Embodied Vision

4D Modeling

Tsung-Wei Ke

Spring 2026

Disclaimer

- This lecture borrows contents partly from
  - Computer Vision by Svetlana Lazebnik at UIUC
  - Computer Vision by Kris Kitani at CMU

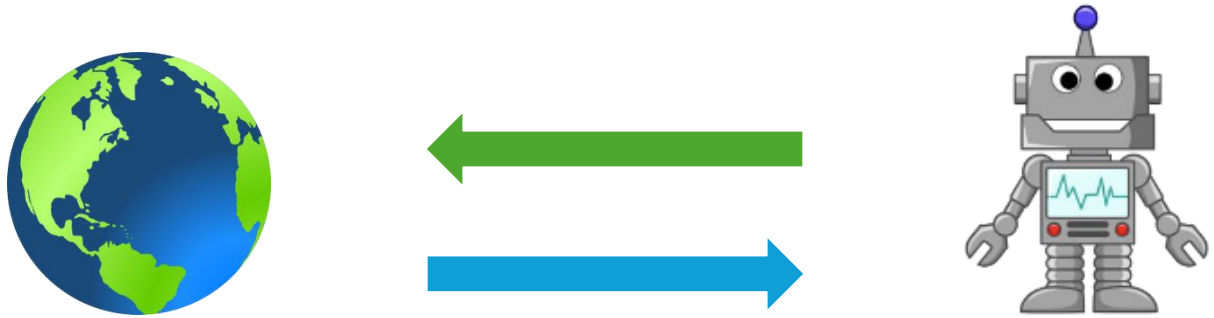
We Live in a Dynamic World

https://i.gifer.com/fxxv.gif

https://ca.sports.yahoo.com/news/mvp-frontrunners-vladimir-guerrero-jr-shohei-ohtani-head-to-head-142258898.html

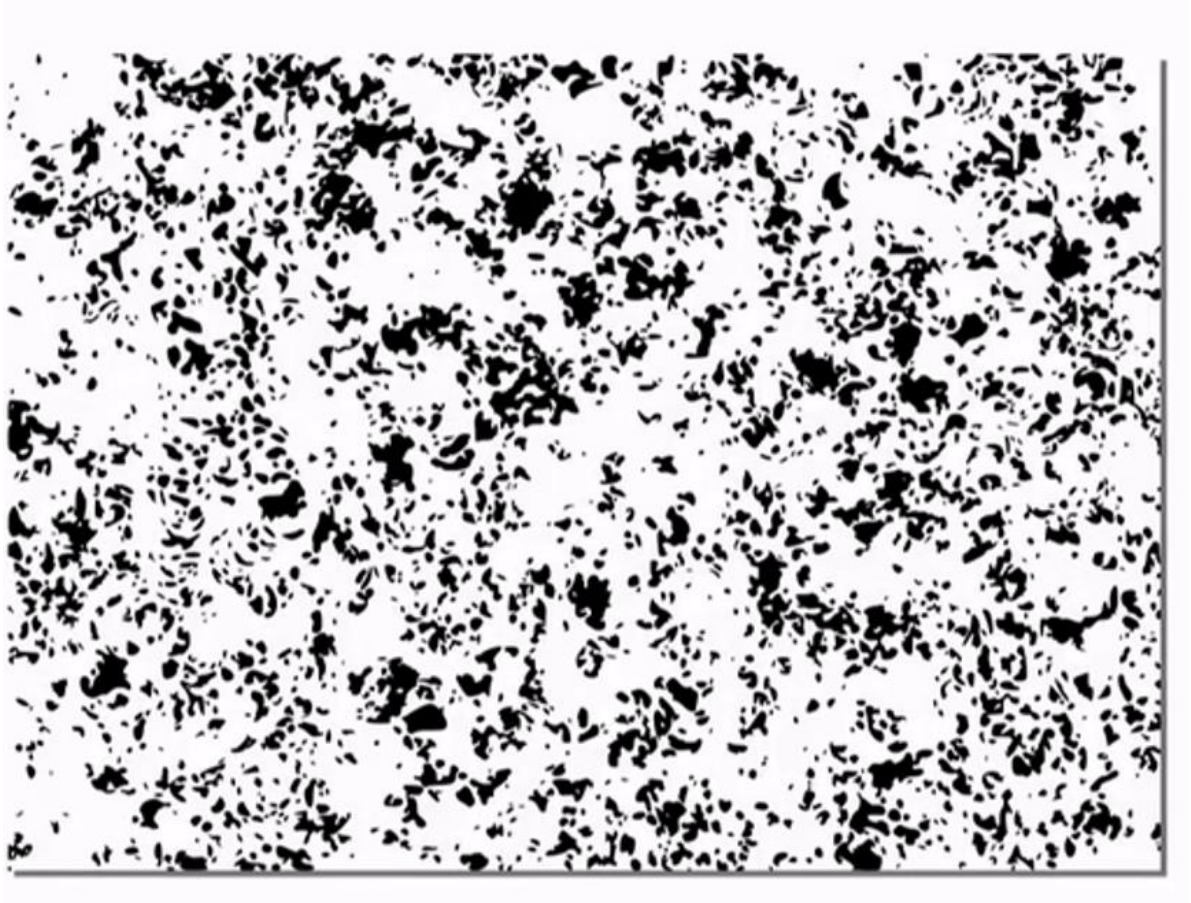
We Live in a Dynamic World

Action:
- What is the right action

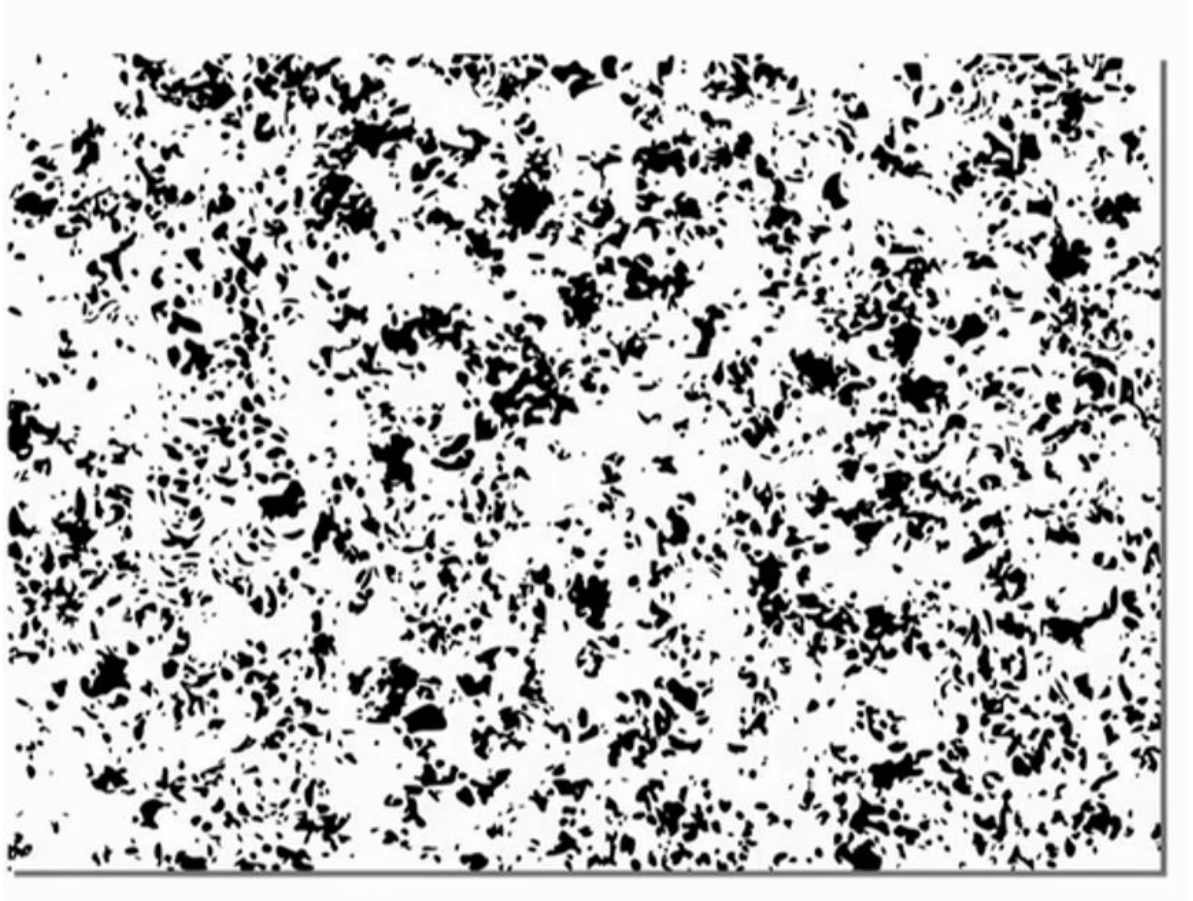


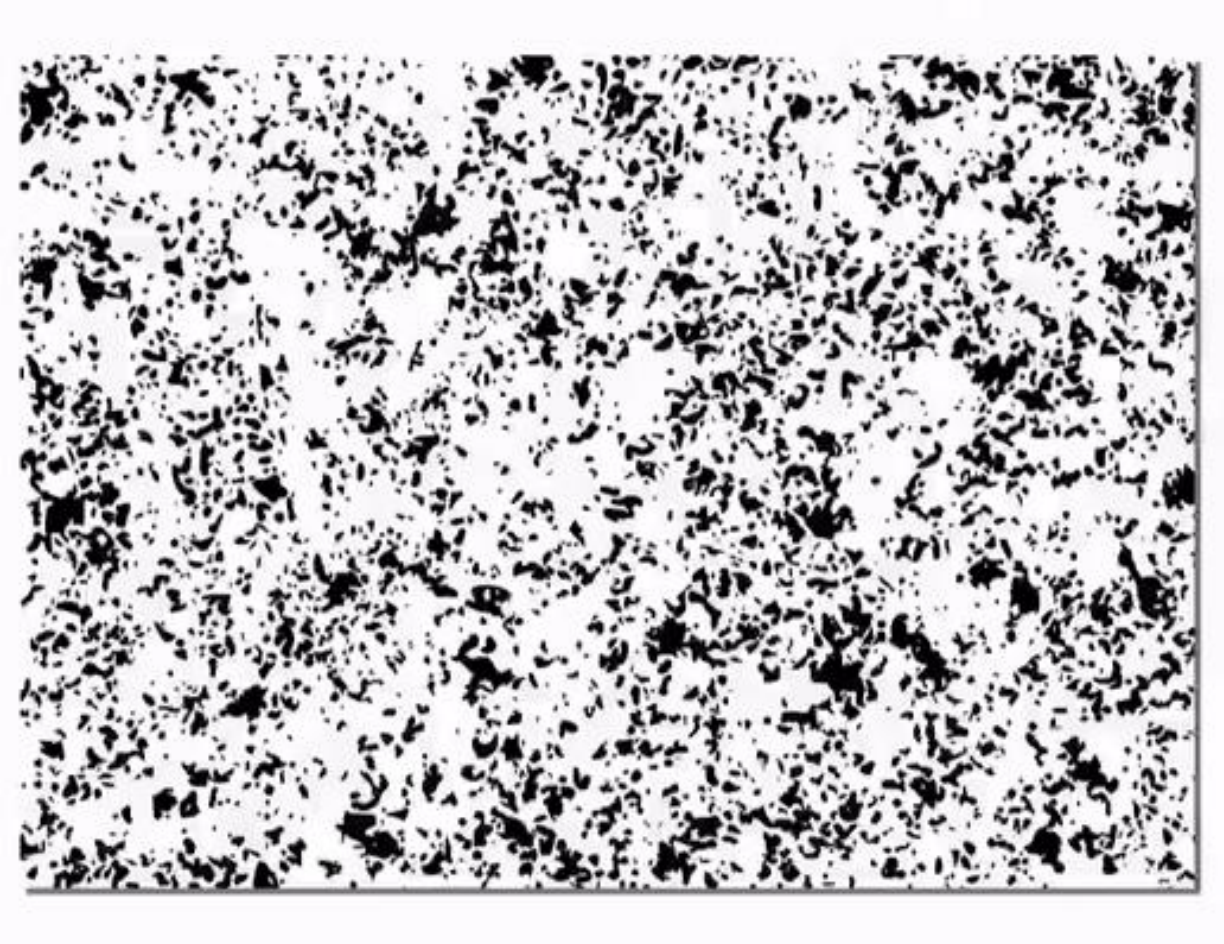
Perception:
- What and where are the objects
- How do objects move

Motion Perception Does Not Rely on Object Recognition



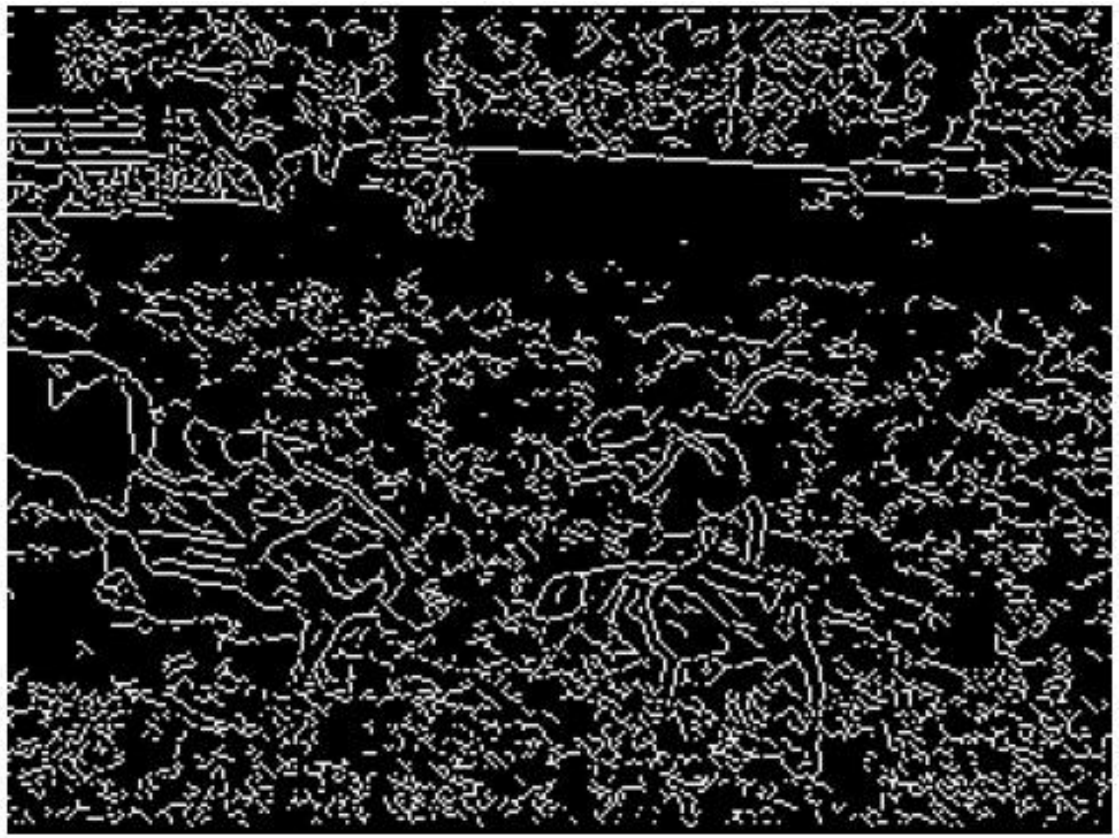
Motion Perception Does Not Rely on Object Recognition

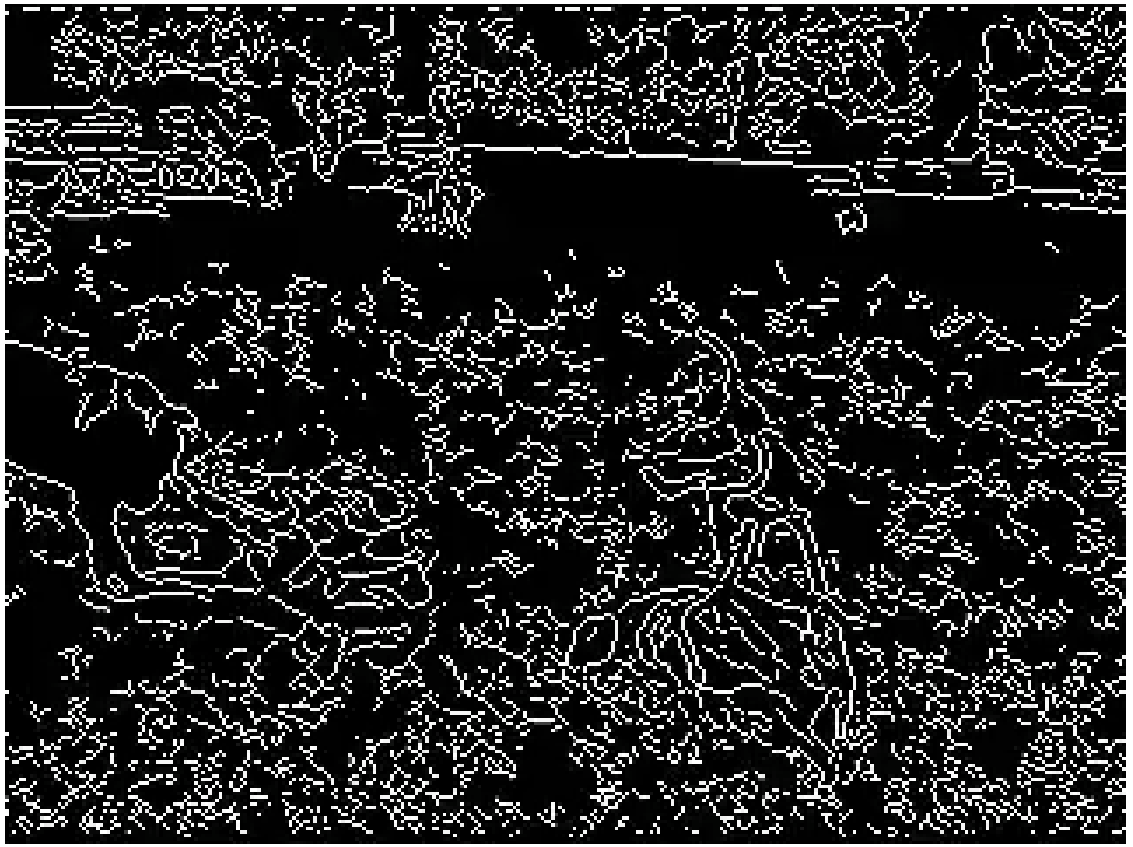




http://graphics.stanford.edu/~niloy/research/emergence/emergence_image_siga_09.html

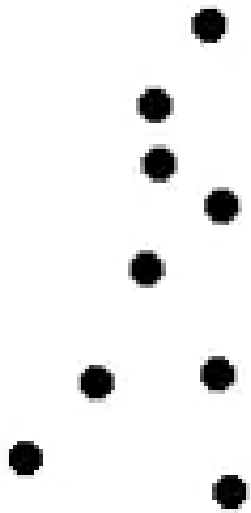
Motion Perception Does Not Rely on Object Recognition

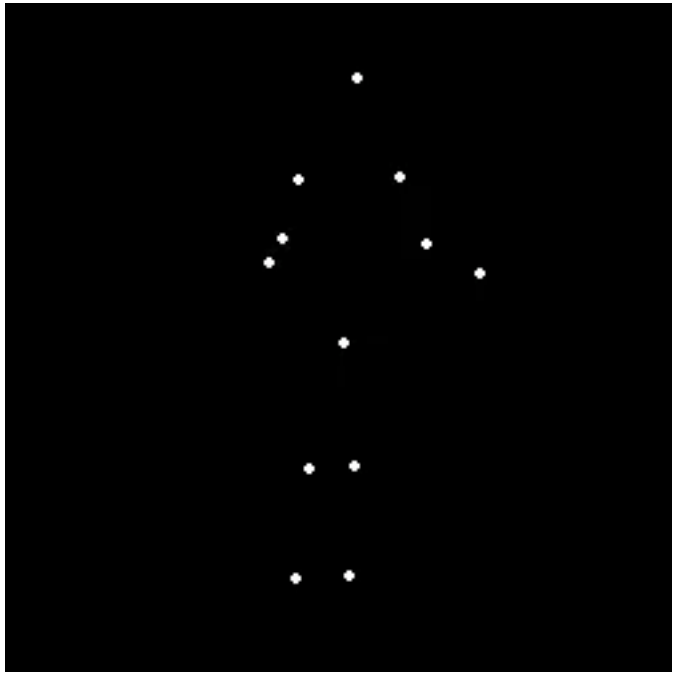




https://www.cns.nyu.edu/~david/courses/perception/lecturenotes/motion/motion.html

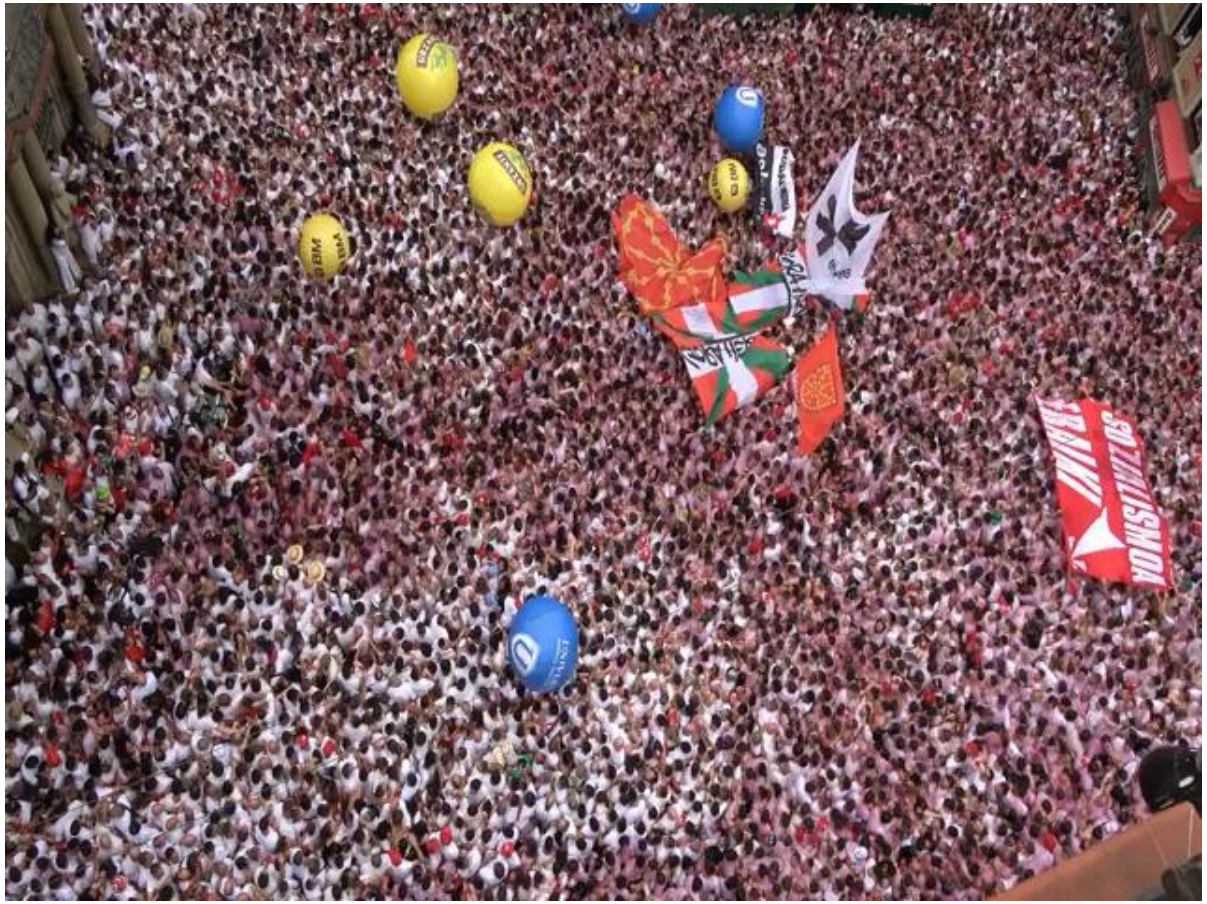
Motion Perception Does Not Rely on Object Recognition

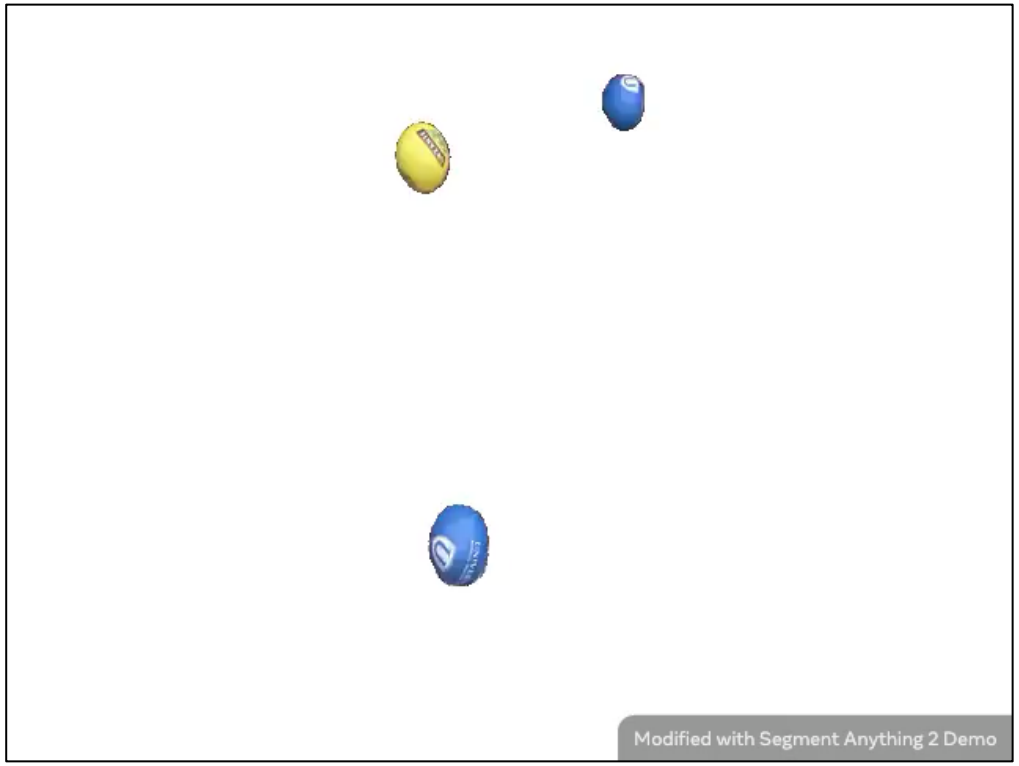




https://www.cns.nyu.edu/~david/courses/perception/lecturenotes/motion/motion.html

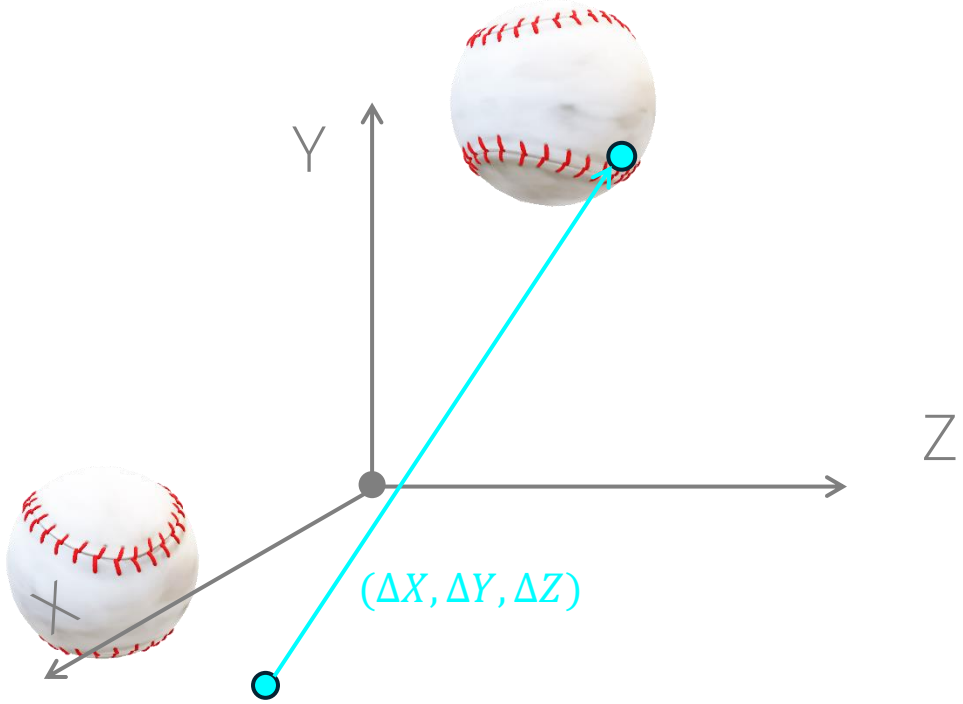
How to Represent Motion?

