSt3R [102]	2.677	0.805	1.741
	MASt3R	0.403	0.344	0.374

Table 4: Visual localization results on Aachen Day-Night and InLoc. We report our results for different number of retrieved database images (topN).

Conclusion

Conclusion

Which techniques are the best?

It depends on what you want to achieve!

	PRO	CONS
One-Shot (DUSt3R, MASt3R)	• Fast and Effective • Uncalibrated images (no pose, no intrinsics) • Outputs dense 2D-3D points	• Coarser reconstruction (compared to per-scene learning) • Generalization issues • Computationally intensive
SfM + MVS	• Mature & well-understood • Explicit and interpretable outputs • Large-scale scenes	• Need calibrated cameras • Slow • Problem with texture-less environments
Per-Scene (Nerf, GS)	• Photorealistic fidelity • Compact Representation • Applicable for many tasks	• Requires SfM priors • One training per scene

Conclusion

Some work now combine the best of both world like . . .

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

Zhiwen Fan[1,2*†], Kairun Wen[3*], Wenyan Cong[1*], 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],
[1] The University of Texas at Austin [2] Nvidia [3] Xiamen University
[4] Georgia Institute of Technology [5] Stanford University [6] University of Southern California

SUNY Korea

Conclusion

The field is moving very fast, here is a novel Mast3r-based SLAM

MASt3R-SLAM

Real-Time Dense SLAM with 3D Reconstruction Priors

Riku Murai*, Eric Dexheimer*, Andrew J. Davison
Imperial College London
*Authors contributed equally to this work

Conclusion

What do you think will come next?

What are the remaining challenges to be tackled? (Real-time, Dynamic Scene, Light Representation, Illumination changes, etc…)