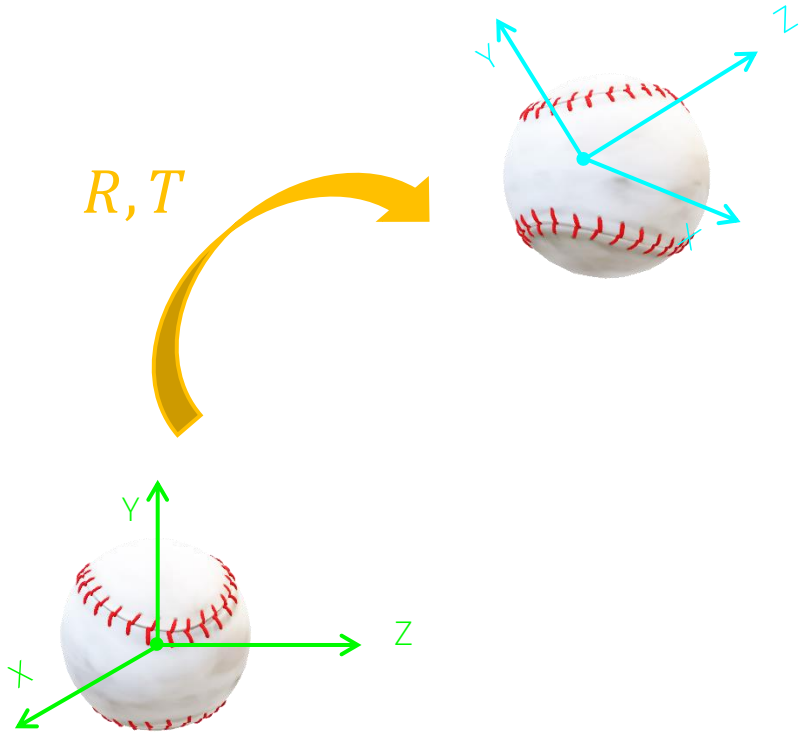




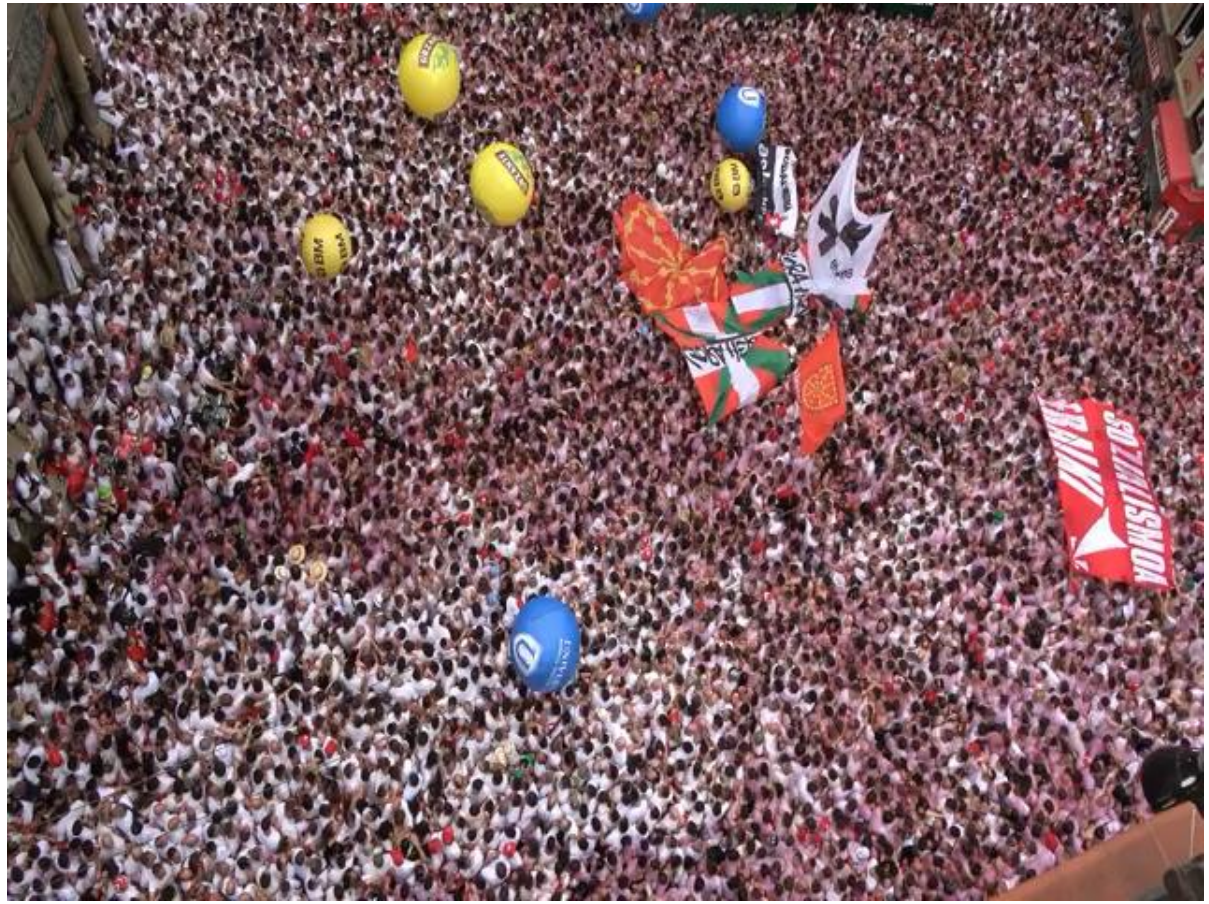
Video source: Emergence of collective oscillations in massive human crowds. Gu et al.

Rigid-Body Motion as a Motion Representation



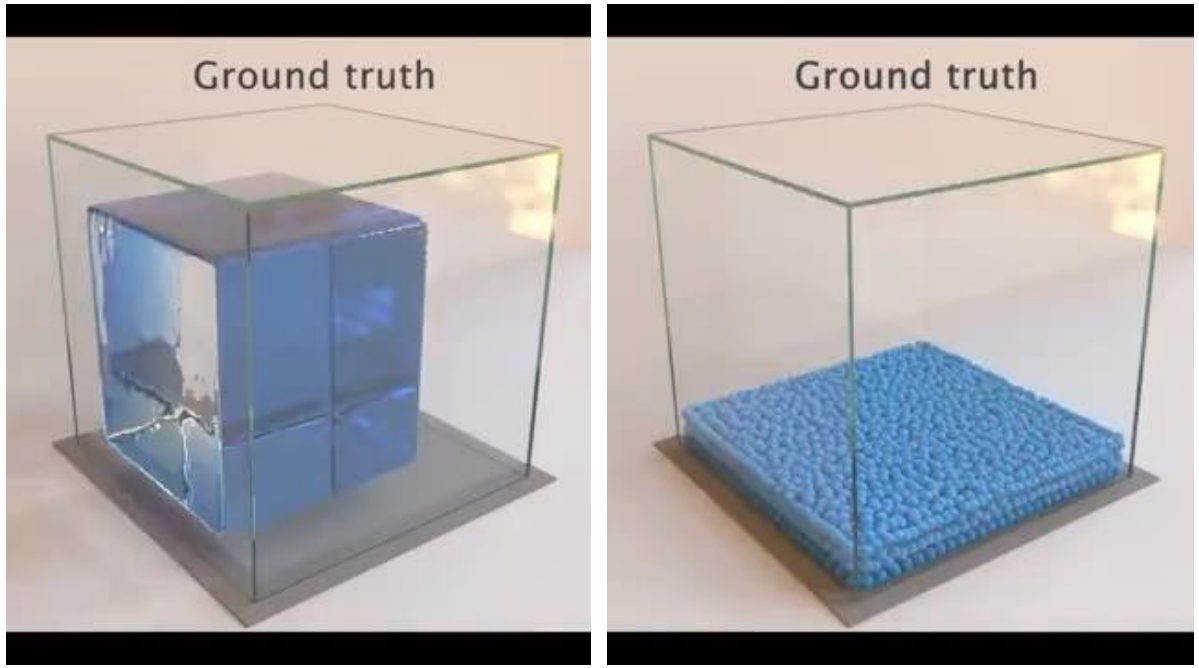


How to Represent Motion?



Video source: Emergence of collective oscillations in massive human crowds. Gu et al.

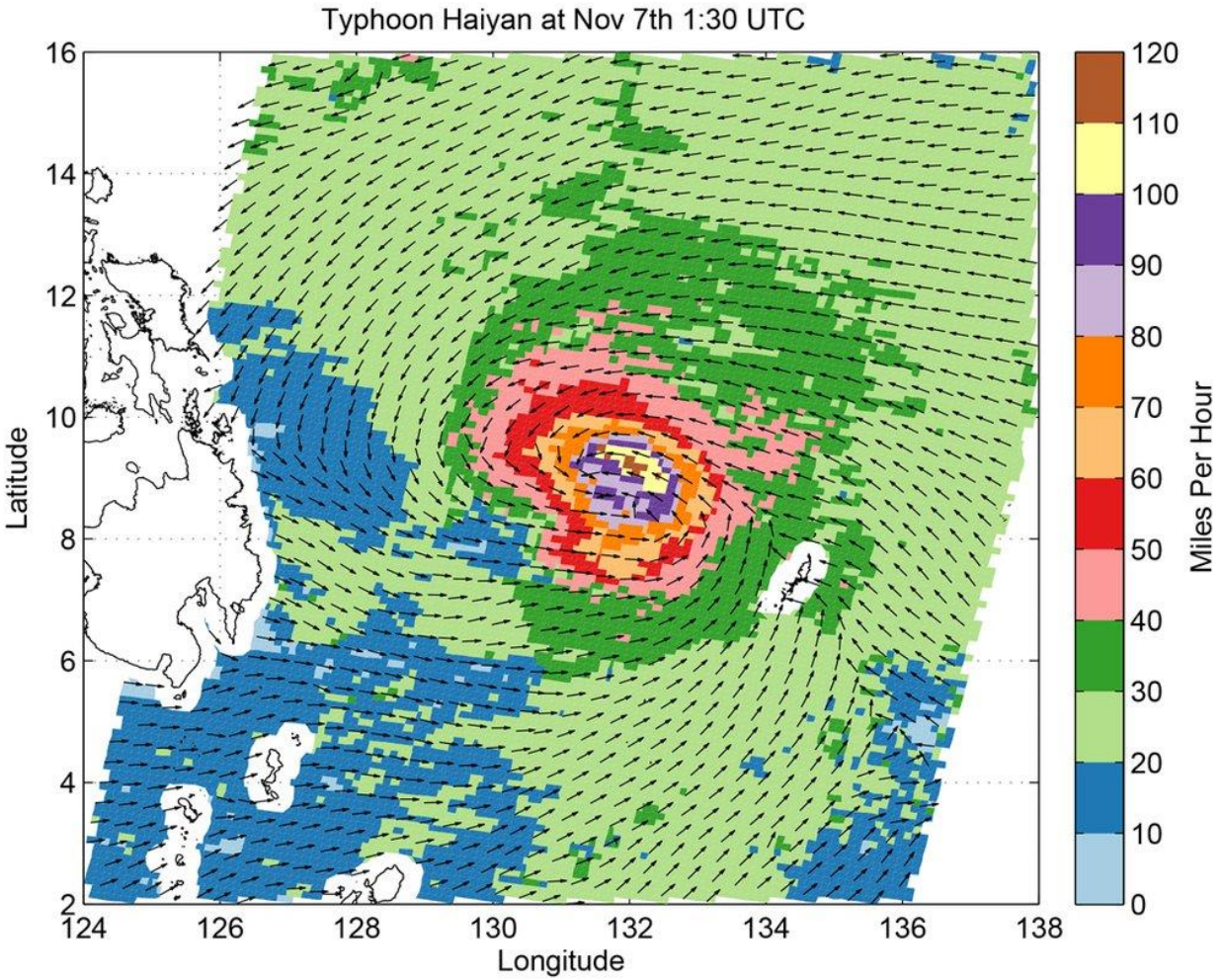
Particle Motion as a Motion Representation



Learning to simulate complex physics with graph networks. Google Deepmind.

Dynamic 3D Gaussians: Tracking by Persistent Dynamic View Synthesis. Luiten et al. 3DV 2024.

Vector Field as a Motion Representation

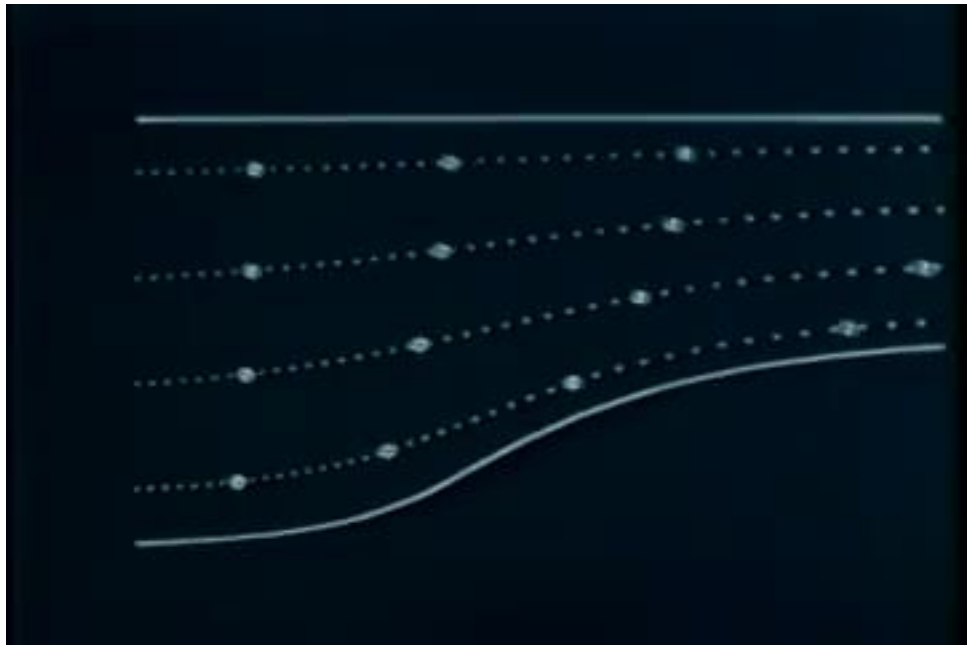


https://www.jpl.nasa.gov/news/nasa-peers-into-one-of-earths-strongest-storms-ever

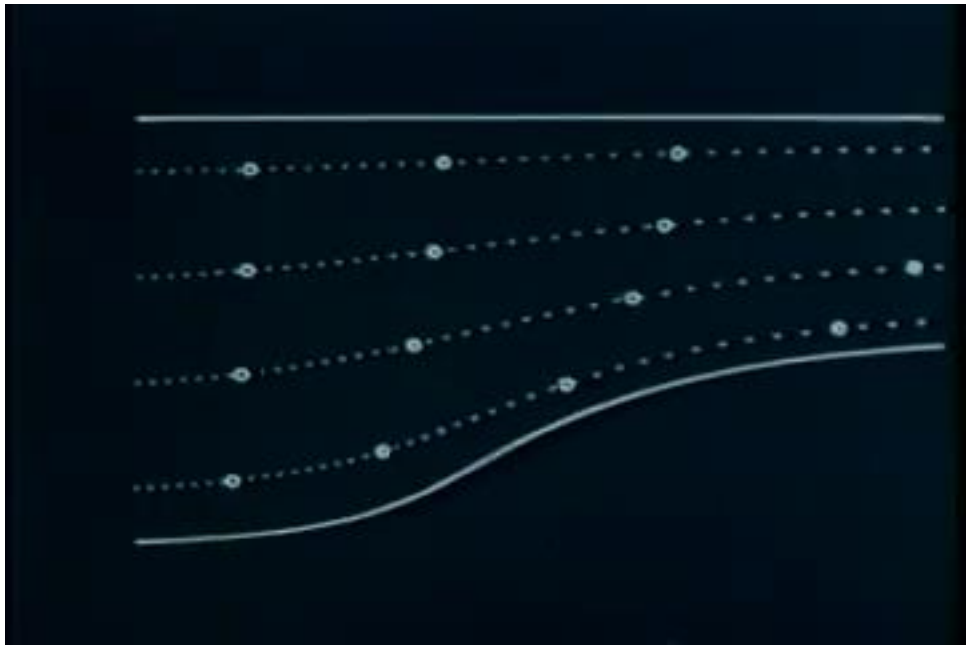
https://youtu.be/mdN8OOkx2ko?si=DPRdwEUYnObugwKl

Particle Motion vs. Vector Field Describe the Same Motion from Different Coordinate Systems

Particle motion
(Lagrangian representations)



Vector field
(Eulerian representations)



- We will revisit for physical modeling non-rigid bodies

https://youtu.be/mdN8OOkx2ko?si=DPRdwEUYnObugwKl

Content

- 2D Motion Representations
- 3D Motion Representations
  - ➢4D Volume Rendering with NERF
  - ➢Explicit Motion Field with NERF
  - ➢4D Volume Rendering with 3DGS
  - ➢Explicit Motion Field with 3DGS
  - ➢Implicit Motion Field with 3DGS
  - ➢4D Foundation Models
  - ➢Multi-view Video Generation (we'll cover in the next lecture)

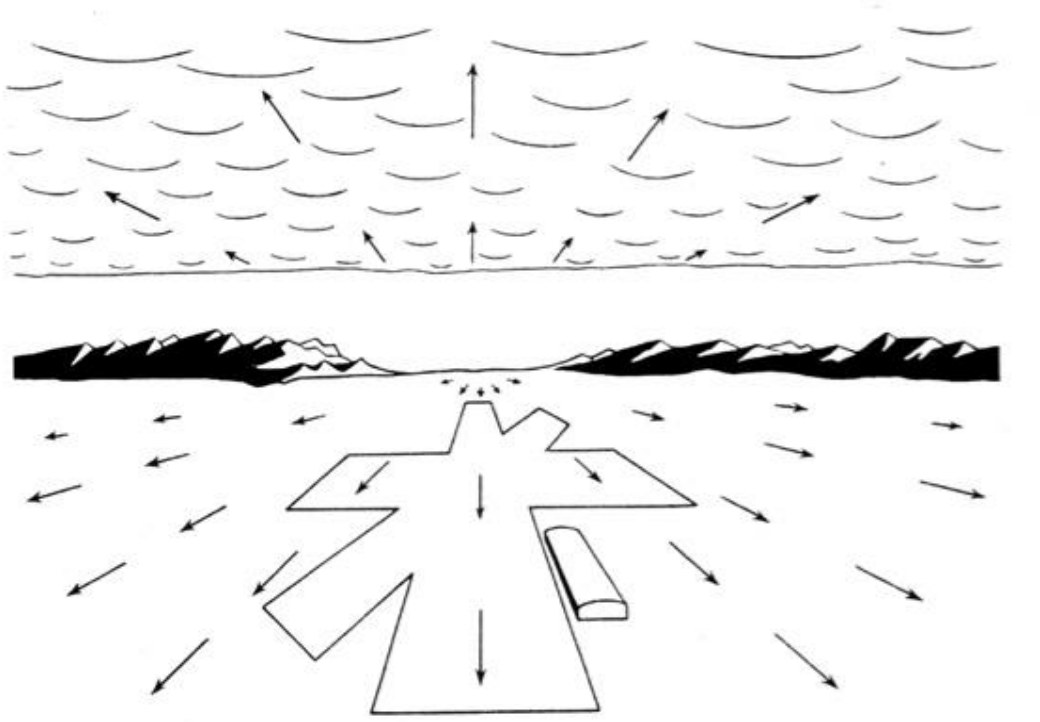
Content

- 2D Motion Representations
- 3D Motion Representations
  - ➢4D Volume Rendering with NERF
  - ➢Explicit Motion Field with NERF
  - ➢4D Volume Rendering with 3DGS
  - ➢Explicit Motion Field with 3DGS
  - ➢Implicit Motion Field with 3DGS
  - ➢4D Foundation Models
  - ➢Multi-view Video Generation (we'll cover in the next lecture)

Measure Motion from 2D Images

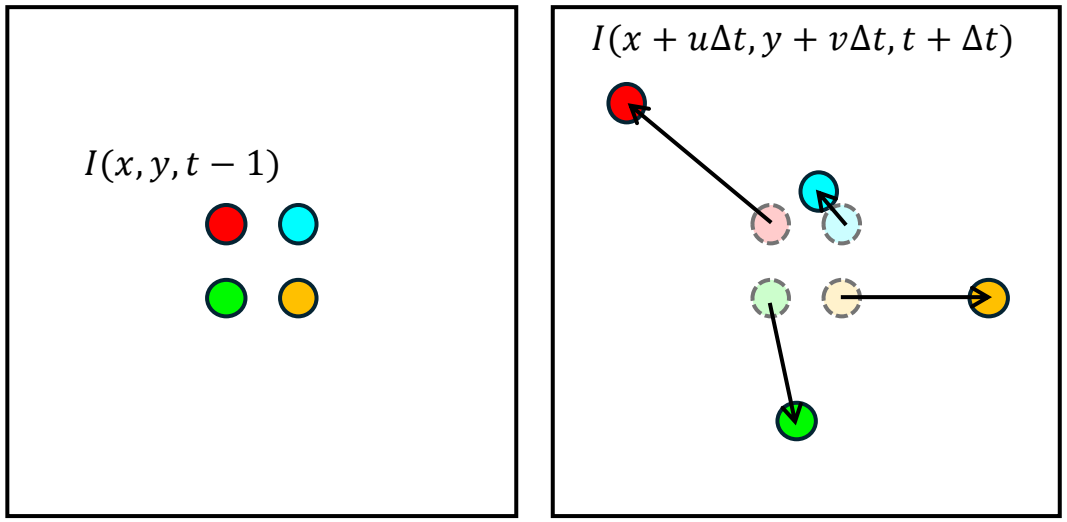
For 2D images, we don't know the exact 3D motions. We can only measure the pixel apparent motion

https://in.pinterest.com/pin/594686325771980172/



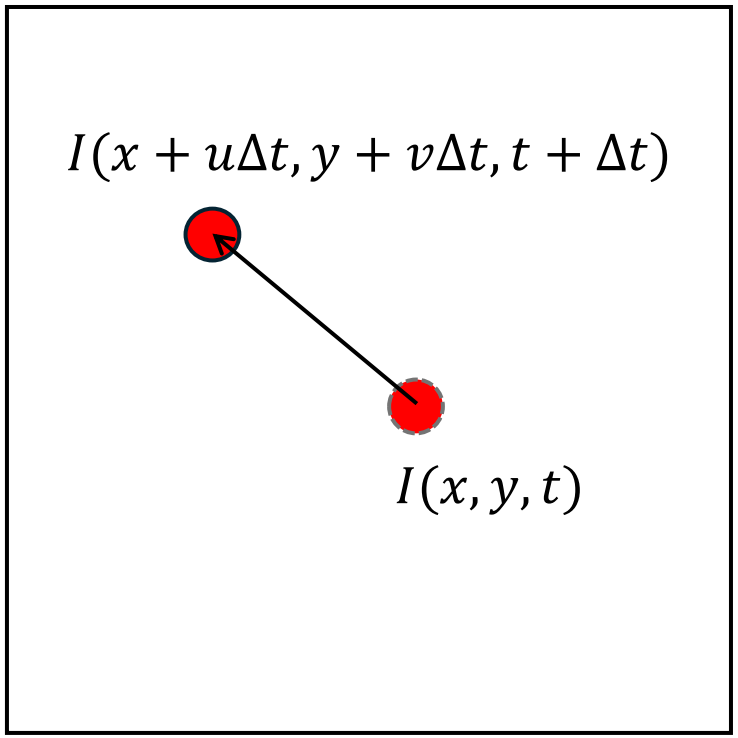
https://www.cns.nyu.edu/~david/courses/perception/lecturenotes/motion/motion.html

Optical Flow: Pixel Apparent Motion



- Given frames at times t and $t+\Delta t$, estimate the apparent motion field that describes the velocity $u(x, y, t)$ and $\mathbf{v}(x, y, t)$ of a pixel at location x, y and time $t+\Delta t$.
- Problem: how can we estimate the velocity field $u(x, y, t)$ and $\mathbf{v}(x, y, t)$ by observing a sequence of images?

Brightness Constancy Constraints



- **Brightness Constancy Constraints**: the brightness of a particular point remains constant across time

$$I(x + u\Delta t, y + v\Delta t, t + \Delta t) \approx I(x, y, t)$$

- **Brightness Constancy Equation:**

$$\boxed{\frac{\mathrm{D}I}{Dt}} = 0 \Rightarrow \boxed{\frac{\partial I}{\partial t}} + \boxed{\frac{\partial I}{\partial x}}\frac{\partial x}{\partial t} + \boxed{\frac{\partial I}{\partial y}}\frac{\partial y}{\partial t} = 0$$

Total derivative Partial derivative

Derivation of Brightness Constancy Equation

- Multi-variable Taylor Expansion:

$$f(x,y) \approx f(a,b) + \frac{\partial f(a,b)}{\partial x}(x-a) + \frac{\partial f(a,b)}{\partial y}(y-b)$$

- Brightness Constancy Equation:

$$I(x+u\Delta t, y+v\Delta t, t+\Delta t) \approx I(x,y,t) + \frac{\partial I(x,y,t)}{\partial x}u\Delta t + \frac{\partial I(x,y,t)}{\partial y}v\Delta t + \frac{\partial I(x,y,t)}{\partial t}\Delta t$$

$$\Rightarrow \frac{\partial I(x,y,t)}{\partial x}u + \frac{\partial I(x,y,t)}{\partial y}v + \frac{\partial I(x,y,t)}{\partial t} = 0 \Rightarrow \frac{\partial I}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial I}{\partial y}\frac{\partial y}{\partial t} + \frac{\partial I}{\partial t} = 0$$

Take a Closer Look at the Equation

$$\frac{\partial I}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial I}{\partial y}\frac{\partial y}{\partial t} + \frac{\partial I}{\partial t} = 0$$

- Image gradients $\frac{\partial I}{\partial x}$ and $\frac{\partial I}{\partial y}$: spatial derivatives of image intensity, which can be estimated by forward difference / Sober filter / Scharr filter …
- Temporal gradients $\frac{\partial I}{\partial t}$: temporal derivatives of image intensity, which can be estimated by subtracting images at different timestep
- Optical flow $\frac{\partial x}{\partial t}$ and $\frac{\partial y}{\partial t}$: the unknown variables we want to estimate

Image Gradients and Temporal Gradients

t

1	1	1	1	1
1	1	1	1	1
1	10	10	10	10
1	10	10	10	10
1	10	10	10	10

$t+1$

1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	10	10	10	10
1	10	10	10	10

Image gradients $\frac{\partial I}{\partial x}$

t

1	1	1	1	1
1	1	1	1	1
1	10	10	10	10
1	10	10	10	10
1	10	10	10	10

$-$

t

1	1	1	1	1
1	1	1	1	1
1	10	10	10	10
1	10	10	10	10
1	10	10	10	10

$=$

-	0	0	0	-
-	0	0	0	-
-	9	0	0	-
-	9	0	0	-
-	0	0	0	-

Temporal gradients $\frac{\partial I}{\partial t}$

t

1	1	1	1	1
1	1	1	1	1
1	10	10	10	10
1	10	10	10	10
1	10	10	10	10

$-$

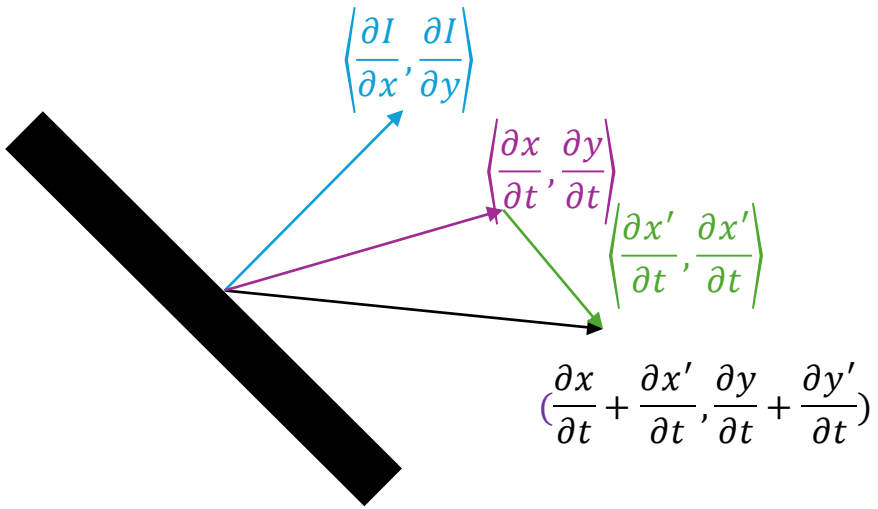
$t+1$

1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	10	10	10	10
1	10	10	10	10

$=$

0	0	0	0	0
0	0	0	0	0
0	9	9	9	9
0	0	0	0	0
0	0	0	0	0

Given Image and Temporal Gradient, Can We Recover Optical Flow Uniquely?



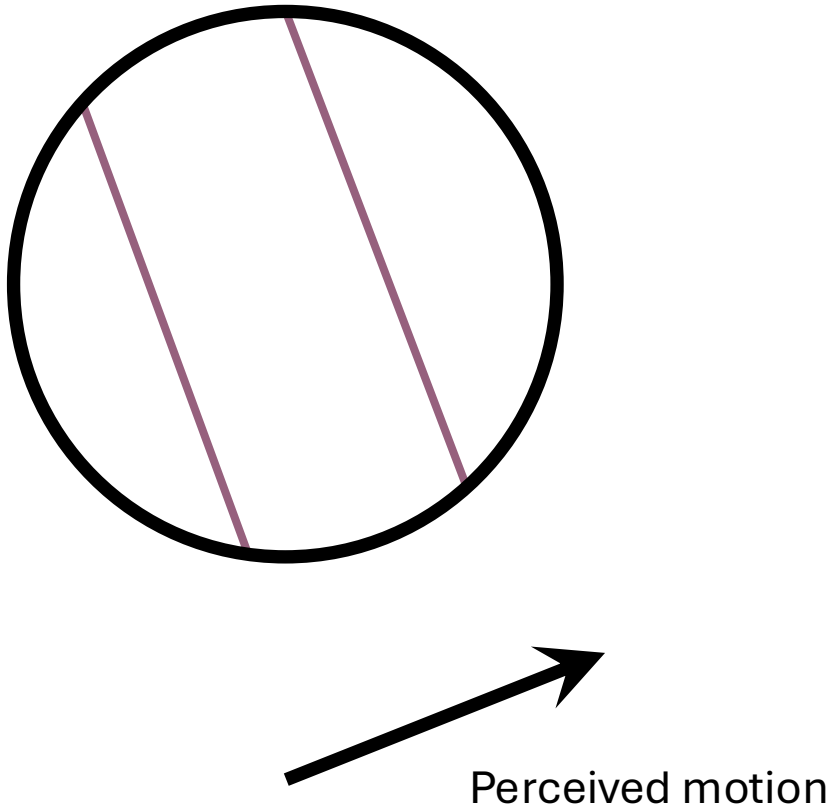
- Let's first write the brightness constancy constraint in vector form:

$$\left\langle \frac{\partial I}{\partial x}, \frac{\partial I}{\partial y} \right\rangle \cdot \left\langle \frac{\partial x}{\partial t}, \frac{\partial y}{\partial t} \right\rangle + \frac{\partial I}{\partial t} = 0$$

- We have another solution with $\frac{\partial x}{\partial t} + \frac{\partial x'}{\partial t}$ and $\frac{\partial y}{\partial t} + \frac{\partial y'}{\partial t}$ when $\left\langle \frac{\partial I}{\partial x}, \frac{\partial I}{\partial y} \right\rangle \cdot \left\langle \frac{\partial x'}{\partial t}, \frac{\partial x'}{\partial t} \right\rangle = 0$
- In other worlds, the component perpendicular to the gradient (i.e., parallel to the edge) cannot be recovered!

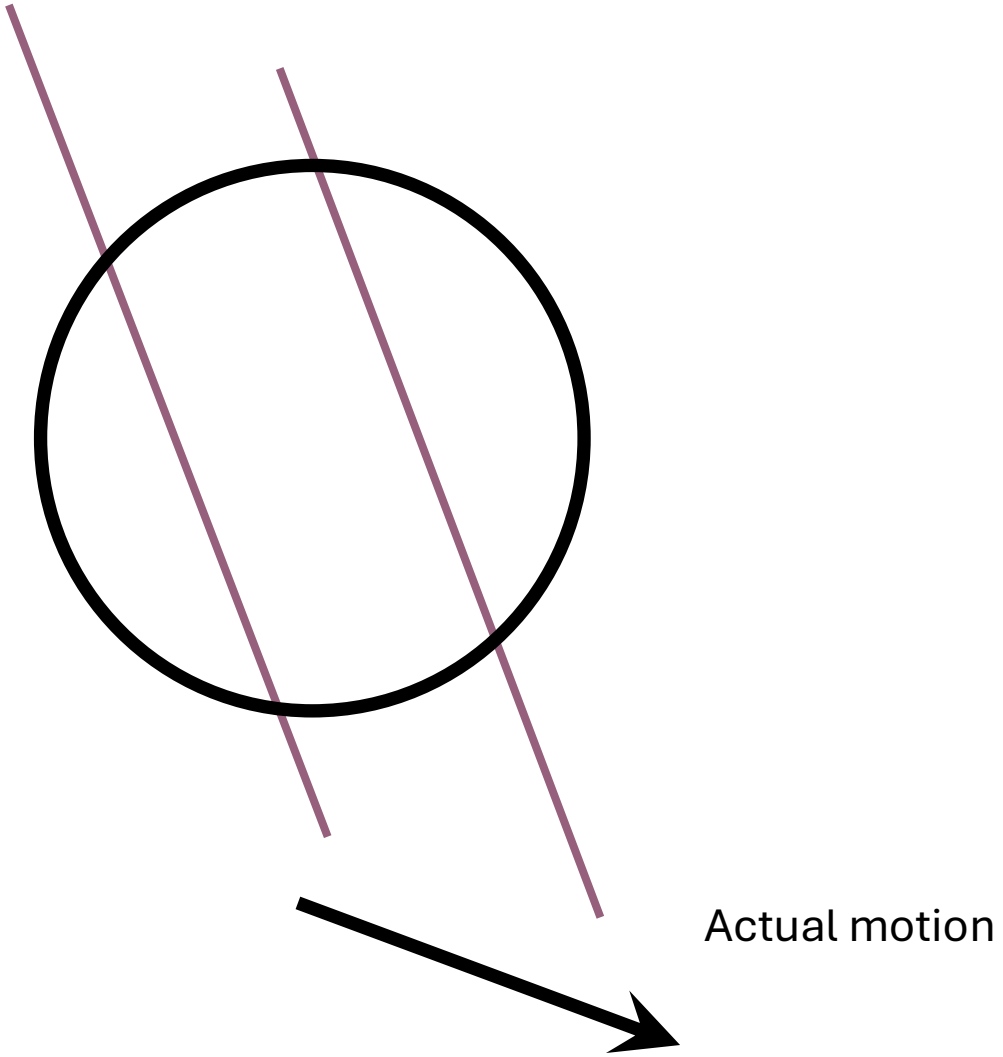
Slide adapted from S. Lazebnik

The Aperture Problem



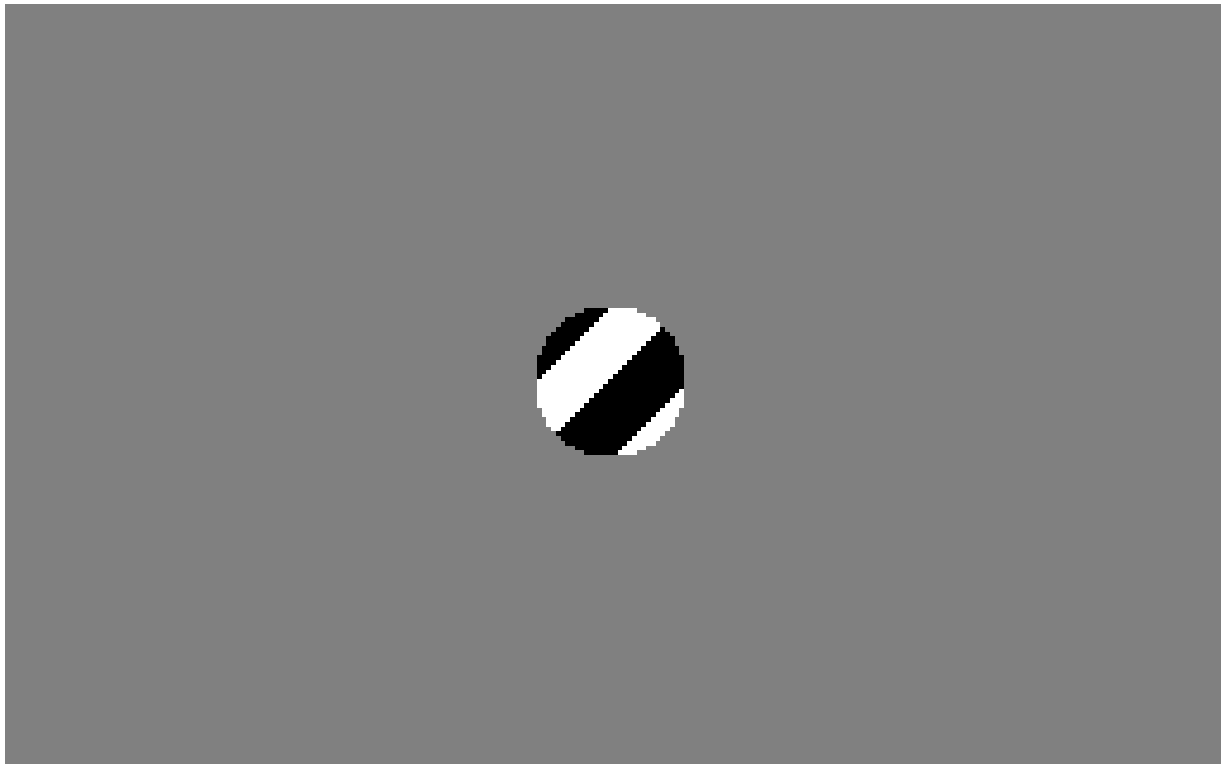
Slide credit S. Lazebnik

The Aperture Problem



Slide credit S. Lazebnik

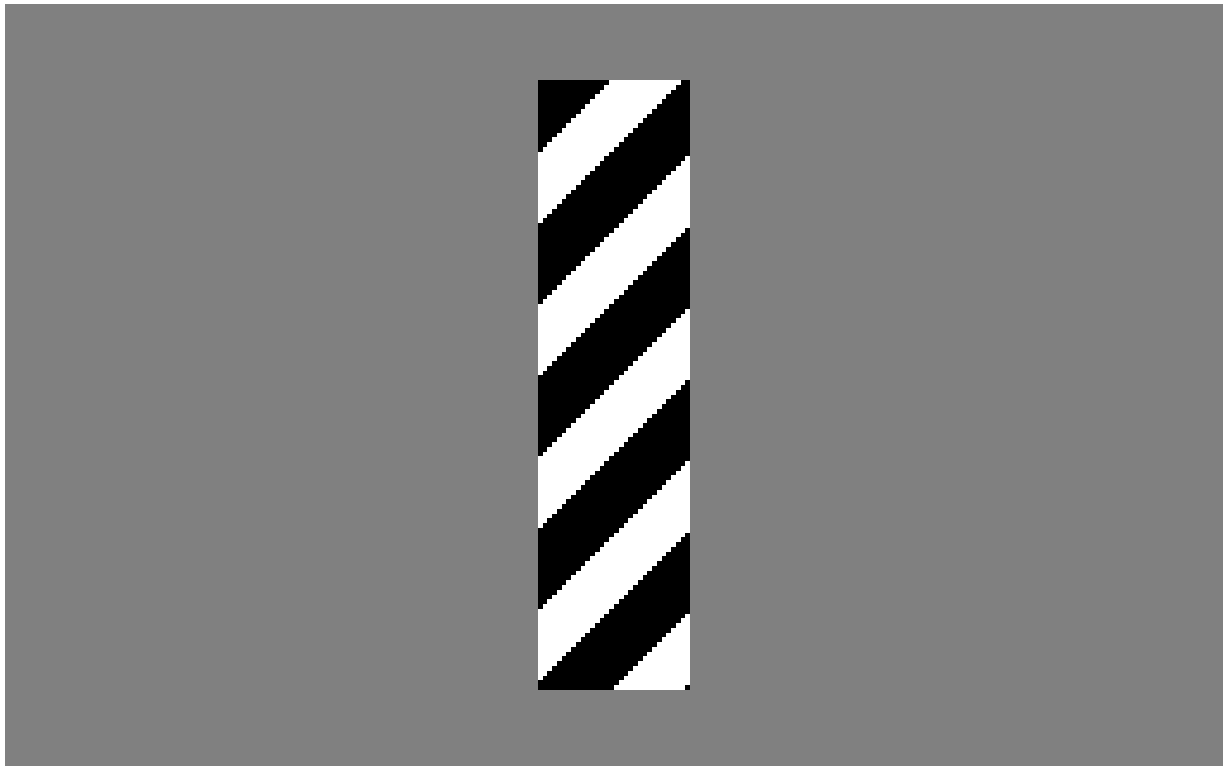
The Barber Pole Illusion



http://en.wikipedia.org/wiki/Barberpole_illusion

Slide credit S. Lazebnik

The Barber Pole Illusion



http://en.wikipedia.org/wiki/Barberpole_illusion

Slide credit S. Lazebnik

How to Address the Ambiguity?

- Brightness Constancy Equation: $\frac{\partial I}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial I}{\partial y}\frac{\partial y}{\partial t} + \frac{\partial I}{\partial t} = 0$ has two unknowns $(\frac{\partial x}{\partial t}, \frac{\partial y}{\partial t})$ and 1 equation for 1 pixel
- From the previous examples, we know that looking at more pixels of the same motion helps to resolve the ambiguity
- Idea: solve optical flow with a group of pixels moving in the same velocity!

Lukas-Kanade Method: Address the Ambiguity with Spatial Consistency Constraint

- Spatial Consistency Constraint: assume neighboring pixels have the same motion
- If we have $\boldsymbol{n}$ pixels in the neighborhood, then we can set up a linear least squares system:

$$\begin{bmatrix} \frac{\partial I(x_1, y_1)}{\partial x} & \frac{\partial I(x_1, y_1)}{\partial y} \\ \vdots & \vdots \\ \frac{\partial I(x_n, y_n)}{\partial x} & \frac{\partial I(x_b, y_n)}{\partial y} \end{bmatrix} \begin{bmatrix} \frac{\partial x}{\partial t} \\ \frac{\partial y}{\partial t} \end{bmatrix} = - \begin{bmatrix} \frac{\partial I(x_1, y_1)}{\partial t} \\ \vdots \\ \frac{\partial I(x_n, y_b)}{\partial t} \end{bmatrix}$$

Slide adapted from S. Lazebnik

Estimate Optical Flow

$$\begin{bmatrix} \frac{\partial I(x_1,y_1)}{\partial x} & \frac{\partial I(x_1,y_1)}{\partial y} \\ \vdots & \vdots \\ \frac{\partial I(x_n,y_n)}{\partial x} & \frac{\partial I(x_b,y_n)}{\partial y} \end{bmatrix} \begin{bmatrix} \frac{\partial x}{\partial t} \\ \frac{\partial y}{\partial t} \end{bmatrix} = -\begin{bmatrix} \frac{\partial I(x_1,y_1)}{\partial t} \\ \vdots \\ \frac{\partial I(x_n,y_b)}{\partial t} \end{bmatrix}$$

$A \qquad d \qquad b$

$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} \begin{bmatrix} \frac{\partial x}{\partial t} \\ \frac{\partial y}{\partial t} \end{bmatrix} = -\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$$

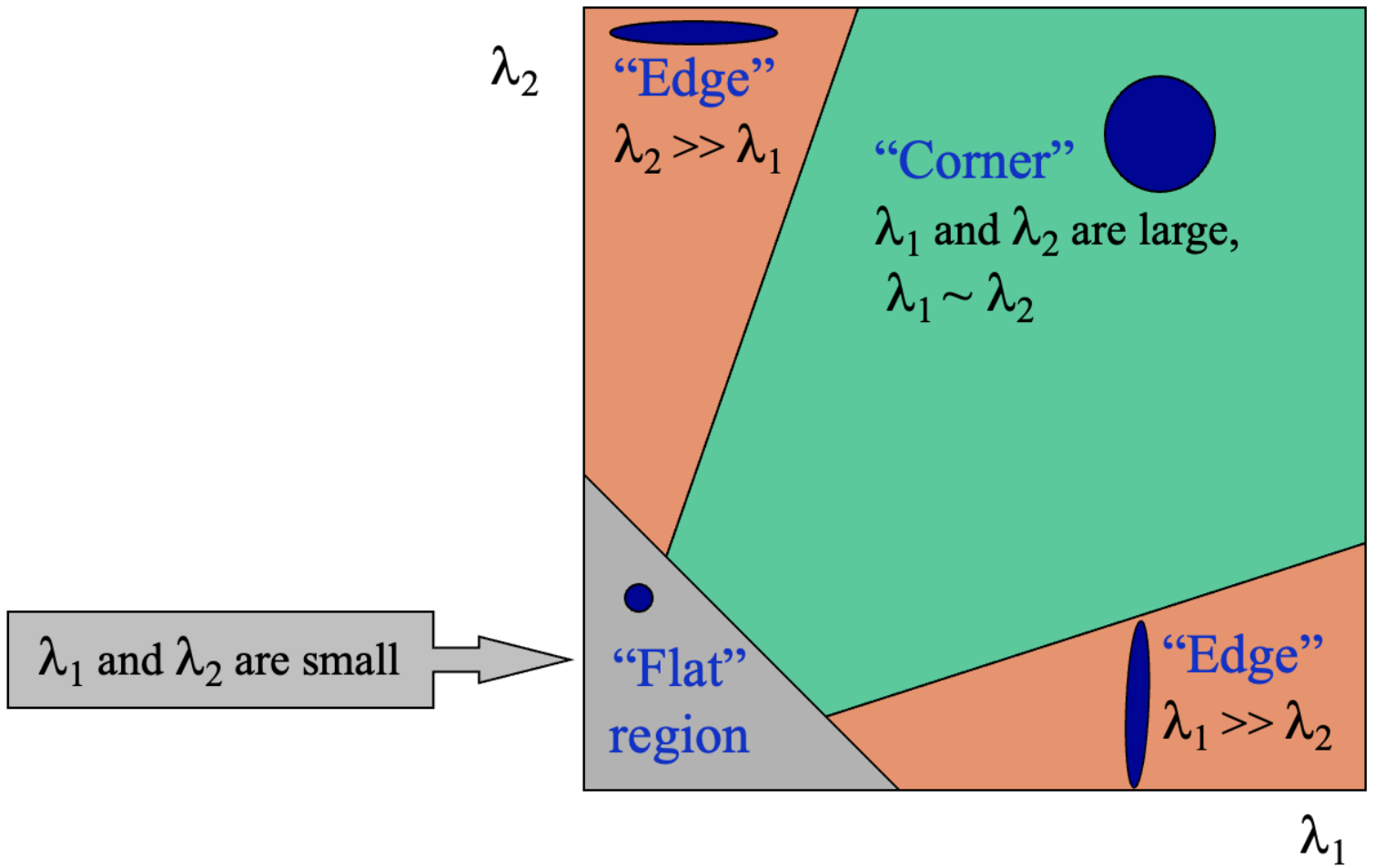
$A^{\intercal}A \qquad d \qquad A^{\intercal}b$

- This is a least square problem: the solutions of $Ad = b$ are the solutions of $A^{\intercal}Ad = A^{\intercal}b$
- Solution: $\mathrm{d} = (A^{\intercal}A)^{-1}A^{\intercal}b$
- When is the system solvable?

Slide adapted from S. Lazebnik

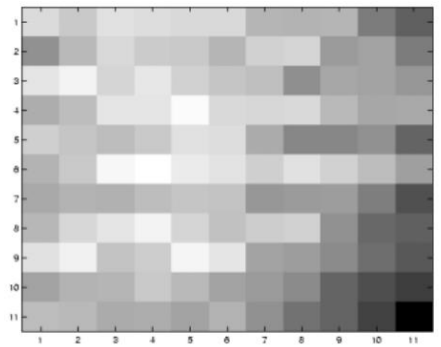
Recall: second moment matrix

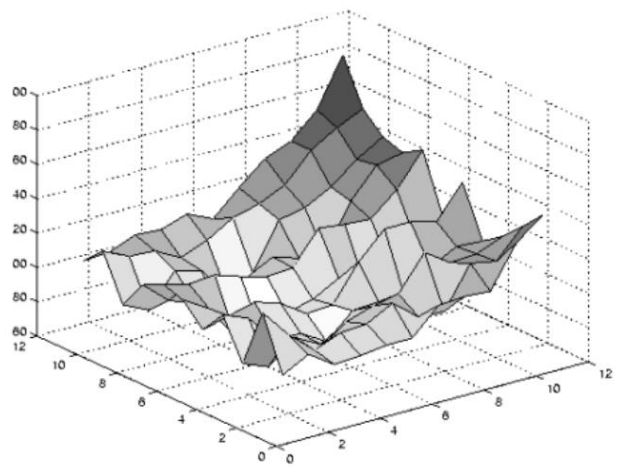
- Estimation of optical flow is well-conditioned precisely for regions with high "cornerness":



Slide credit S. Lazebnik

Low texture region





$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \nabla I (\nabla I)^T$$

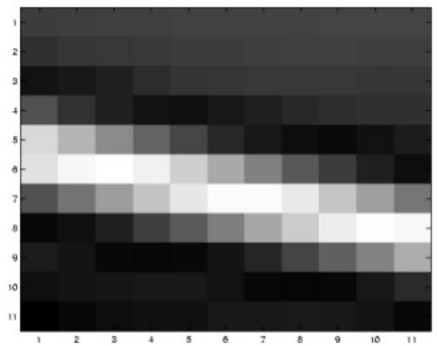
– gradients have small magnitude

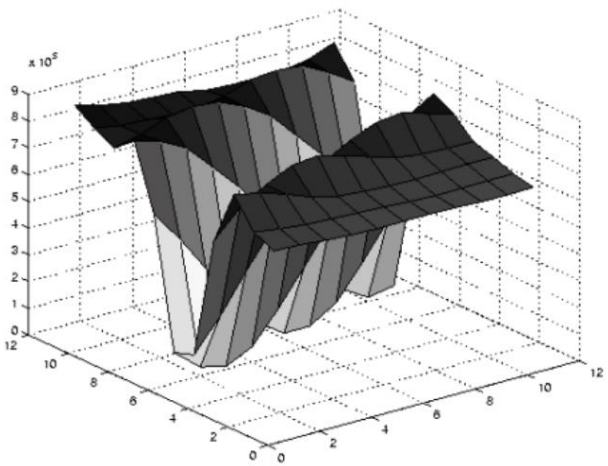
– small λ_1, small λ_2

Slide credit: S. Lazebnik

Slide credit J. Malik and A. Efros

Edge





$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \nabla I (\nabla I)^T$$

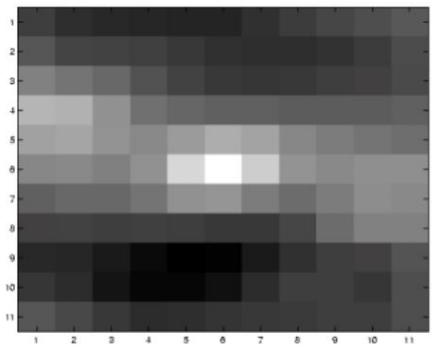
– large gradients, all the same

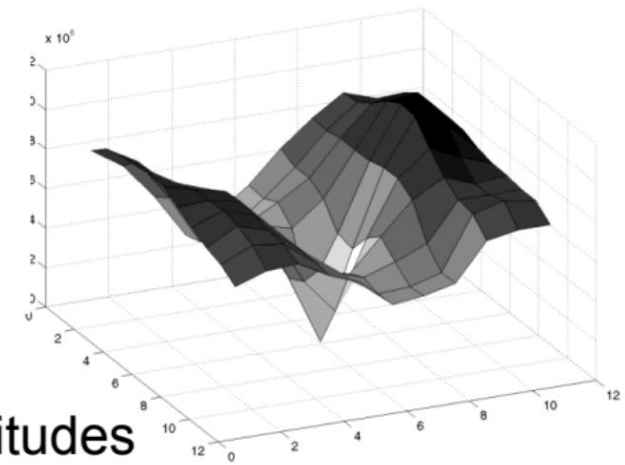
– large λ_1, small λ_2

Slide credit: S. Lazebnik

Slide credit J. Malik and A. Efros

High texture region





$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \nabla I (\nabla I)^{\mathrm{T}}$$

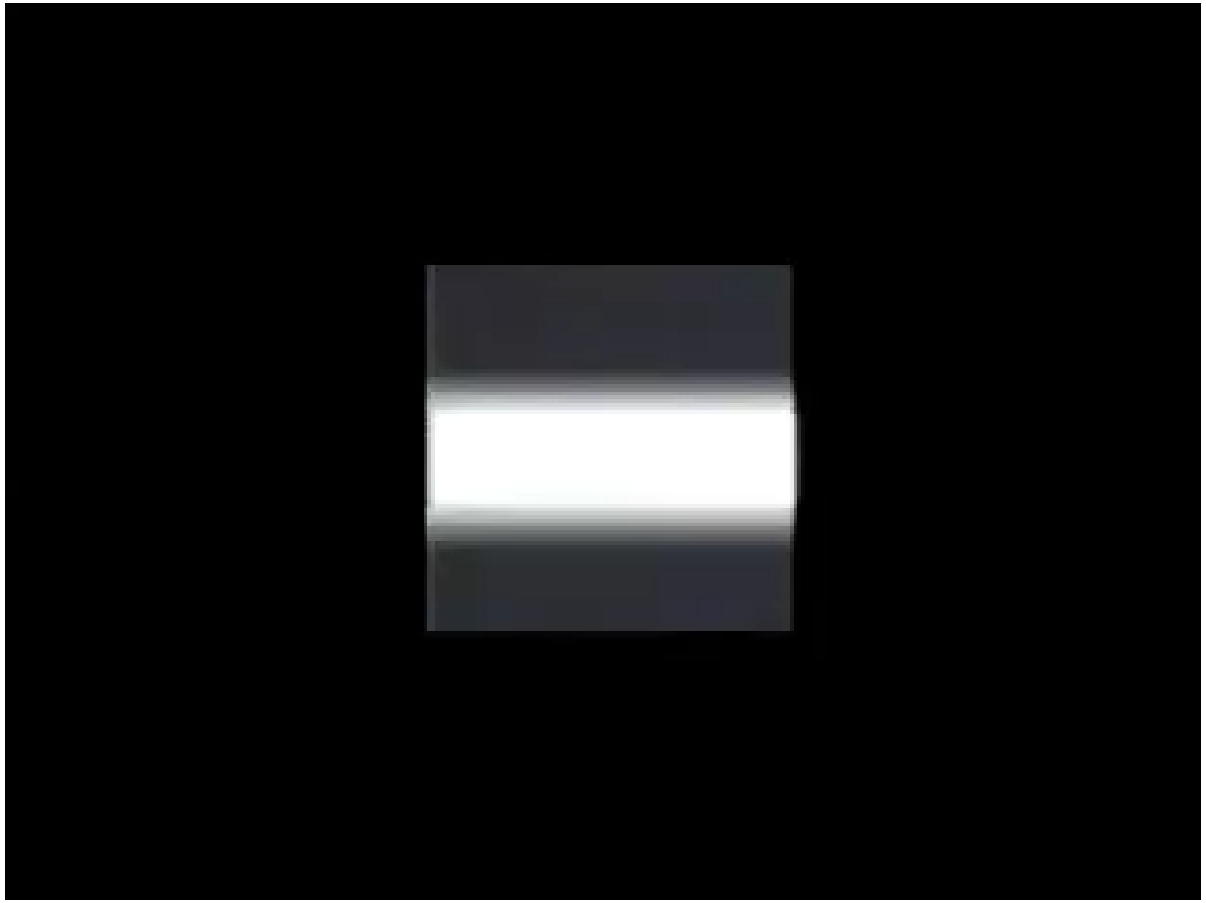
– gradients are different, large magnitudes

– large λ_1, large λ_2

Slide credit: S. Lazebnik

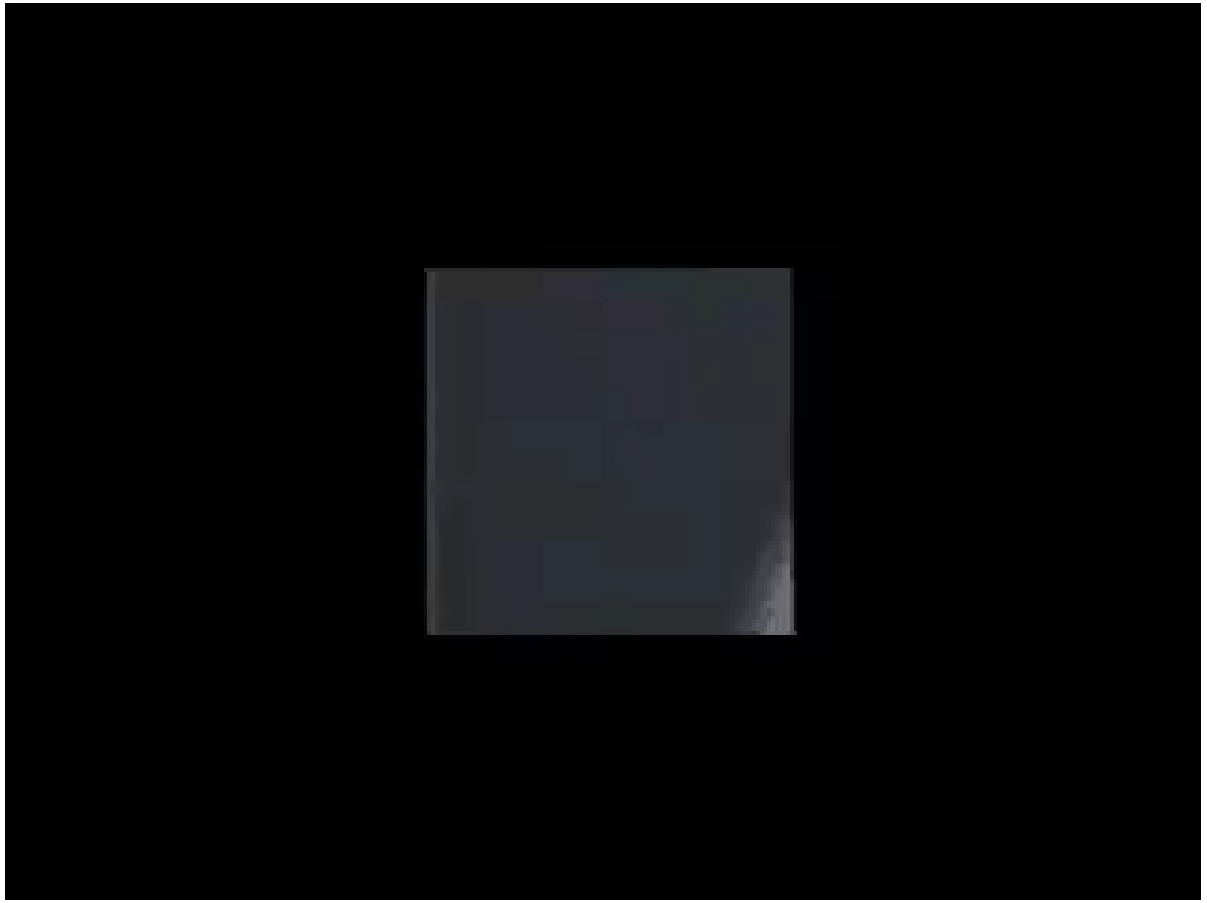
Slide credit J. Malik and A. Efros

Conditions for solvability: A Bad Case of Single Straight Edge



Slide credit S. Lazebnik

Conditions for solvability: A Good Case

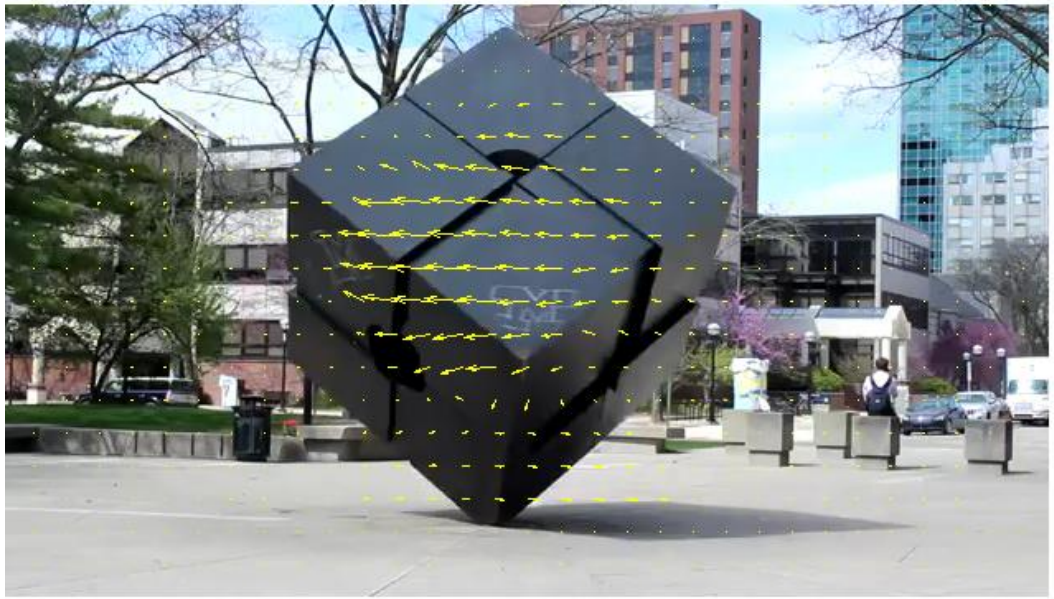


Slide credit S. Lazebnik

Lukas-Kanade Flow Example

Input frames

Output

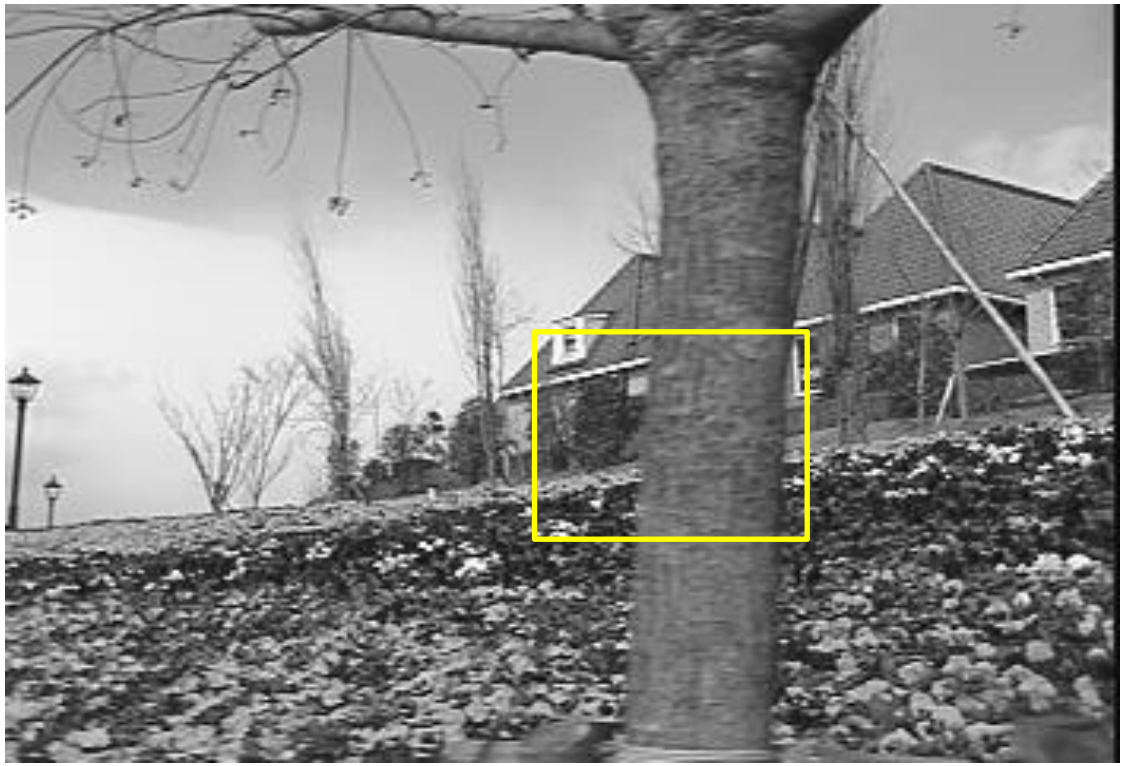


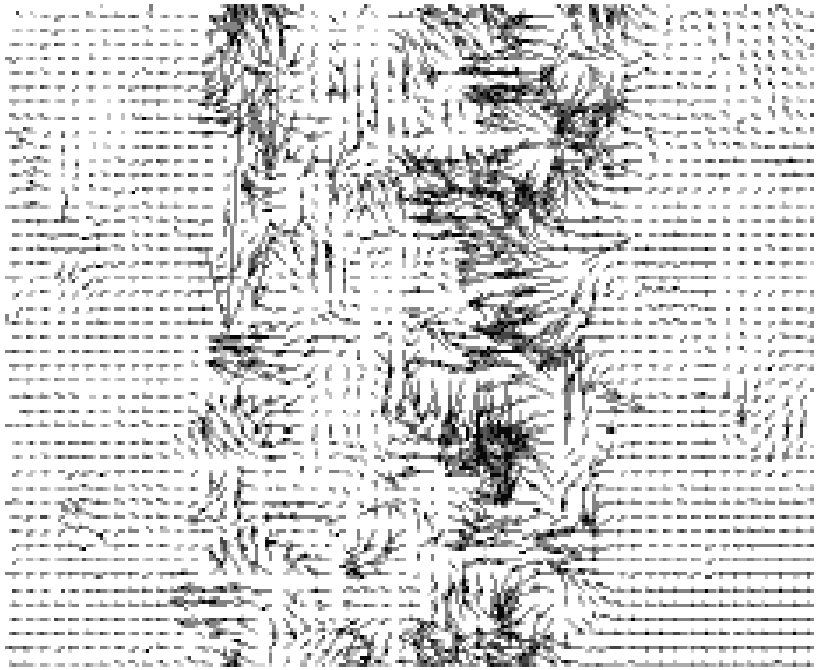
Slide credit S. Lazebnik

When Does Lukas-Kanade Method Fail?

- Up to now, we have assumed:
  1. 1st order Taylor expansion assumes small motions
  2. Spatial consistency constraint
  3. Brightness constancy constraint

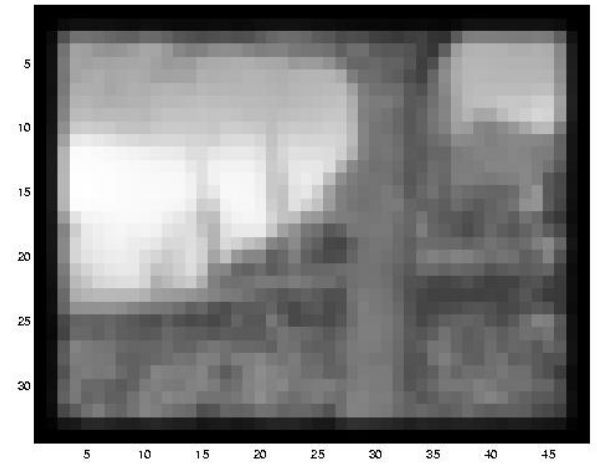
Lukas-Kanade Method Fails when Motion is Large

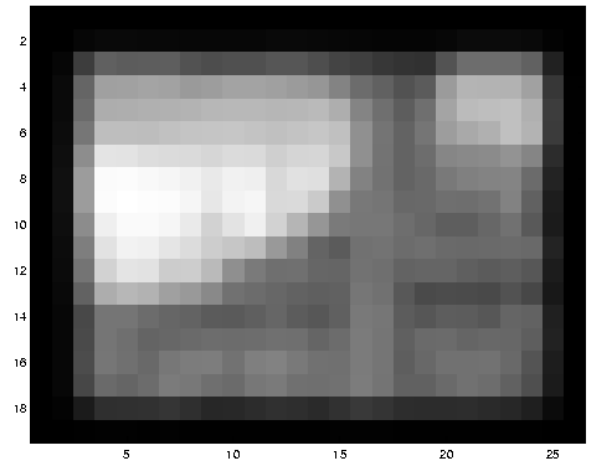




Slide credit S. Lazebnik

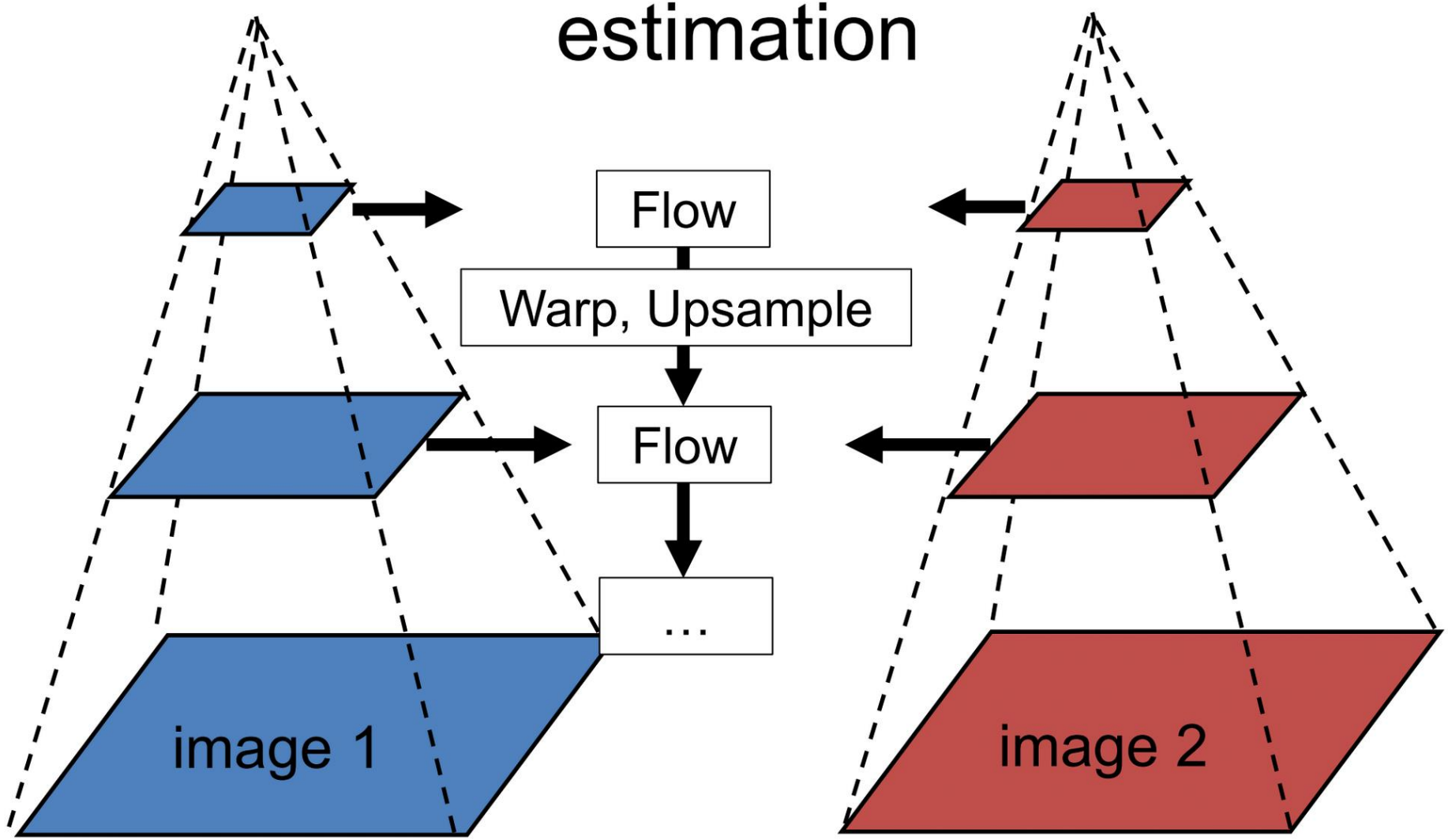
Solution: Reduce the Resolution!





Slide credit S. Lazebnik

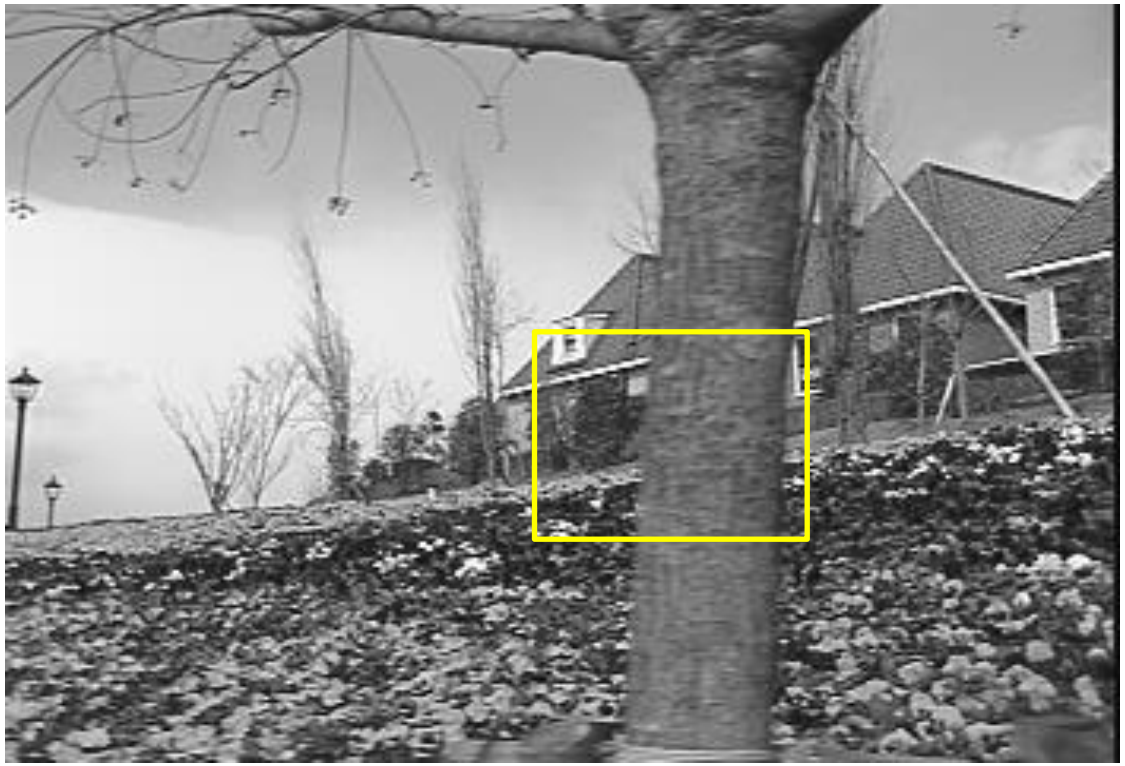
Coarse-to-fine optical flow estimation

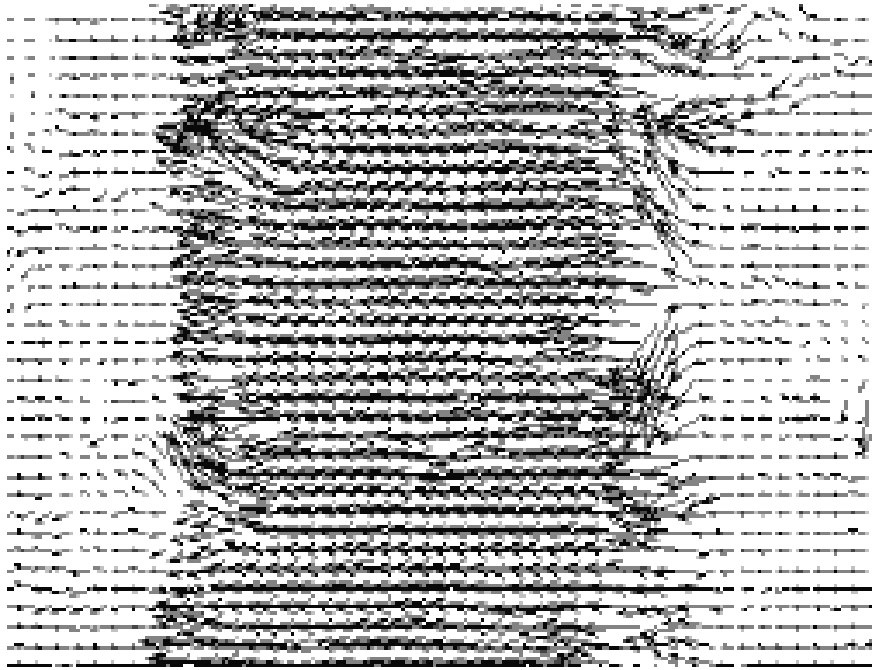


Slide credit: S. Lazebnik

Slide credit J. Malik and A. Efros

After Fixing





Slide credit S. Lazebnik

Lukas-Kanade Method Fails when Neighboring Pixels Don't Move in the Same Way

- Solution: Figure out which pixels move together then come back and fix

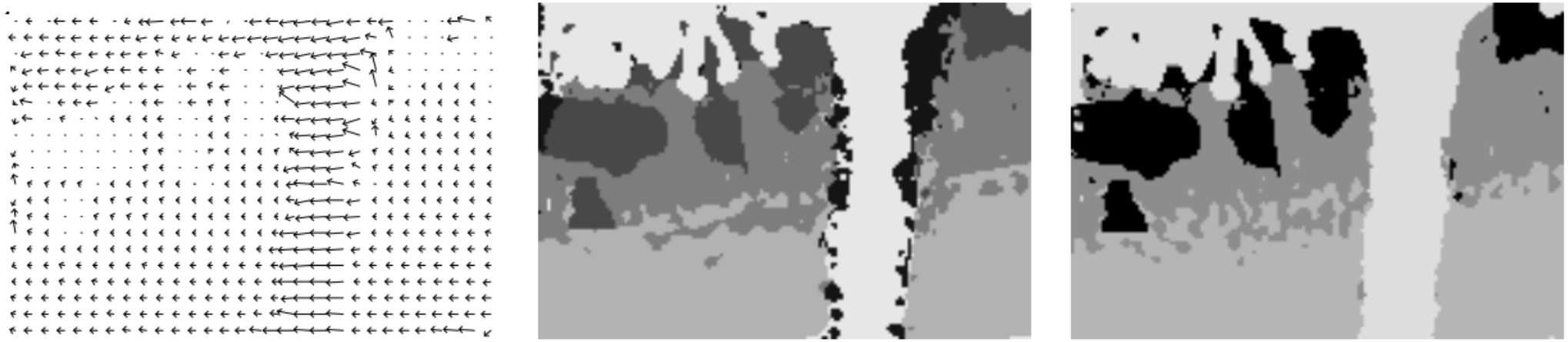


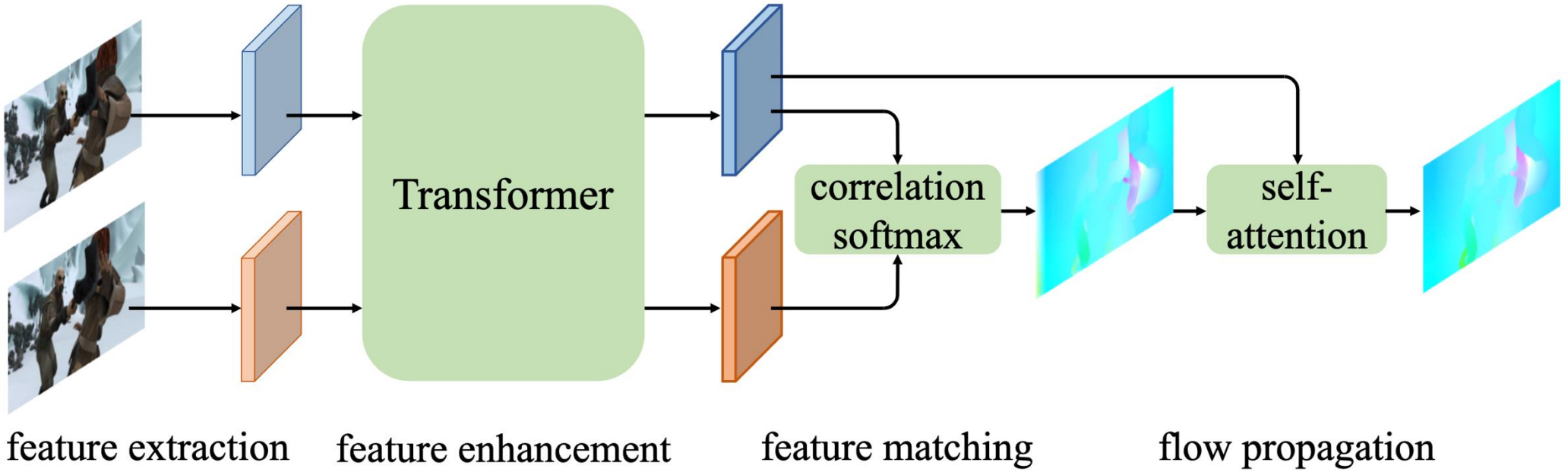
Figure 11: (a) The optic flow from multi-scale gradient method. (b) Segmentation obtained by clustering optic flow into affine motion regions. (c) Segmentation from consistency checking by image warping. Representing moving images with layers.

J. Wang and E. Adelson. Representing Moving Images with Layers. IEEE Transactions on Image Processing, 1994

Slide credit S. Lazebnik

Lukas-Kanade Method Fails when Brightness Constancy Constraint Doesn't Hold

- Solution: Use other forms of matching (e.g. interest points) or learn a neural network for matching!

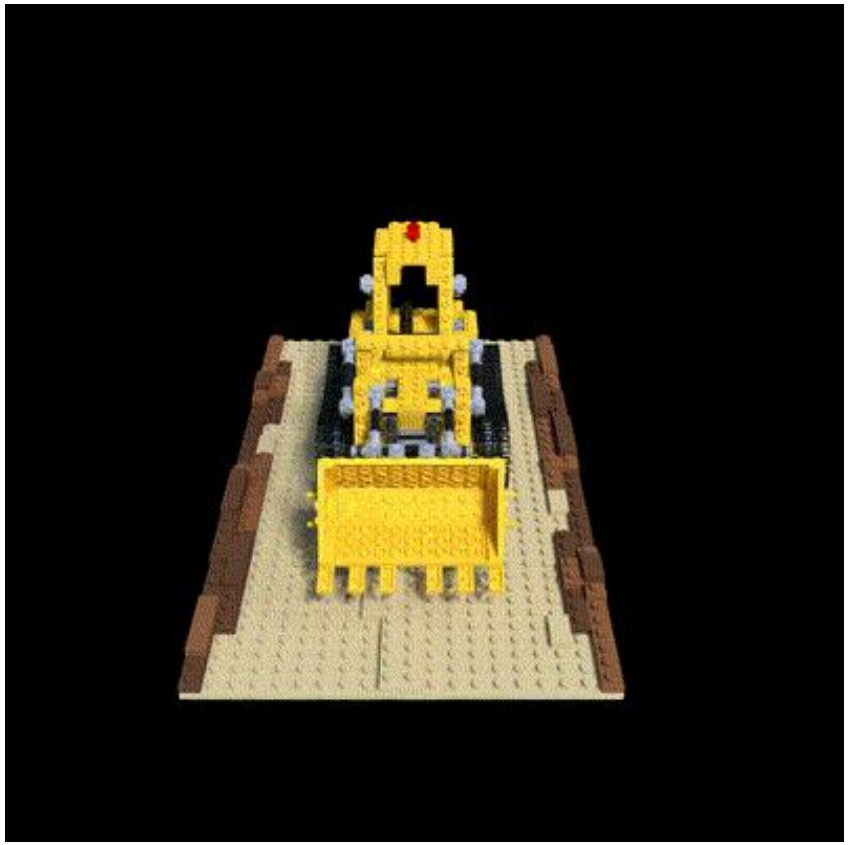


GMFlow: Learning Optical Flow via Global Matching. Xu et al.

Content

- 2D Motion Representations
- 3D Motion Representations
  - ➢4D Volume Rendering with NERF
  - ➢Explicit Motion Field with NERF
  - ➢4D Volume Rendering with 3DGS
  - ➢Explicit Motion Field with 3DGS
  - ➢Implicit Motion Field with 3DGS
  - ➢4D Foundation Models
  - ➢Multi-view Video Generation (we'll cover in the next lecture)

Optical Flow Only Describes 2D Motion. However, We Live in a 3D World, and 3D Motion Matters



Deforming Radiance Fields with Cages. Xu et al.

Deformable 3D Gaussians for High-Fidelity Monocular Dynamic Scene Reconstruction. Yang et al.

How Can We Model 3D World in Time?

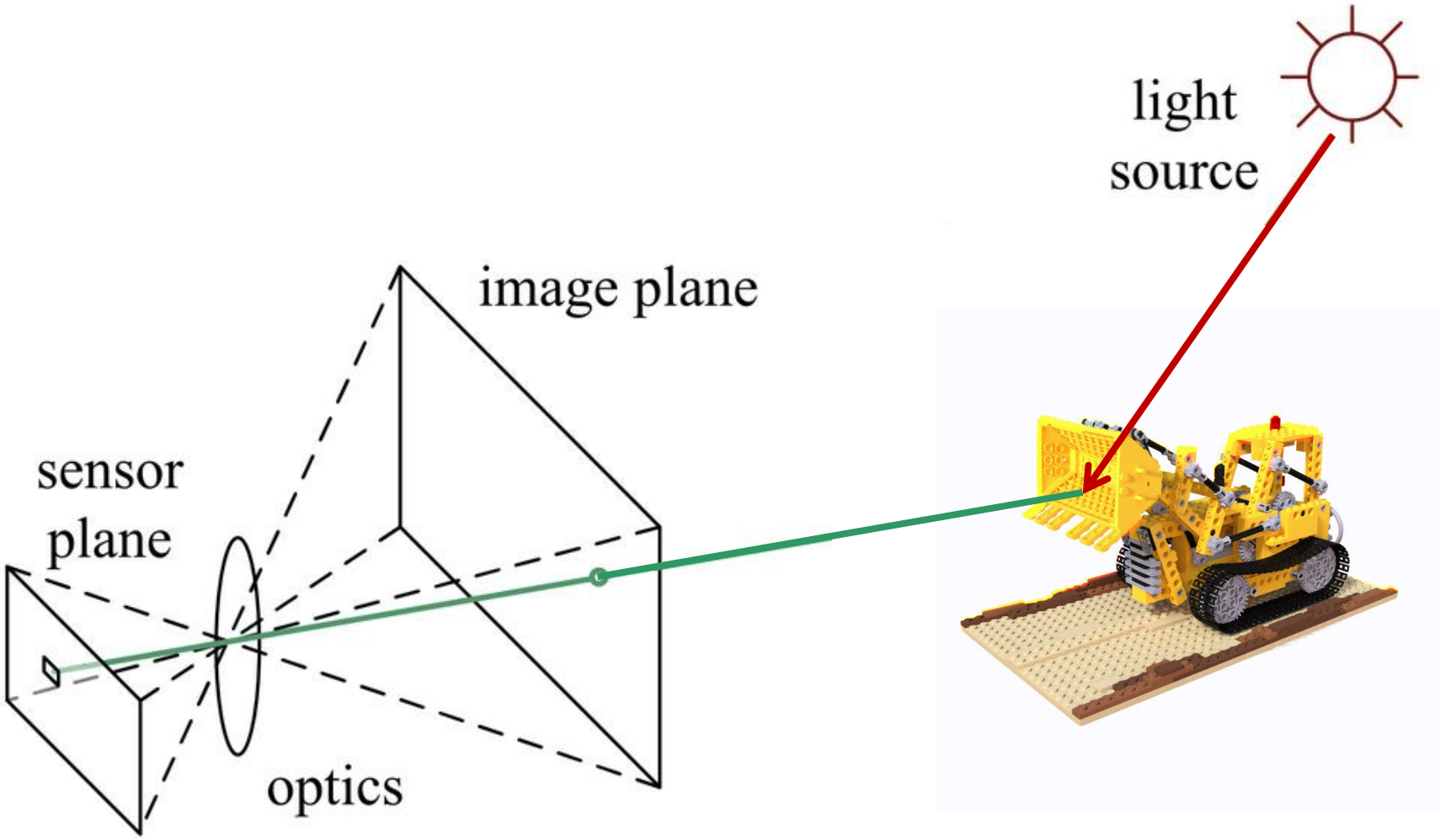
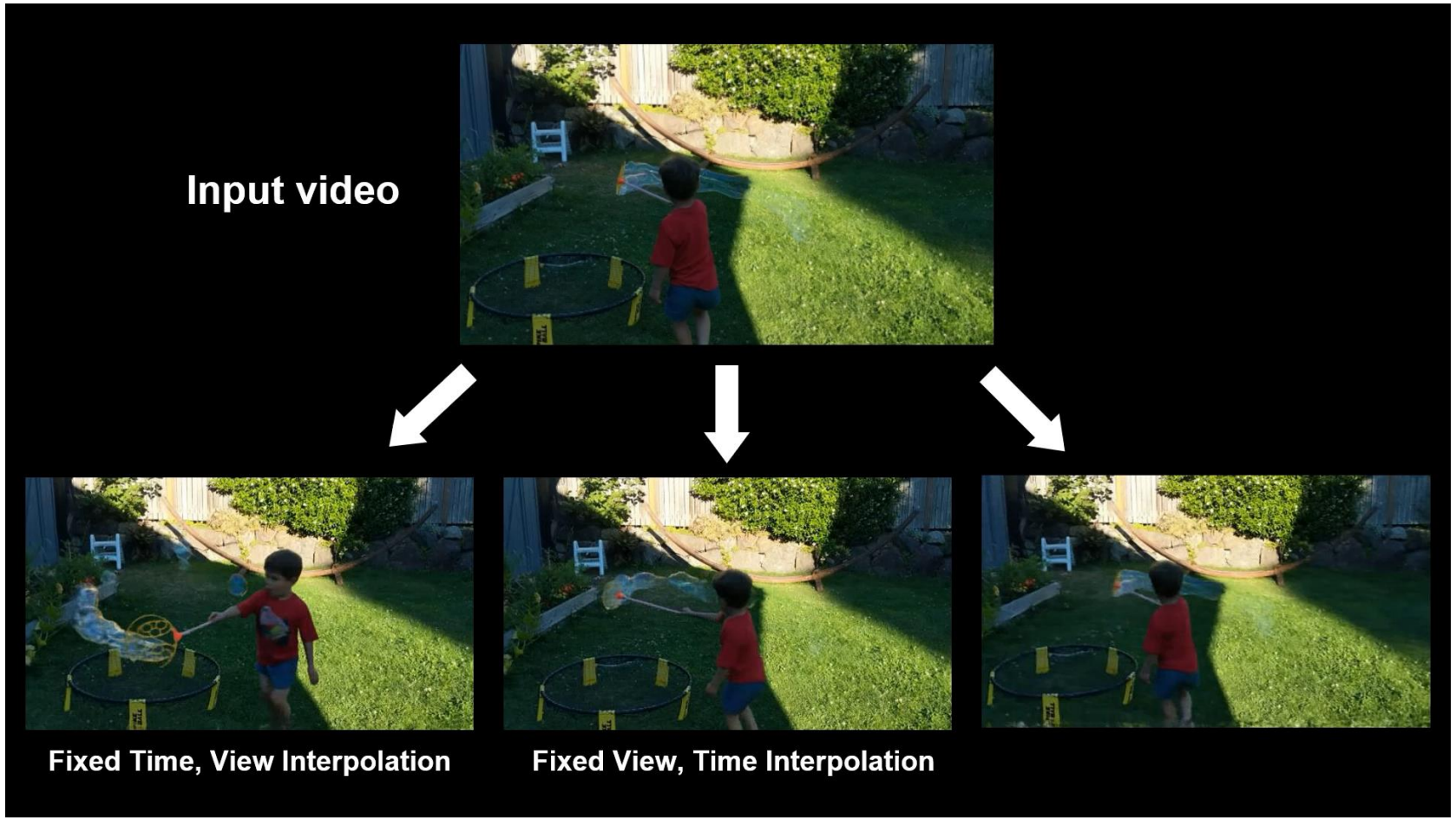


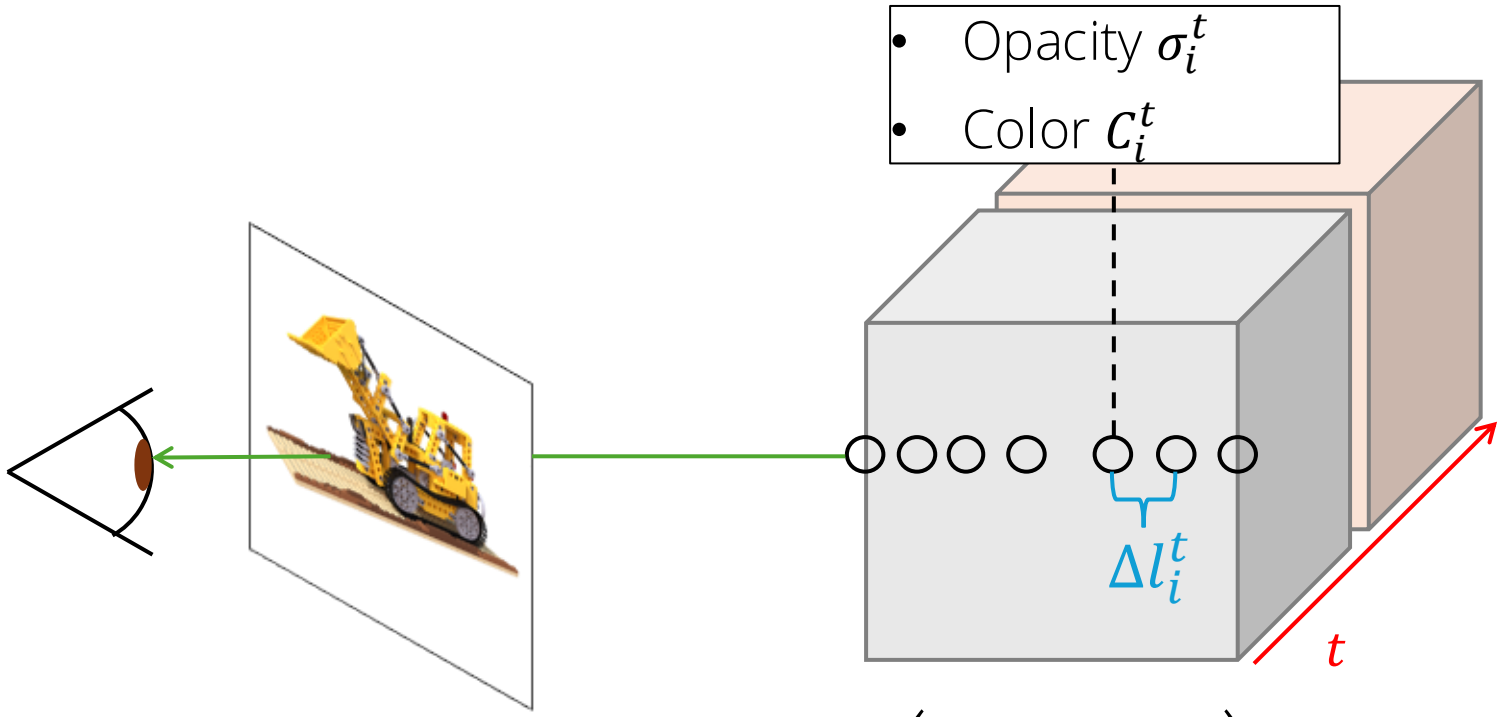
Image source S. Seitz text book

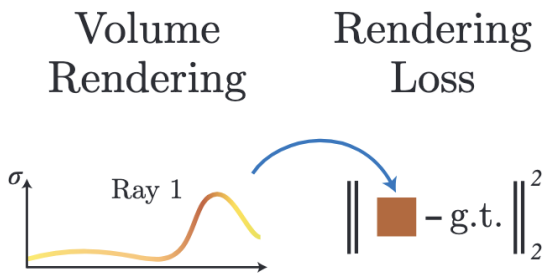
How Can We Model 3D World in Time by Observing Monocular Videos?



Neural Scene Flow Fields for Space Time View Synthesis of Dynamic Scenes. Li et al.

Idea 1: Extend 3D Volume Rendering to 4D Volume Rendering

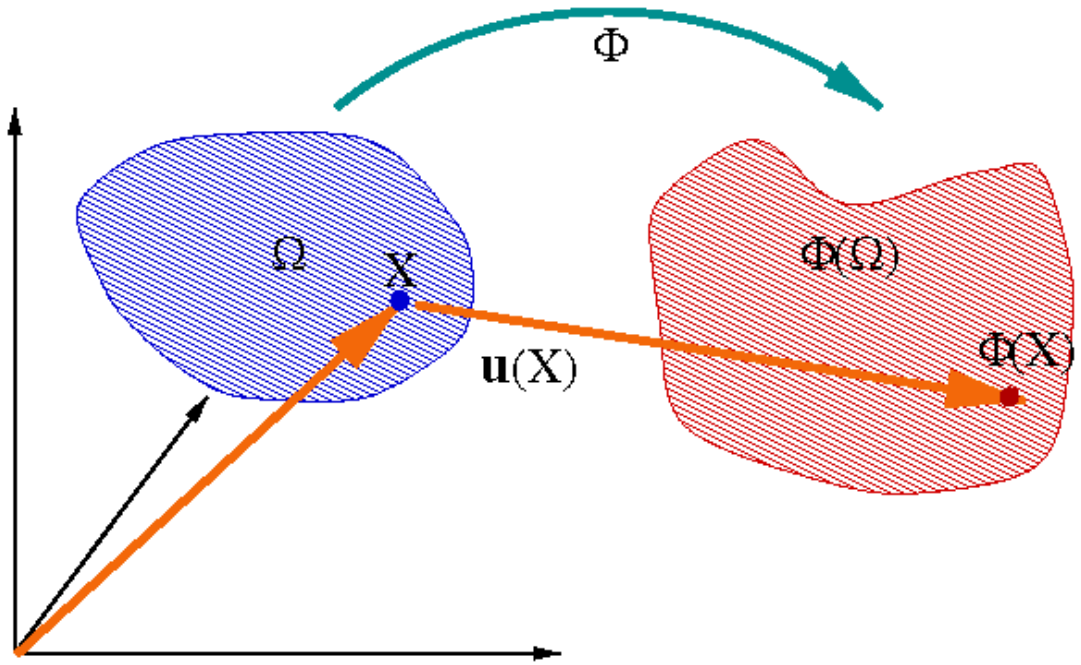




$$C(x,\omega,t) \approx \sum_{i=1}^{N}\left(\prod_{j=1}^{i-1} e^{-\sigma_j^t \Delta l_j^t}\right)\left(1-e^{-\sigma_i^t \Delta l_i^t}\right) C_i^t$$

Learning objective: $\min_{\sigma_i^t, C_i^t} \| C(x,\omega,t) - \text{g.t.} \|$

Idea 2: Model the Motion Explicitly



Φ : Deformation Function

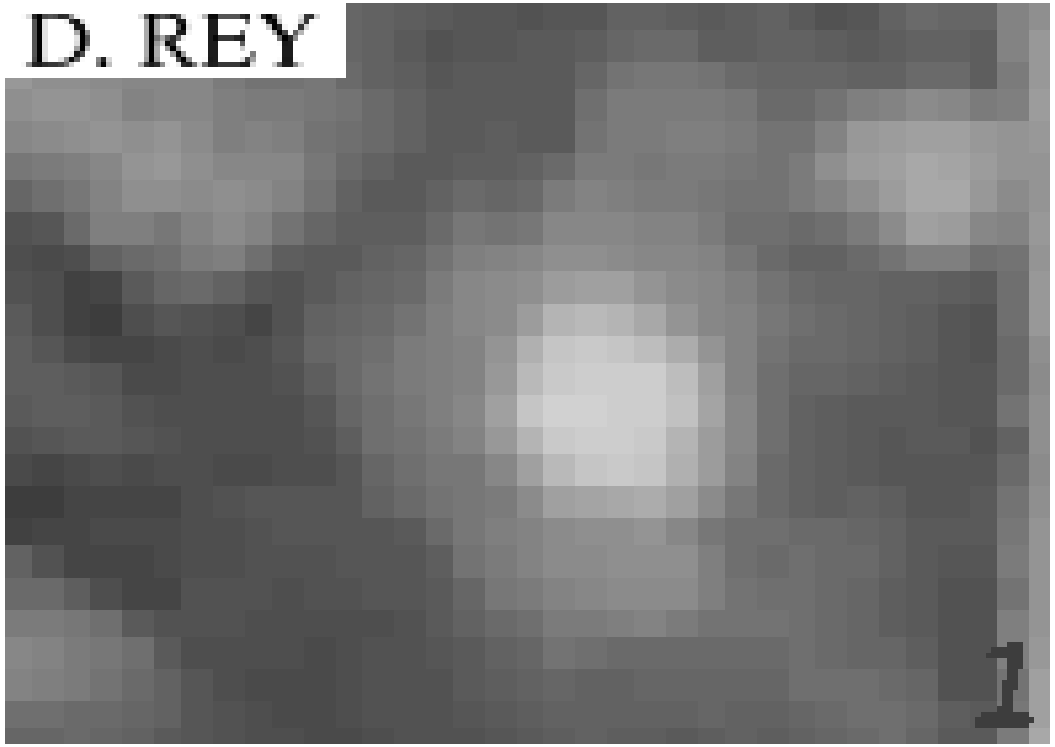
u : Displacement Field

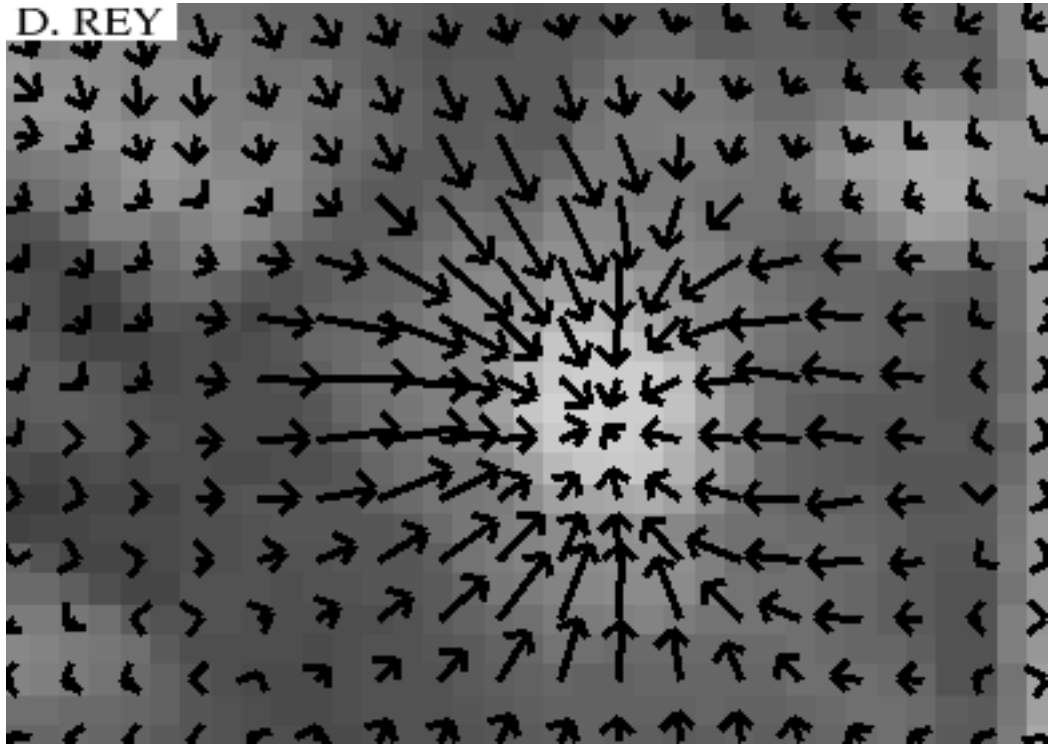
$$\phi(x) = x + \mathbf{u}(x) \Longrightarrow \nabla\phi = Id + \nabla\mathbf{u}$$

https://www-sop.inria.fr/epidaure/Demonstrations/ms_evolution/HTML/Field.html

Idea 2: Model the Motion Explicitly

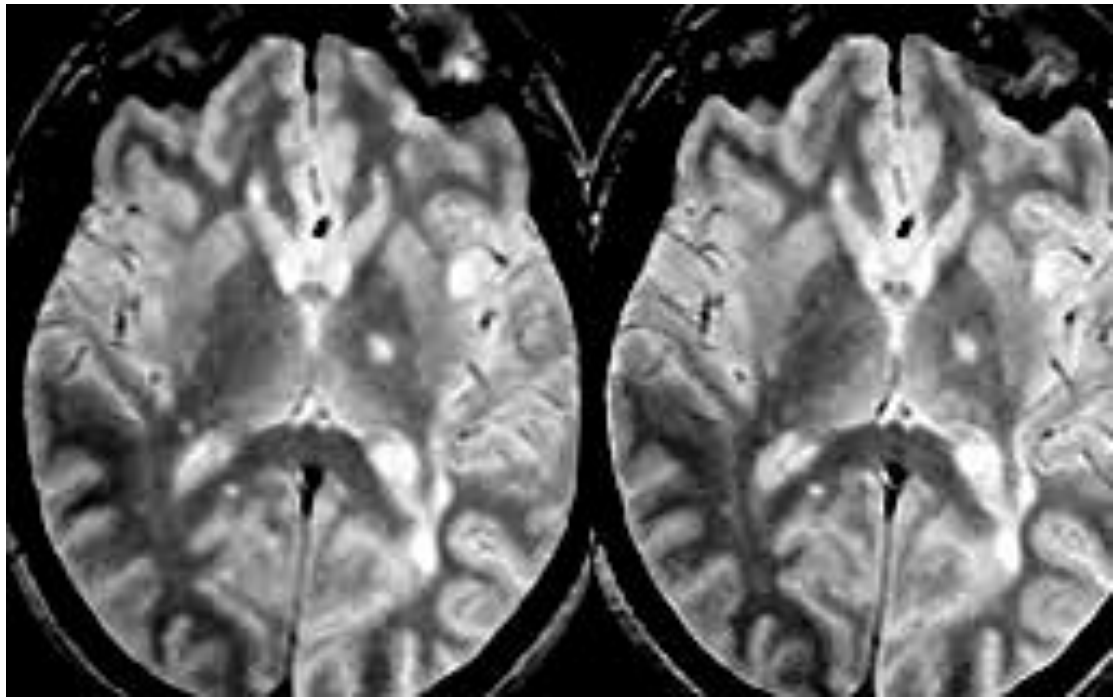
- Optical flow: Pixel Apparent Motion
- 2D Motion flow: the exact particle 3D motion, projected/sliced at certain surface

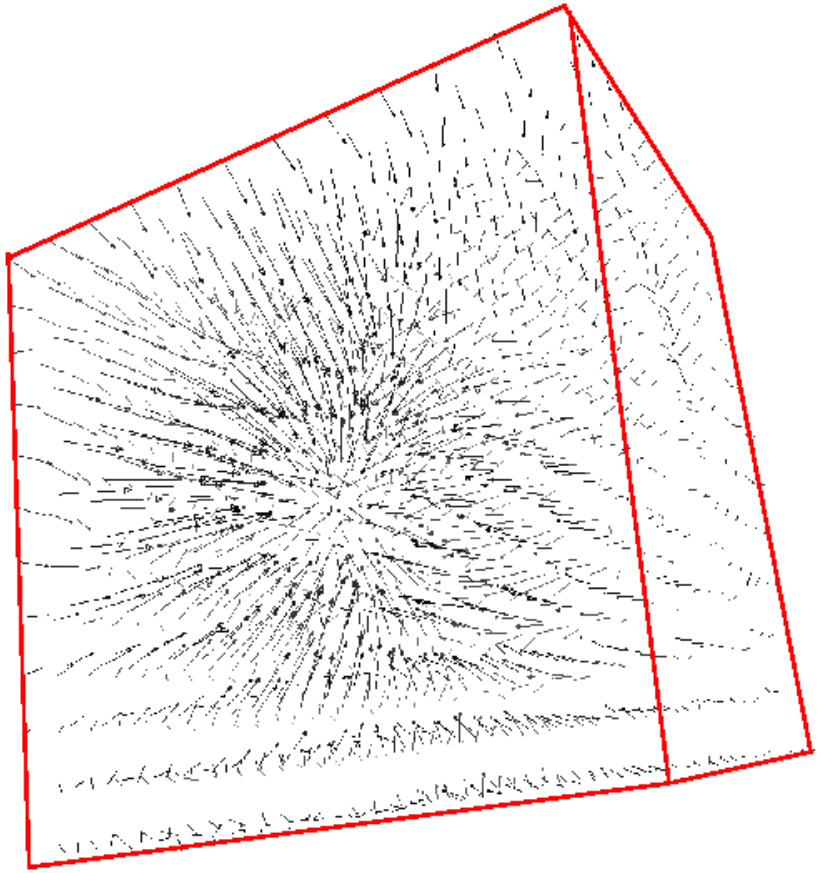




https://www-sop.inria.fr/epidaure/Demonstrations/ms_evolution/HTML/Field.html

Idea 2: Model the Motion Explicitly





https://www-sop.inria.fr/epidaure/Demonstrations/ms_evolution/HTML/Field.html

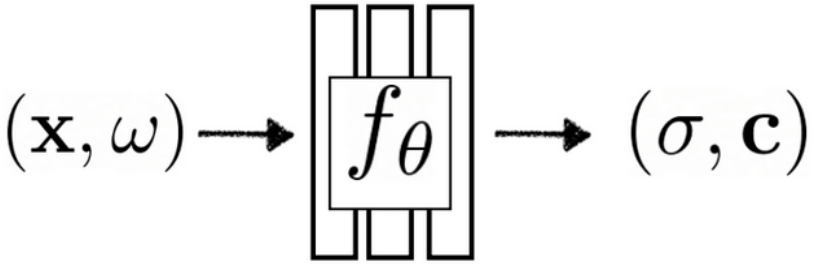
Idea 3: Model Multi-view Videos

ReCamMaster: Camera-Controlled Generative Rendering from A Single Video. Bai et al.

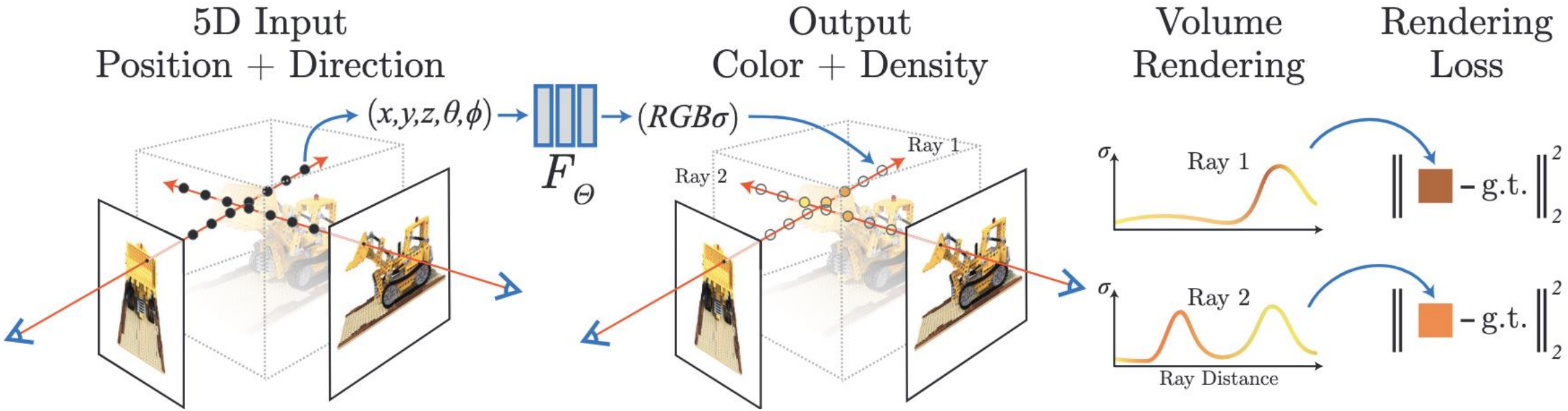
Content

- 2D Motion Representations
- 3D Motion Representations
  - ➢4D Volume Rendering with NERF
  - ➢Explicit Motion Field with NERF
  - ➢4D Volume Rendering with 3DGS
  - ➢Explicit Motion Field with 3DGS
  - ➢Implicit Motion Field with 3DGS
  - ➢4D Foundation Models
  - ➢Multi-view Video Generation (we'll cover in the next lecture)

NEural Radiance Field (NERF)

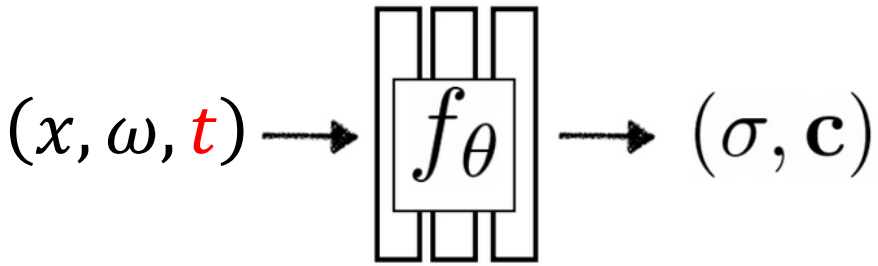


- Use a neural network f_θ to predict at each point x and view direction ω:
  - Opacity σ
  - Color c

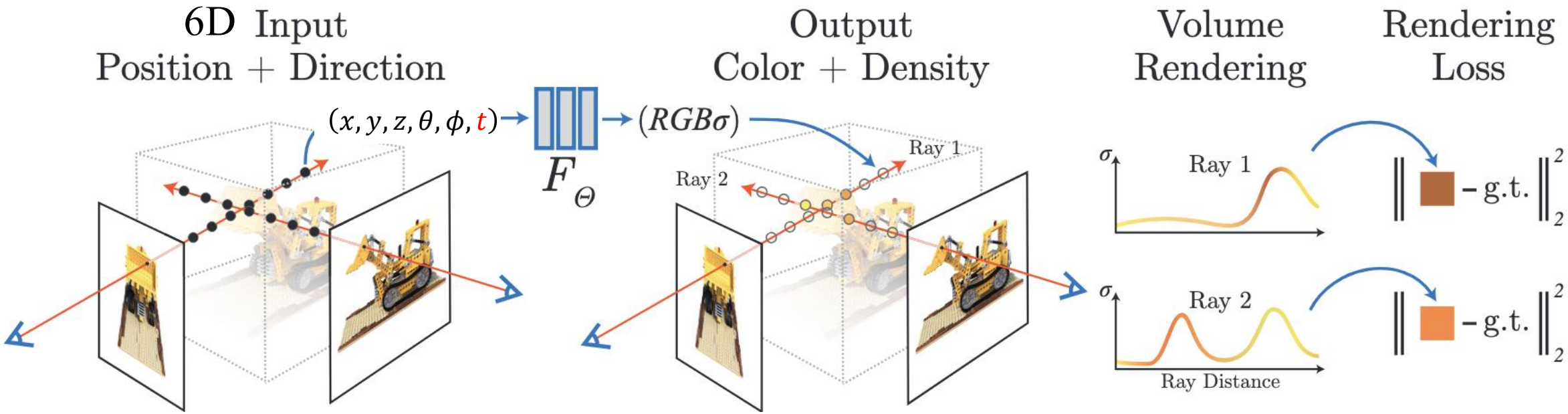


NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. Mildenhall et al.

Idea 1: Extend NERF to SpatioTemporal NERF



- Use a neural network f_θ to predict at each point x, view direction ω, and time step t :
  - Opacity σ
  - Color$c$



SpatioTemporal NERF Doesn't Work Out-of-Box

- **Problem**: A single video contains only one observation of the scene at any point in time. 3D geometry is under-constrained, e.g. "flat TV" solution

Space-time Neural Irradiance Fields for Free-Viewpoint Video. Xian et al.

SpatioTemporal NERF Doesn't Work Out-of-Box

- Solution:
  - Depth prediction loss: Enforce geometry by reconstructing depth

$$\mathcal{L}_{\text{depth}} = \sum_{(\mathbf{r},t)\in\mathcal{R}} \left\| \frac{1}{\hat{D}(\mathbf{r},t)} - \frac{1}{D(\mathbf{r},t)} \right\|_2^2, \qquad \hat{D}(\mathbf{r},t) = \int_{s_\text{n}}^{s_\text{f}} T(s,t)\,\sigma(\mathbf{r}(s),t)\,s\,\mathrm{d}s,$$

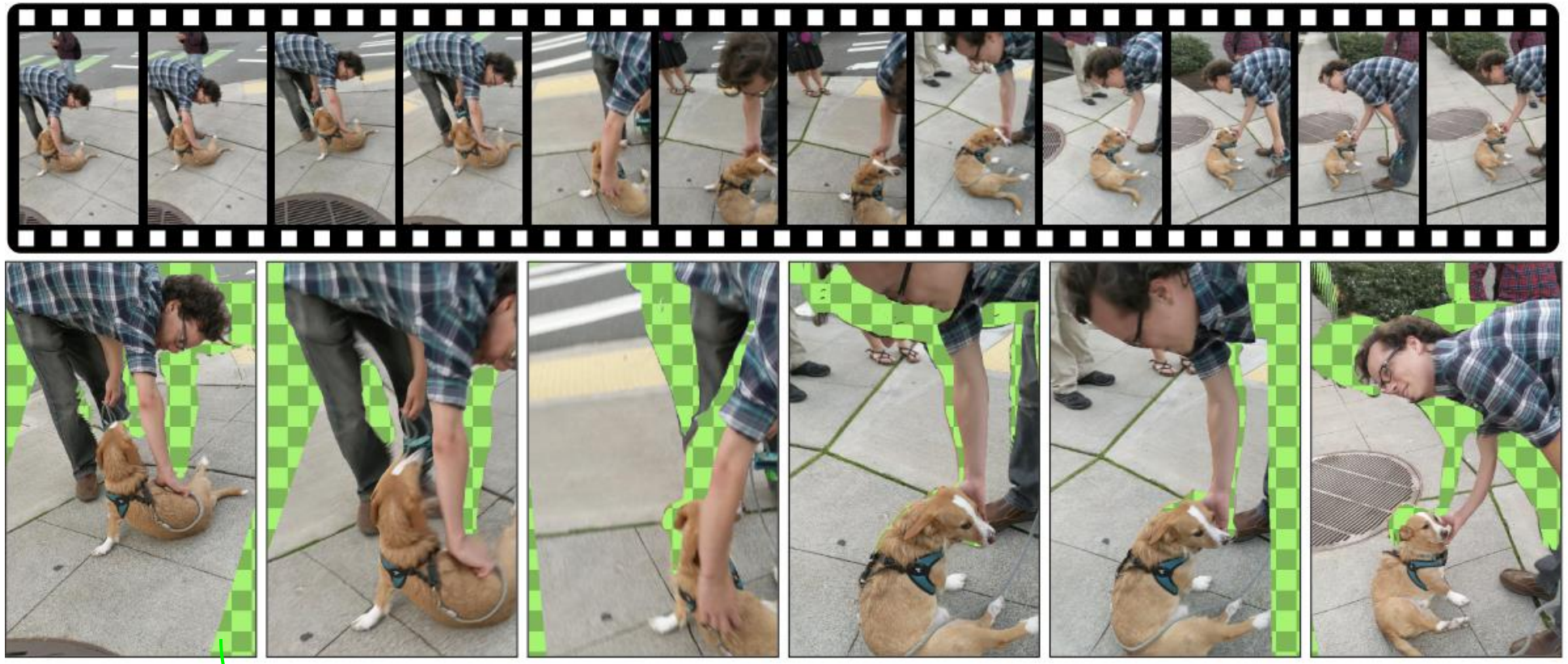
  - Empty-space loss: Depth loss is not enough, since the predicted depth is a *weighted sum* of depth values along the ray, resulting in haze-like artifacts. Penalize non-zero volume densities away from the depth surface

$$\mathcal{L}_{\text{empty}} = \sum_{(\mathbf{r},t)\in\mathcal{R}} \int_{s_\text{n}}^{d_t(\mathbf{u})-\varepsilon} \sigma(\mathbf{r}(s),t)\,\mathrm{d}s,$$

Space-time Neural Irradiance Fields for Free-Viewpoint Video. Xian et al.

SpatioTemporal NERF Doesn't Work Out-of-Box

- **Problem**: A large portion of spaces that is hidden from the input frame's viewpoint at any given time is still not constrained



Space-time Neural Irradiance Fields for Free-Viewpoint Video. Xian et al.

SpatioTemporal NERF Doesn't Work Out-of-Box

- Solution:
  - Static scene loss: Assume most parts of the scene are static, we can see the same regions, occluded at the current frame, from other *different viewpoints* at other *time*

$$\mathcal{L}_{\text{static}} = \sum_{(\mathbf{x},t)\in\mathcal{X}} \left\| F(\mathbf{x},t) - F(\mathbf{x},t') \right\|_2^2,$$

$$F: (x, t) \rightarrow (c, \sigma)$$

(a) Input frame (b) w/o static loss (c) w/ static loss

Space-time Neural Irradiance Fields for Free-Viewpoint Video. Xian et al.

Results

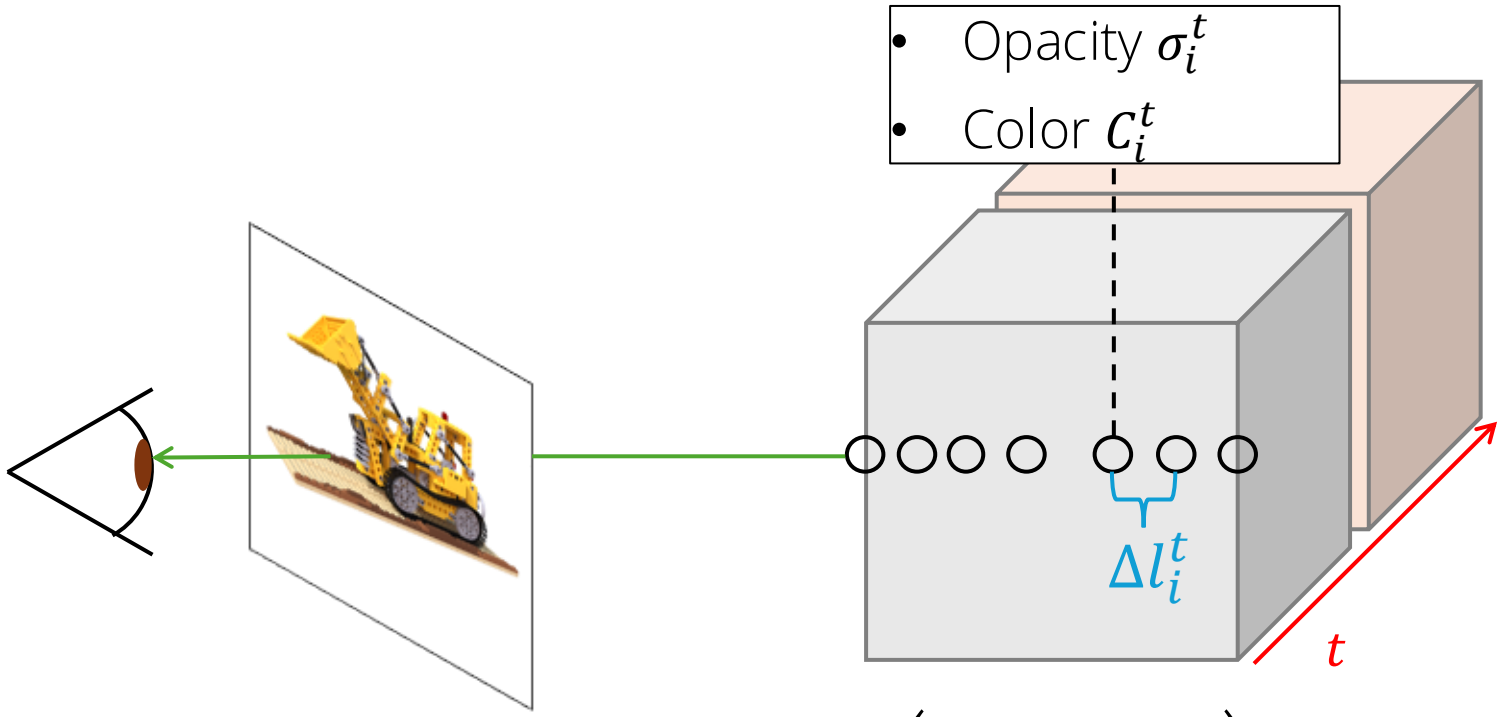
(a) Input (b) Mesh (c) Inpainted (d) NeRF-T (e) Ours

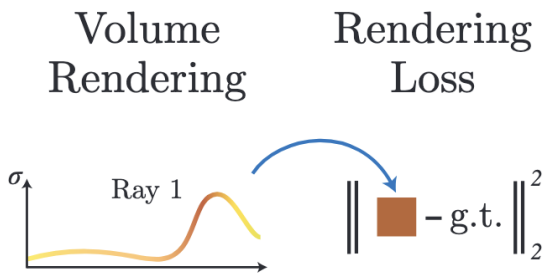
Space-time Neural Irradiance Fields for Free-Viewpoint Video.
Xian et al.

Content

- 2D Motion Representations
- 3D Motion Representations
  - ➢4D Volume Rendering with NERF
  - ➢Explicit Motion Field with NERF
  - ➢4D Volume Rendering with 3DGS
  - ➢Explicit Motion Field with 3DGS
  - ➢Implicit Motion Field with 3DGS
  - ➢4D Foundation Models
  - ➢Multi-view Video Generation (we'll cover in the next lecture)

However, 4D NERF Doesn't Model Motion Explicitly. We don't Know the Position of the Same Point Across Time.

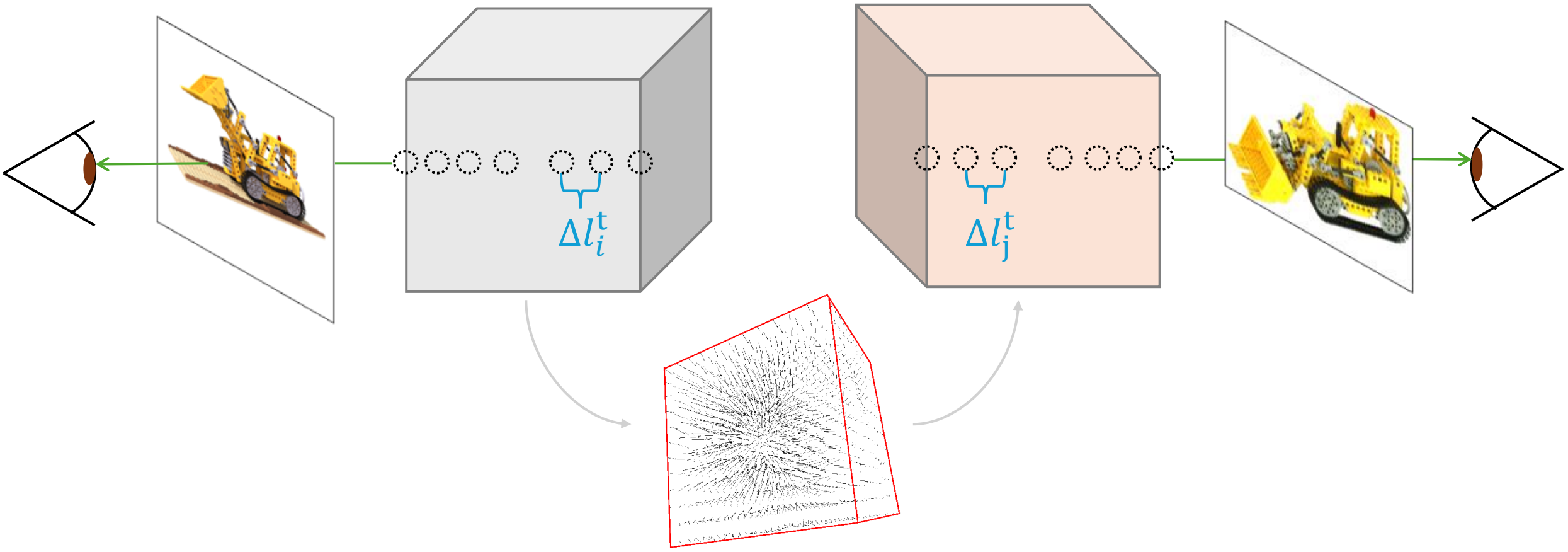




$$C(x, \omega, t) \approx \sum_{i=1}^{N} \left(\prod_{j=1}^{i-1} e^{-\sigma_j^t \Delta l_j^t} \right) \left(1 - e^{-\sigma_i^t \Delta l_i^t} \right) C_i^t$$

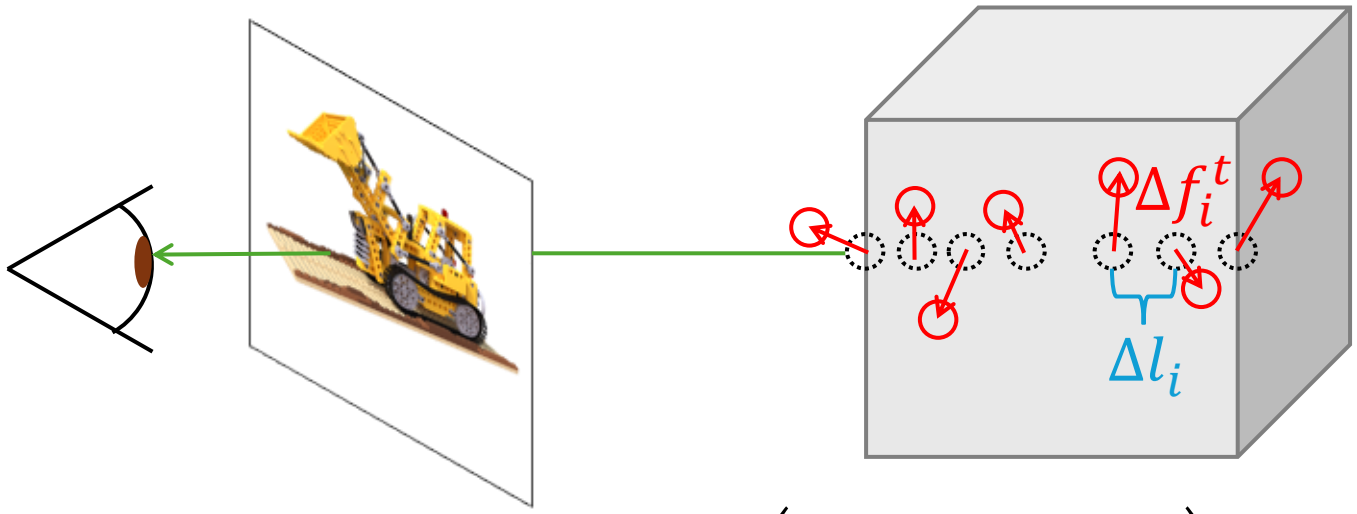
Learning objective: $\min_{\sigma_i^t, C_i^t} \| C(x, \omega, t) - \text{g.t.} \|$

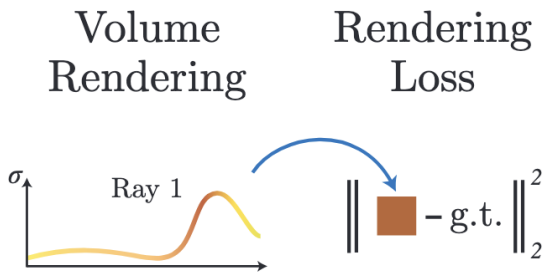
Idea 2: Model the Motion Explicitly



Idea 2: Model the Motion Explicitly

Δf_i^t denotes the motion field



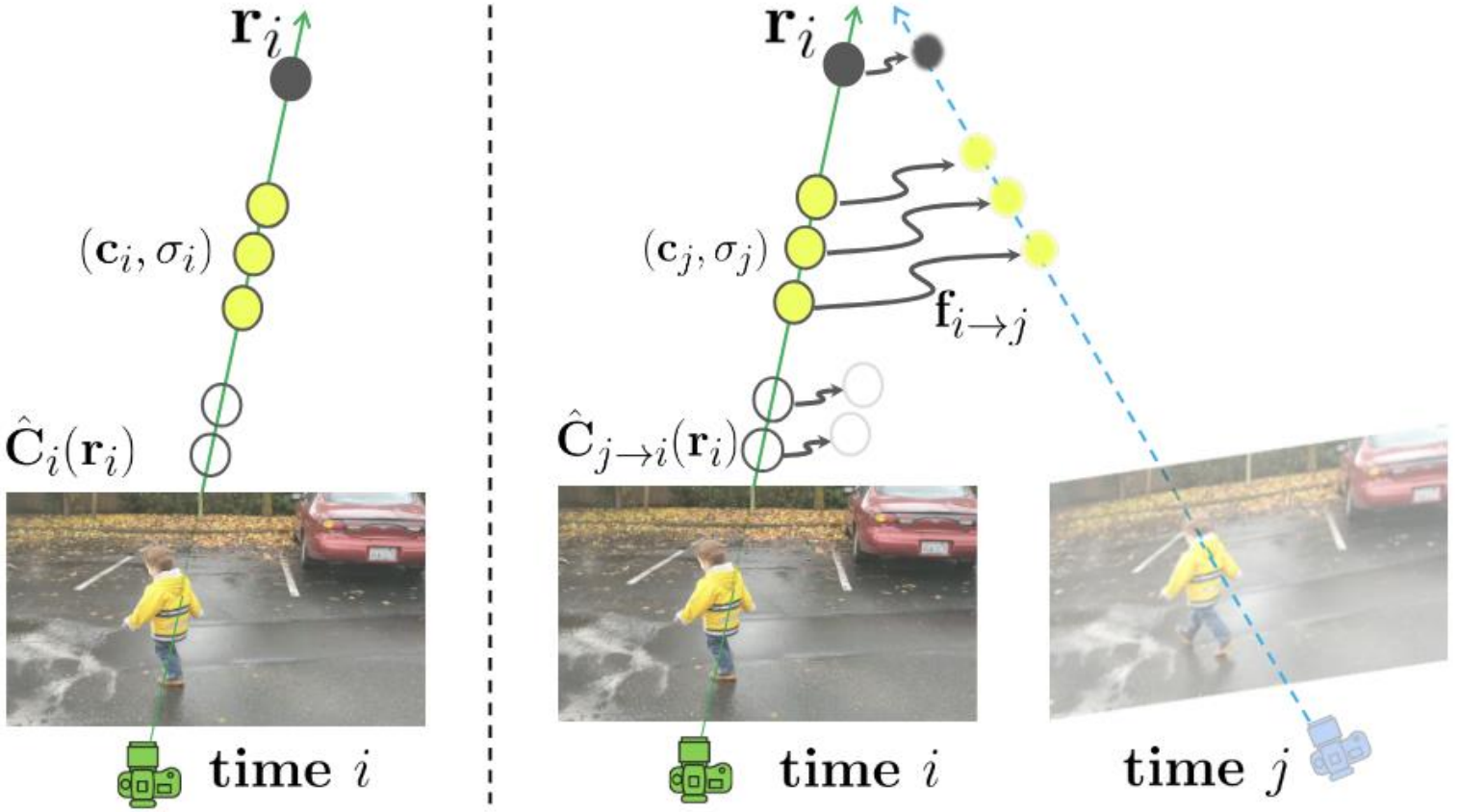


$$C(x,\omega,t) \approx \sum_{i=1}^{N}\left(\prod_{j=1}^{i-1} e^{-\sigma_{j+\Delta f_j^t}\Delta l_j}\right)\left(1-e^{-\sigma_{i+\Delta f_i^t}\Delta l_i}\right)C_{i+\Delta f_i^t}$$

Learning objective: $\min_{\sigma_i, C_i} \| C(x,\omega,t) - \text{g.t.} \|$

Neural Scene Flow Fields for Space-Time View Synthesis of Dynamic Scenes

- To render a dynamic scene, we need to model the appearance $(\boldsymbol{c_i}, \boldsymbol{\sigma_i})$ and the motion $(\boldsymbol{f_{i\to j}})$ of the scene
- Define $\boldsymbol{f_{i\to j}(\mathbf{x})}$ as the 3D offset vector moving position $\mathbf{x}$ from time $\boldsymbol{i}$ to time $\boldsymbol{j}$



 Neural Scene Flow Fields for Space-Time View Synthesis of Dynamic Scenes. Li et al.

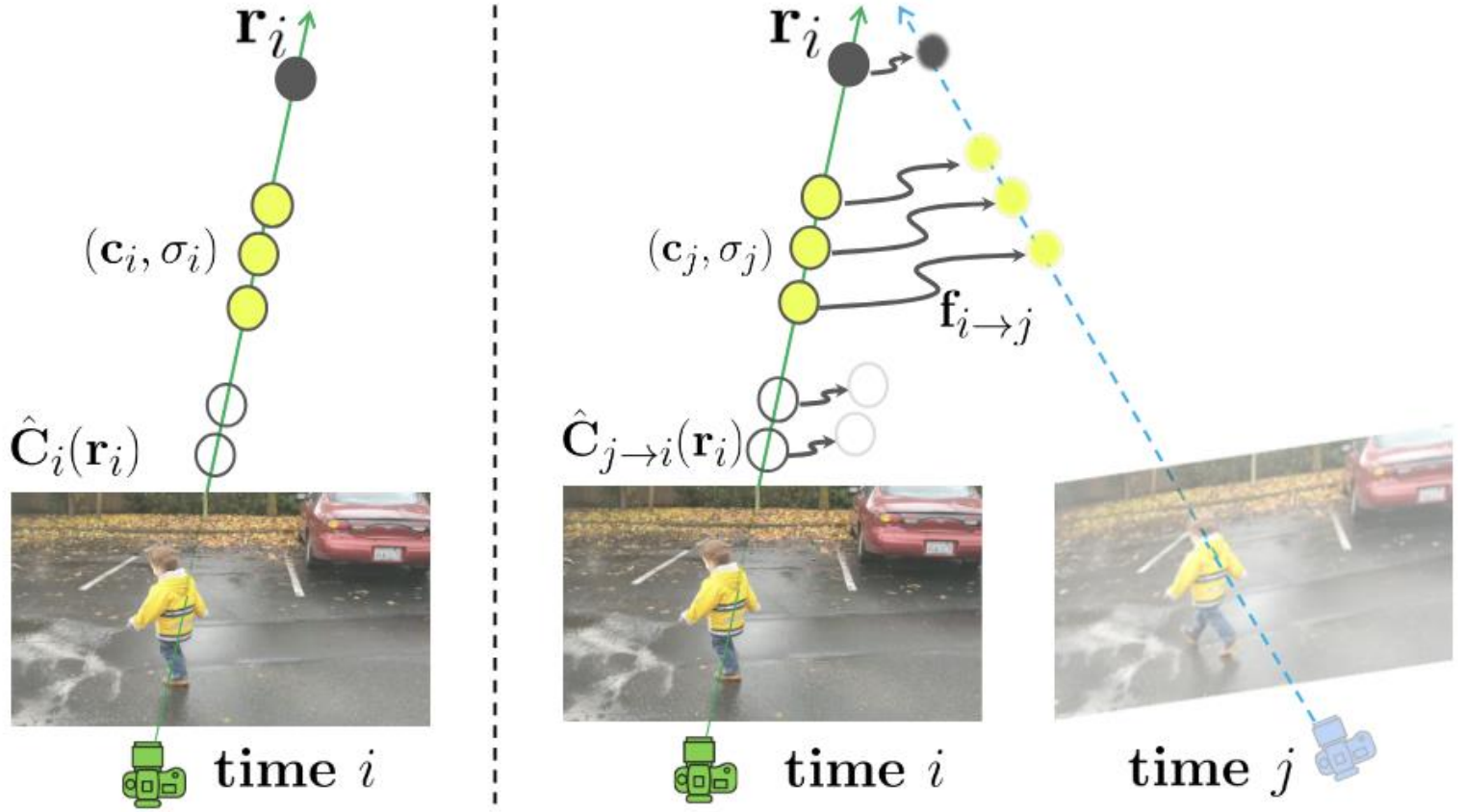
Scene Flow Field Warping

- **Idea**: We can render an object at the current time i, if we look up the scene created at time j

$$\hat{\mathbf{C}}_{j\to i}(\mathbf{r}_i) = \int_{t_n}^{t_f} T_j(t)\,\sigma_j(\mathbf{r}_{i\to j}(t))\,\mathbf{c}_j(\mathbf{r}_{i\to j}(t),\mathbf{d}_i)dt$$

$$\text{where } \boxed{\mathbf{r}_{i\to j}(t)} = \mathbf{r}_i(t) + \mathbf{f}_{i\to j}(\mathbf{r}_i(t)).$$

The new position at time j for the position $\mathbf{x}$ from time i



 Neural Scene Flow Fields for Space-Time View Synthesis of Dynamic Scenes. Li et al.

Scene Flow Field Warping

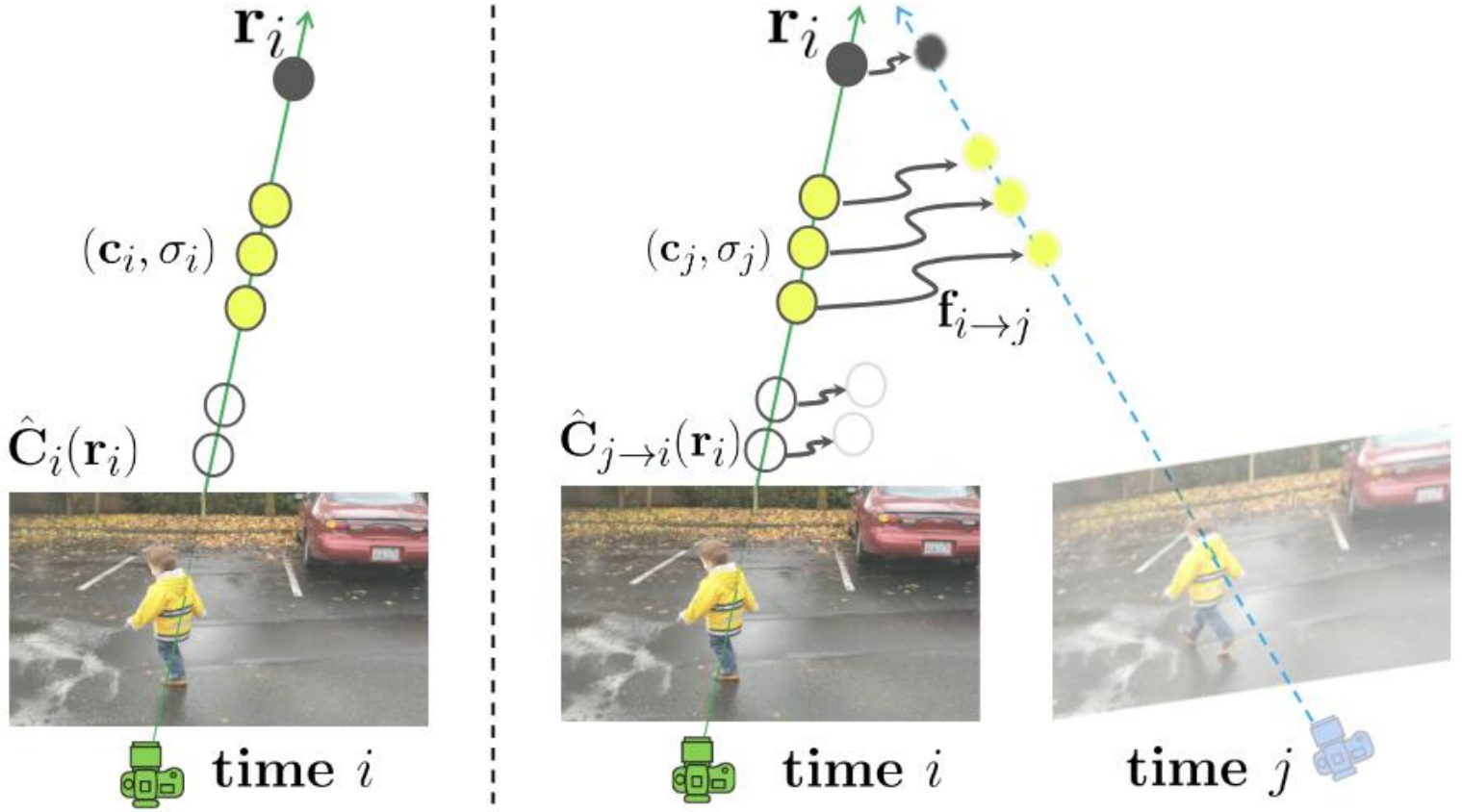
- **Idea**: We can render an object at the current time i, if we look up the scene created at time j

$$\hat{\mathbf{C}}_{j\to i}(\mathbf{r}_i) = \int_{t_n}^{t_f} T_j(t)\,\sigma_j(\mathbf{r}_{i\to j}(t))\,\mathbf{c}_j(\mathbf{r}_{i\to j}(t),\mathbf{d}_i)dt$$

$$\text{where } \mathbf{r}_{i\to j}(t) = \mathbf{r}_i(t) + \mathbf{f}_{i\to j}(\mathbf{r}_i(t)).$$

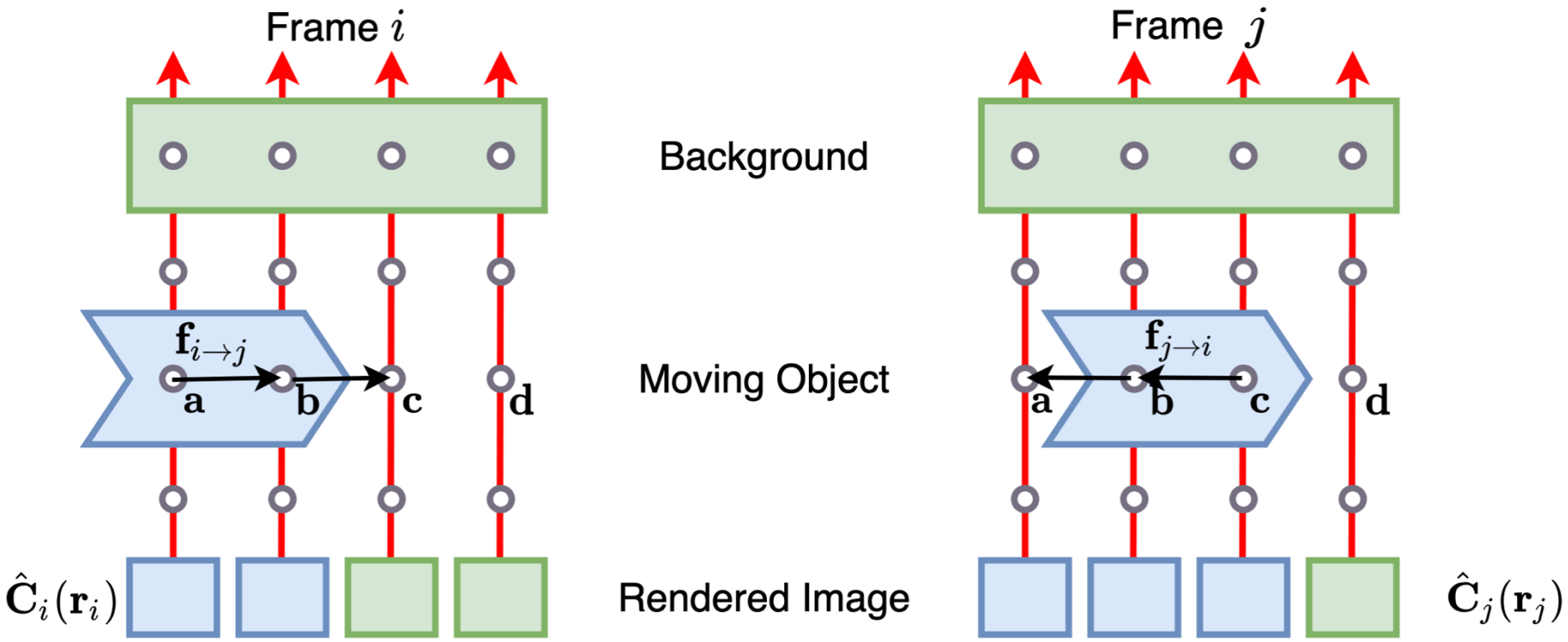
- Optimize appearance and motion by rendering:

$$\mathcal{L}_{\text{pho}} = \sum_{\mathbf{r}_i} \sum_{j\in\mathcal{N}(i)} ||\hat{\mathbf{C}}_{j\to i}(\mathbf{r}_i) - \mathbf{C}_i(\mathbf{r}_i)||_2^2$$



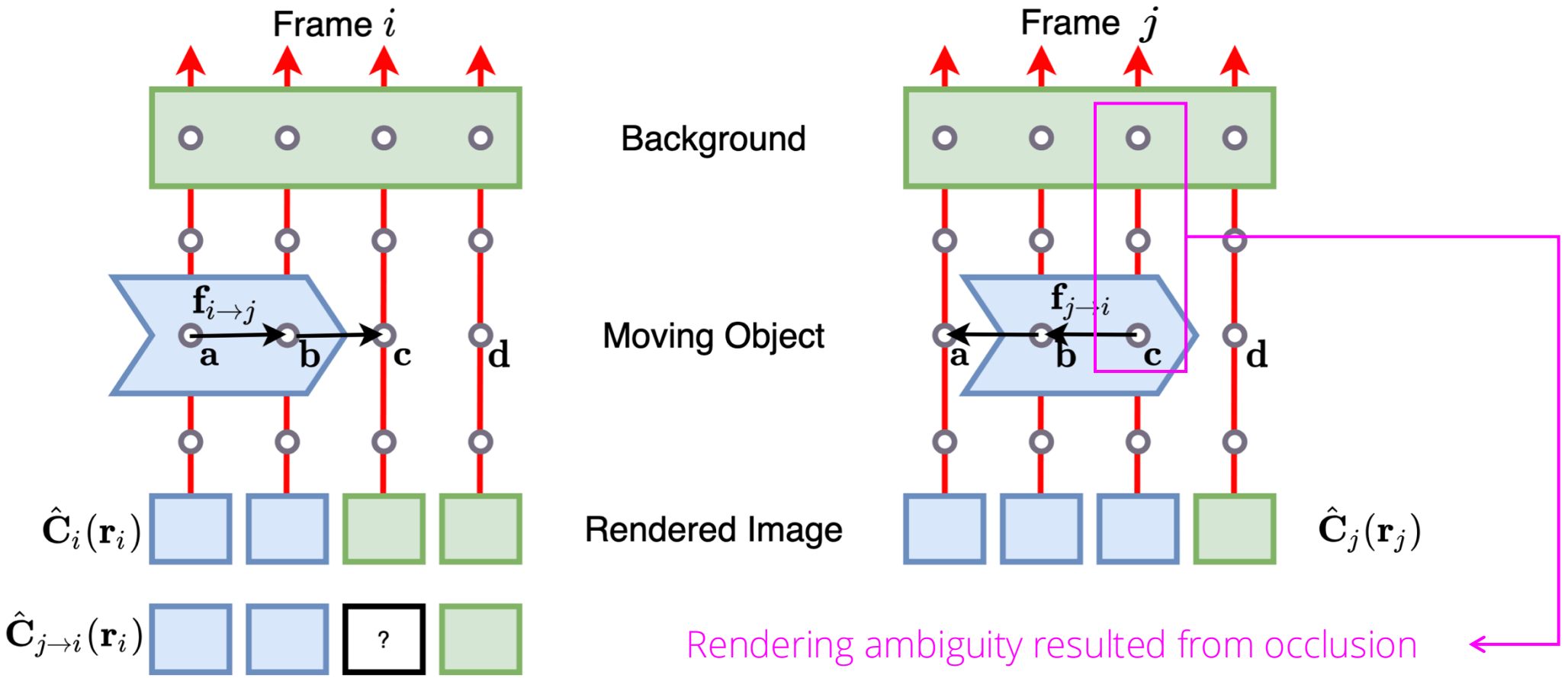
 Neural Scene Flow Fields for Space-Time View Synthesis of Dynamic Scenes. Li et al.

Rendering Loss is Ambiguous with Occlusion



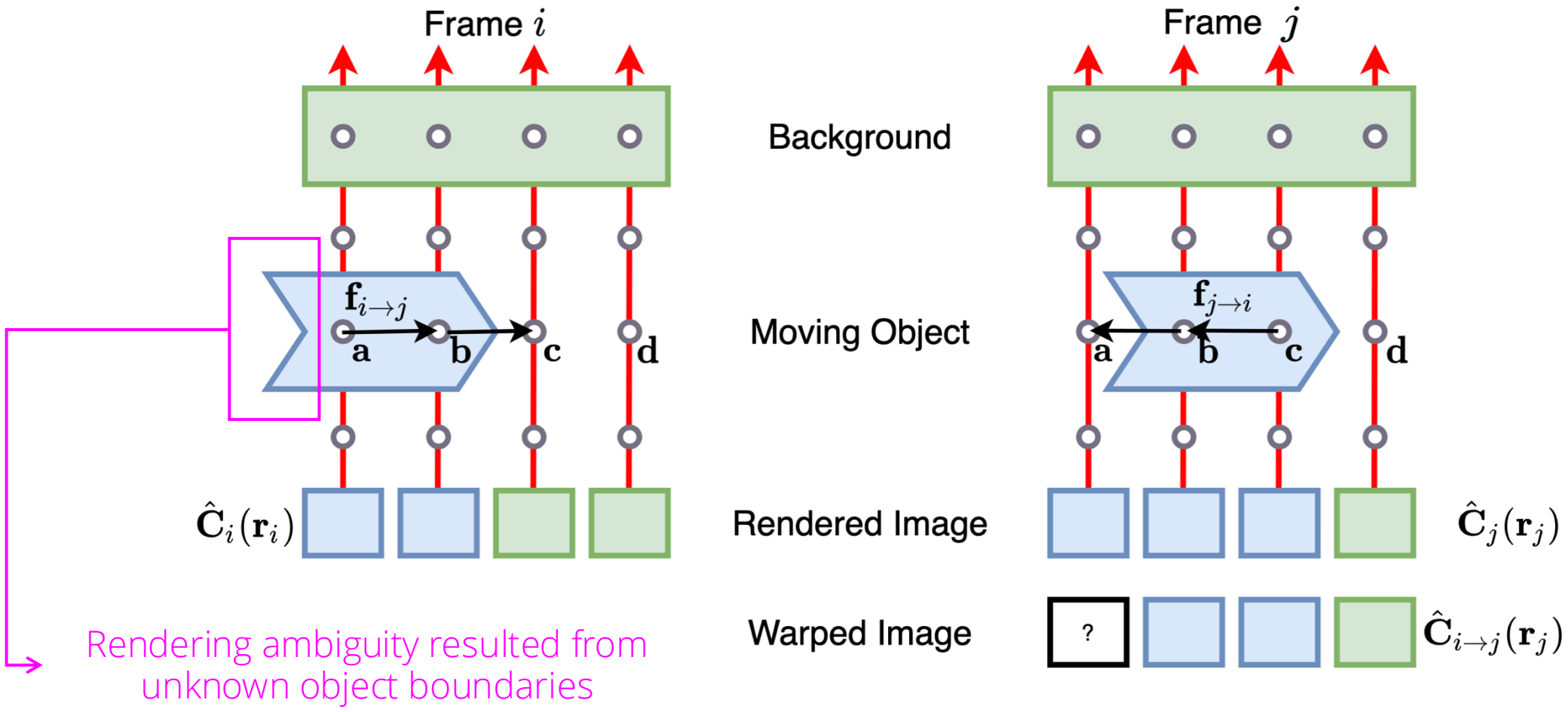
 Neural Scene Flow Fields for Space-Time View Synthesis of Dynamic Scenes. Li et al.

Rendering Loss is Ambiguous with Occlusion



 Neural Scene Flow Fields for Space-Time View Synthesis of Dynamic Scenes. Li et al.

Rendering Loss is Ambiguous with Occlusion



 Neural Scene Flow Fields for Space-Time View Synthesis of Dynamic Scenes. Li et al.

Rendering Loss is Ambiguous with Occlusion

- Solution:
  - Predict if the occlusion of points from time i to j: occlusion weight field $w_{i\to j} \in [0, 1]$
  - The occlusion weight indicates where and how much strength the temporal photo- consistency loss should be applied
  - Aggregate occlusion weights along a ray:

$$\hat{W}_{j\to i}(\mathbf{r}_i) = \int_{t_n}^{t_f} T_j(t)\, \sigma_j(\mathbf{r}_{i\to j}(t))\, w_{i\to j}(\mathbf{r}_i(t))\; dt$$

$$\mathcal{L}_{\text{pho}} = \sum_{\mathbf{r}_i} \sum_{j\in\mathcal{N}(i)} \hat{W}_{j\to i}(\mathbf{r}_i) ||\hat{\mathbf{C}}_{j\to i}(\mathbf{r}_i) - \mathbf{C}_i(\mathbf{r}_i)||_2^2$$

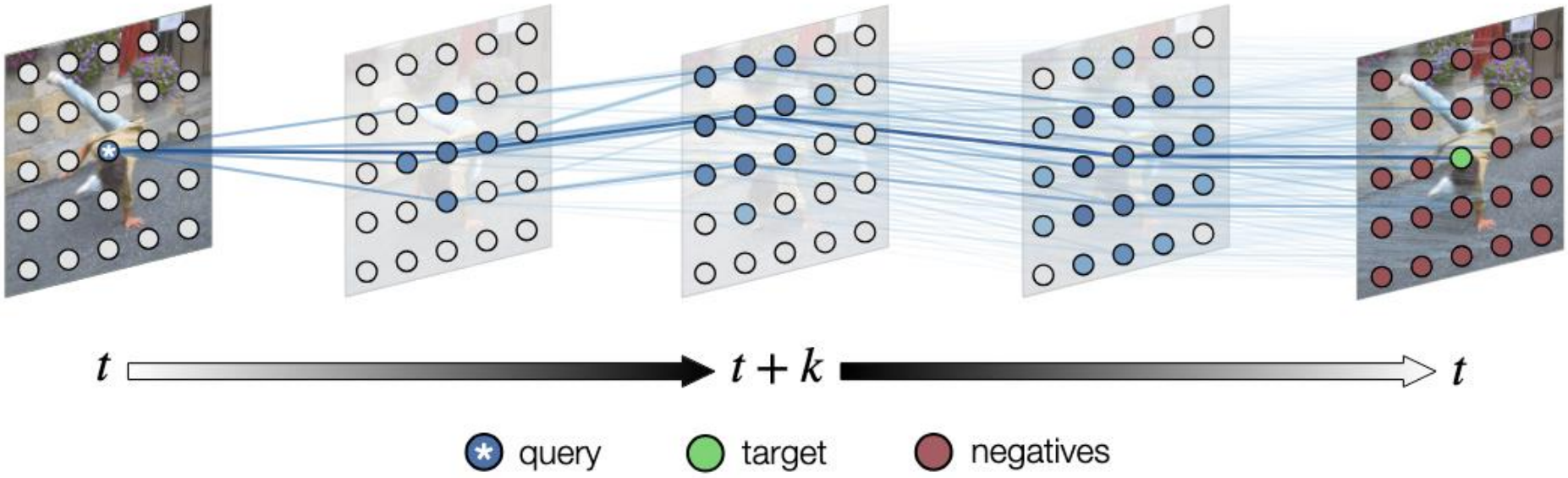
$$+ \beta_w \sum_{\mathbf{x}_i} ||| w_{i\to j}(\mathbf{x}_i) - 1 |||_1,$$

Neural Scene Flow Fields for Space-Time View Synthesis of Dynamic Scenes. Li et al.

Accurate Flow is Difficult to Learn

- Solution:
  - Cycle consistency constraint: a point should stay at the same location when moving forward and backward

$$\mathcal{L}_{\text{cyc}} = \sum_{\mathbf{x}_i} \sum_{j \in i \pm 1} w_{i \to j} ||\mathbf{f}_{i \to j}(\mathbf{x}_i) + \mathbf{f}_{j \to i}(\mathbf{x}_{i \to j})||_1$$



Space-Time Correspondence as a Contrastive Random Walk. Jabri et al.

Accurate Flow is Difficult to Learn

- **Solution**: Data-driven priors
  - ➢ Guidance with predicted 2D optical flow

$$\mathcal{L}_{\text{geo}} = \sum_{\mathbf{r}_i} \sum_{j \in \{i \pm 1\}} ||\hat{\mathbf{p}}_{i \to j}(\mathbf{r}_i) - \mathbf{p}_{i \to j}(\mathbf{r}_i))||_1 .$$

$\hat{p}_{i \to j}$ denotes the perspective projection of the expected 3D point location displaced by the scene flow

  - ➢ Guidance with predicted depth

$$\mathcal{L}_z = \sum_{\mathbf{r}_i} ||\hat{Z}_i^*(\mathbf{r}_i) - Z_i^*(\mathbf{r}_i)||_1$$

$\hat{Z}_i$ denotes the expected termination depth computed along a ray (check previous slides)

Space-time Neural Irradiance Fields for Free-Viewpoint Video. Xian et al.

Accurate Flow is Difficult to Learn

- **Solution**: Decouple static and dynamic scene rendering
  - Create a static and dynamic radiance field
  - Blend both fields by factor v

$$\hat{\mathbf{C}}_i^{\text{cb}}(\mathbf{r}_i) = \int_{t_n}^{t_f} T_i^{\text{cb}}(t)\, \sigma_i^{\text{cb}}(t)\, \mathbf{c}_i^{\text{cb}}(t) dt,$$

$$\sigma_i^{\text{cb}}(t)\, \mathbf{c}_i^{\text{cb}}(t) =$$

$$v(t)\, \mathbf{c}(t)\, \sigma(t) + (1\text{-}v(t))\, \mathbf{c}_i(t)\, \sigma_i(t)$$

$$\mathcal{L}_{\text{cb}} = \sum_{\mathbf{r}_i} ||\hat{\mathbf{C}}_i^{\text{cb}}(\mathbf{r}_i) - \mathbf{C}_i(\mathbf{r}_i)||_2^2.$$

Combined render | Static only | Dynamic only

Space-time Neural Irradiance Fields for Free-Viewpoint Video. Xian et al.

Results

Space-time Neural Irradiance Fields for Free-Viewpoint Video. Xian et al.

Input frames

Fixed Time, View Interpolation

Fixed View, Time Interpolation

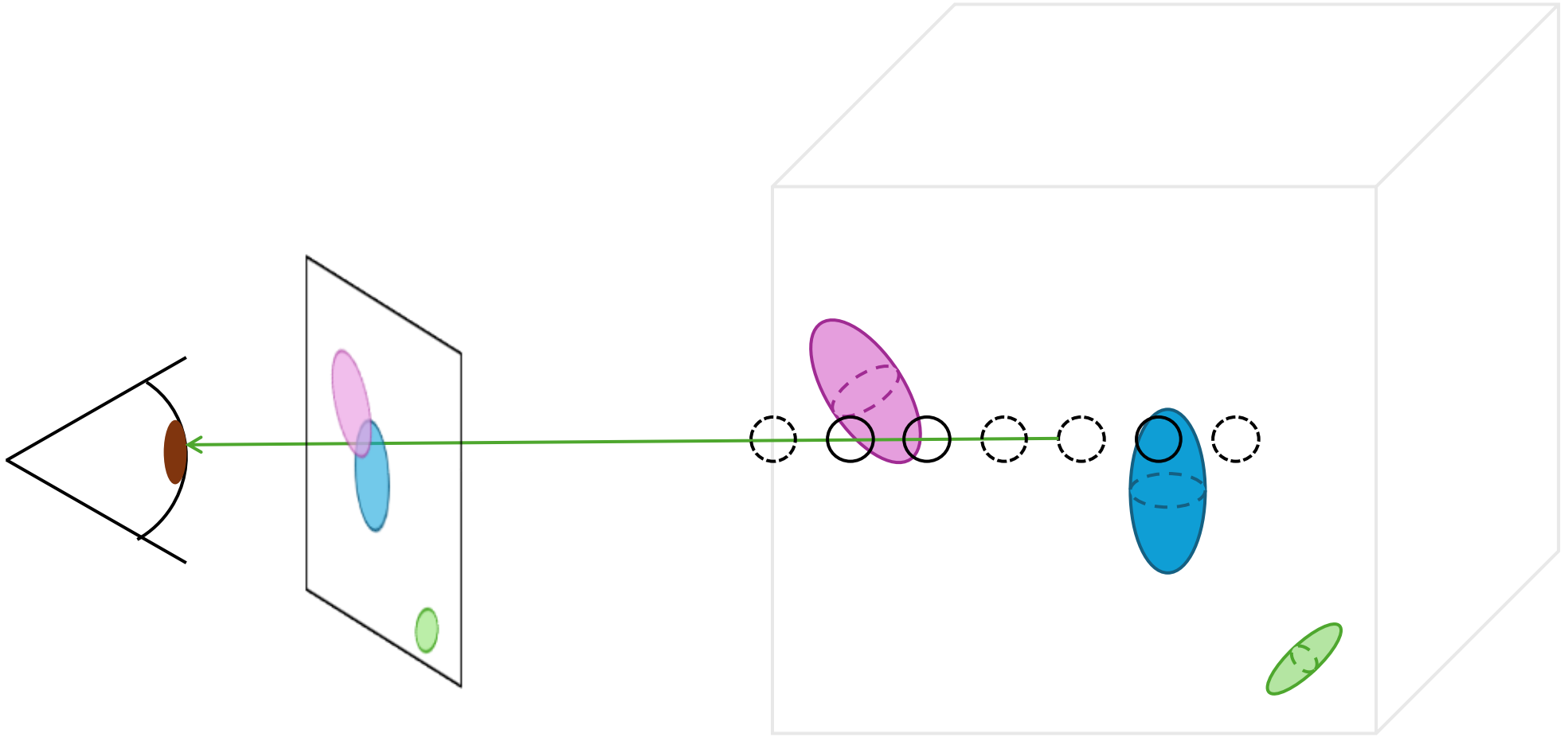
Space-Time Interpolation

Space-time Neural Irradiance Fields for Free-Viewpoint Video. Xian et al.

Content

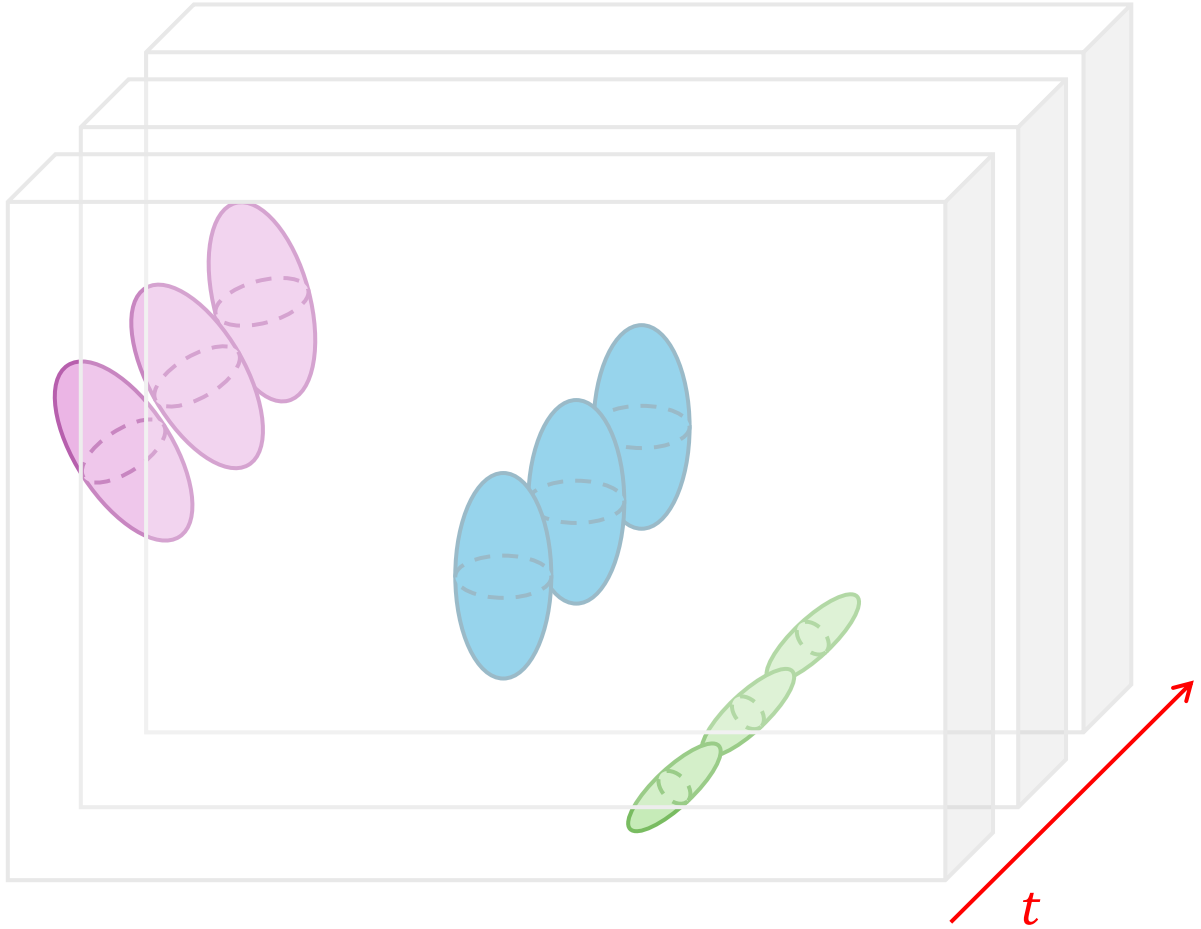
- 2D Motion Representations
- 3D Motion Representations
  - ➢4D Volume Rendering with NERF
  - ➢Explicit Motion Field with NERF
  - ➢4D Volume Rendering with 3DGS
  - ➢Explicit Motion Field with 3DGS
  - ➢Implicit Motion Field with 3DGS
  - ➢4D Foundation Models
  - ➢Multi-view Video Generation (we'll cover in the next lecture)

Volume Rendering with 3D Gaussian Splatting

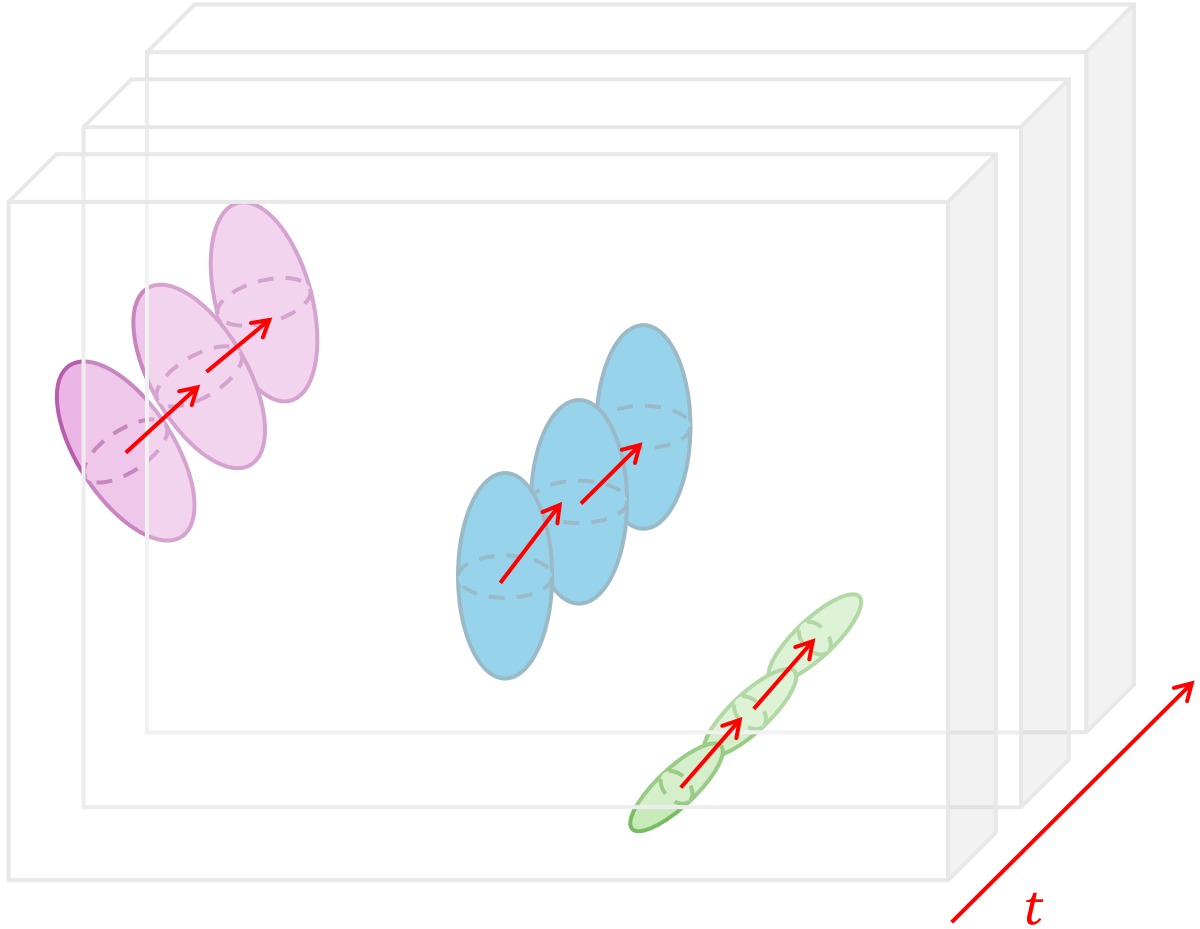


Slide adapted from V. Sitzmann

Two Ways of 4D Volume Rendering



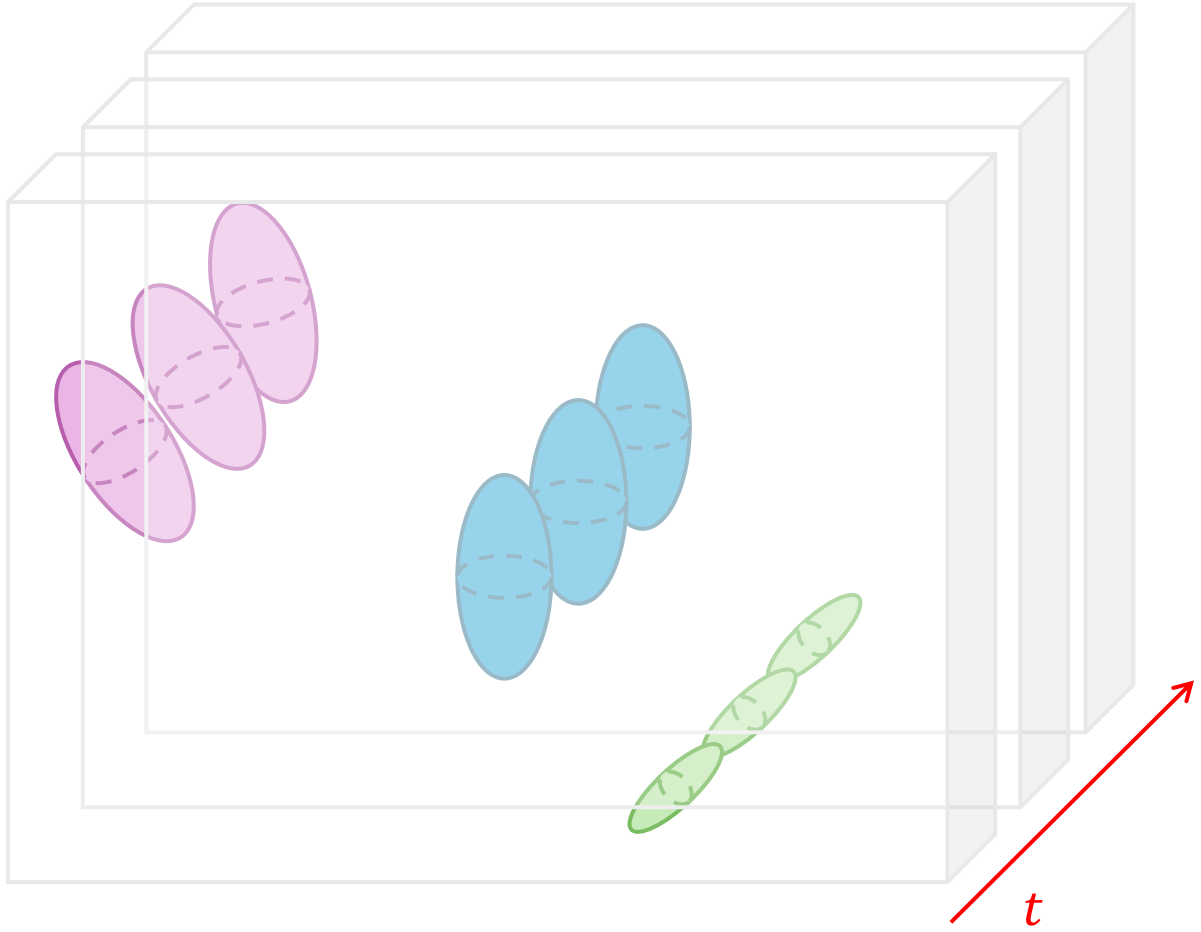
4DGS: Represent a dynamic scene with a set of 4D GS



Dynamic 3DGS: Represent a dynamic scene with a set of 3D GS and their motions

Method 1: Extend 3DGS to 4DGS

- **Idea**: Generalize 3D GS to a 4D formulation.
- **How to render**? Slice a 4D GS into a 3D GS at time t, and render it with 2D rasterization



4D-Rotor Gaussian Splatting: Towards Efficient Novel View Synthesis for Dynamic Scenes. Duan et al.

A Simplified Illustration for Slicing a 4D GS into a 3D GS

- **Idea**: Generalize 3D GS to a 4D formulation.
- **How to render**? Slice a 4D GS into a 3D GS at time t, and render it with 2D rasterization
- **What does it mean**? In a small interval, all motions can be approximated by linear motions, and more complex non-linear cases can be represented by combination of multiple Gaussians

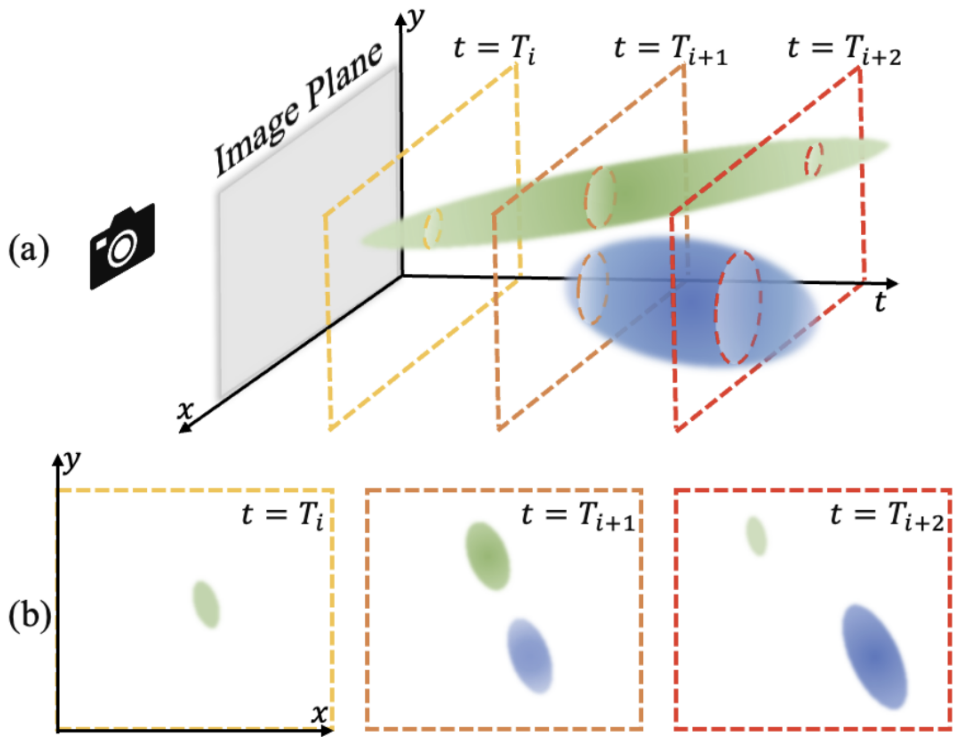


Figure 2: A Simplified 2D Illustration of the Proposed Temporal Slicing. (a) We model 2D dynamics with 3D XYT ellipsoids and slice them with different time queries. (b) The sliced 3D ellipsoids form 2D dynamic ellipses at each timestamp.

4D-Rotor Gaussian Splatting: Towards Efficient Novel View Synthesis for Dynamic Scenes. Duan et al.

Temporally Sliced 4DGS is a 3DGS

- Formulation:

$$G_{4D}(\mathbf{x}) = e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_{4D})^T \Sigma_{4D}^{-1}(\mathbf{x}-\boldsymbol{\mu}_{4D})}$$

- A 4D center position: $\boldsymbol{\mu}_{4D} = (\mu_x, \mu_y, \mu_z, \mu_t)$

- A 4D covariance matrix: $\boldsymbol{\Sigma}_{4D} = \mathbf{R}_{4D}\mathbf{S}_{4D}\mathbf{S}_{4D}^T\mathbf{R}_{4D}^T$

- S_{4D} denotes a diagonal 4D matrix for scaling
- R_{4D} denotes a 4D rotator matrix (a 4D GS reduces to a 3D GS if the last for components of R_{4D} are set to 0)

Generalize to both static and dynamic scene!

4D-Rotor Gaussian Splatting: Towards Efficient Novel View Synthesis for Dynamic Scenes. Duan et al.

Temporally Sliced 4DGS is a 3DGS

- We can write the 4D covariance matrix as:

$$\Sigma_{4D} = \begin{pmatrix} \mathbf{U} & \mathbf{V} \\ \mathbf{V}^T & \mathbf{W} \end{pmatrix} \text{ and } \Sigma_{4D}^{-1} = \begin{pmatrix} \mathbf{A} & \mathbf{M} \\ \mathbf{M}^T & \mathbf{Z} \end{pmatrix}$$

- A temporally sliced 4DGS is a 3DGS

$$G_{3D}(\mathbf{x}, t) = \mathbf{e}^{-\frac{1}{2}\lambda(t-\mu_t)^2} \mathbf{e}^{-\frac{1}{2}[\mathbf{x}-\boldsymbol{\mu}(t)]^T \Sigma_{3D}^{-1} [\mathbf{x}-\boldsymbol{\mu}(t)]},$$

$$\lambda = \mathbf{W}^{-1}, \qquad \Sigma_{3D} = \mathbf{A}^{-1} = \mathbf{U} - \frac{\mathbf{V}\mathbf{V}^T}{\mathbf{W}}, \qquad \boldsymbol{\mu}(t) = (\mu_x, \mu_y, \mu_z)^T + (t - \mu_t)\frac{\mathbf{V}}{\mathbf{W}}.$$

4D-Rotor Gaussian Splatting: Towards Efficient Novel View Synthesis for Dynamic Scenes. Duan et al.

Let's Take a Closer Look

- A temporally sliced 4DGS is a 3DGS

$$G_{3D}(\mathbf{x}, t) = \mathbf{e}^{-\frac{1}{2}\lambda(t-\mu_t)^2} \mathbf{e}^{-\frac{1}{2}[\mathbf{x}-\boldsymbol{\mu}(t)]^T \boldsymbol{\Sigma}_{3D}^{-1} [\mathbf{x}-\boldsymbol{\mu}(t)]},$$

$$\lambda = \mathbf{W}^{-1},$$

$$\boldsymbol{\mu}(t) = (\mu_x, \mu_y, \mu_z)^T + (t - \mu_t)\frac{\mathbf{V}}{\mathbf{W}}.$$

- As t goes by, a Gaussian point first appears once t is sufficiently close to its temporal position μ_t and starts to grow. It reaches the peak opacity when $\mathbf{t} = \mu_t$
- By controlling the temporal position and scaling factor, 4D Gaussian can represent challenging dynamics, e.g., motions that suddenly appear or disappear

4D-Rotor Gaussian Splatting: Towards Efficient Novel View Synthesis for Dynamic Scenes. Duan et al.

Let's Take a Closer Look

- A temporally sliced 4DGS is a 3DGS

$$G_{3D}(\mathbf{x}, t) = \mathbf{e}^{-\frac{1}{2}\lambda(t-\mu_t)^2}\mathbf{e}^{-\frac{1}{2}[\mathbf{x}-\boldsymbol{\mu}(t)]^T\boldsymbol{\Sigma}_{3D}^{-1}[\mathbf{x}-\boldsymbol{\mu}(t)]},$$

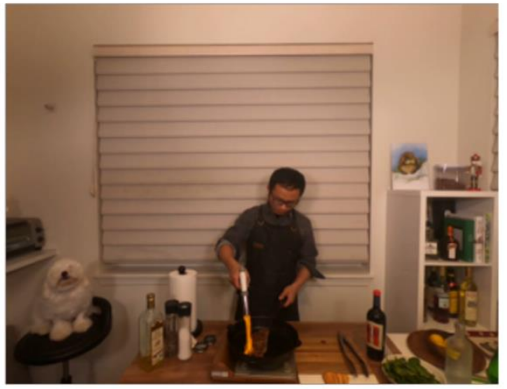
$$\boldsymbol{\mu}(t) = (\mu_x, \mu_y, \mu_z)^T + (t-\mu_t)\frac{\mathbf{V}}{\mathbf{W}}.$$

- the sliced 3D Gaussian exhibits a new motion term of $(t-\mu_t)\frac{v}{w}$ added to the center position $(\mu_x, \mu_y, \mu_z)^\intercal$
- $\frac{v}{w}$ indicates the motion speed in the current timestamp
- By modeling a scene with 4D-Rotator GS, we can acquire speed field for free

4D-Rotor Gaussian Splatting: Towards Efficient Novel View Synthesis for Dynamic Scenes. Duan et al.

Emerging Optical Flow from 4D Rendering






w/o 4D Consistency L. w/ 4D Consistency L. Ground Truth

4D-Rotor Gaussian Splatting: Towards Efficient Novel View Synthesis for Dynamic Scenes. Duan et al.

4D-Rotator Gaussian Splatting is Difficult to Learn

- Solution:
  - 4D Consistency Loss: Neighboring Gaussian points should have similar speed s

$$L_{\text{consistent4D}} = \frac{1}{N}\sum_{i=1}^{N}\left\|\mathbf{s}_i - \frac{1}{K}\sum_{j\in\Omega_i}\mathbf{s}_j\right\|_1 .$$

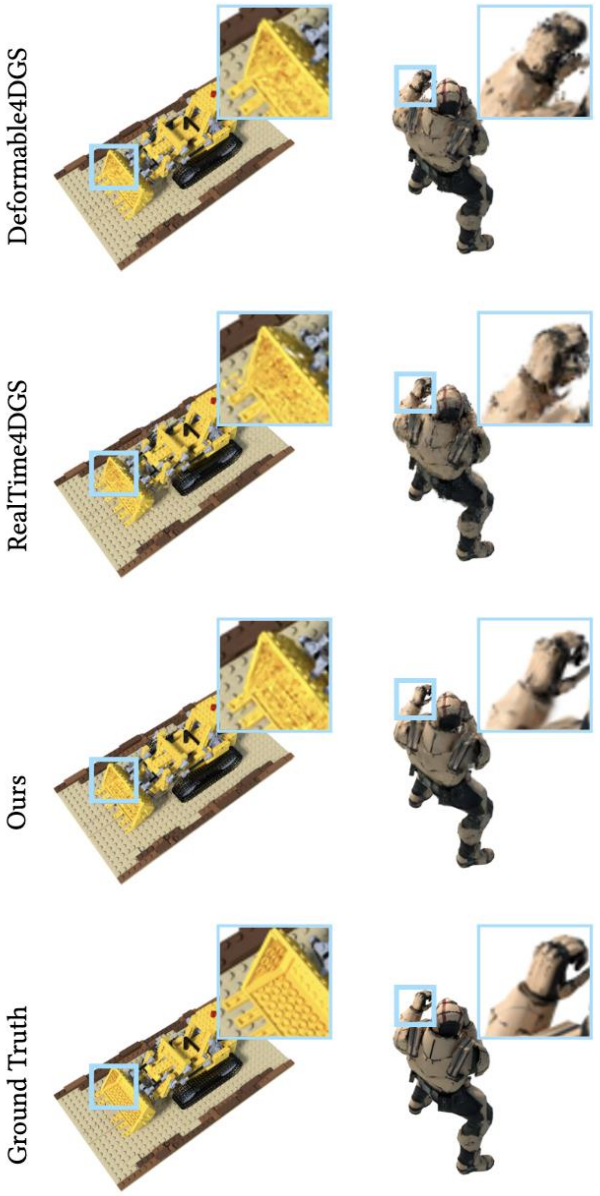
  - Entropy Loss: Remove non-surface Gaussian points by encouraging the opacity to be close to 0 or 1

$$L_{\text{entropy}} = \frac{1}{N}\sum_{i=1}^{N} -o_i log(o_i).$$

4D-Rotor Gaussian Splatting: Towards Efficient Novel View Synthesis for Dynamic Scenes. Duan et al.

Results

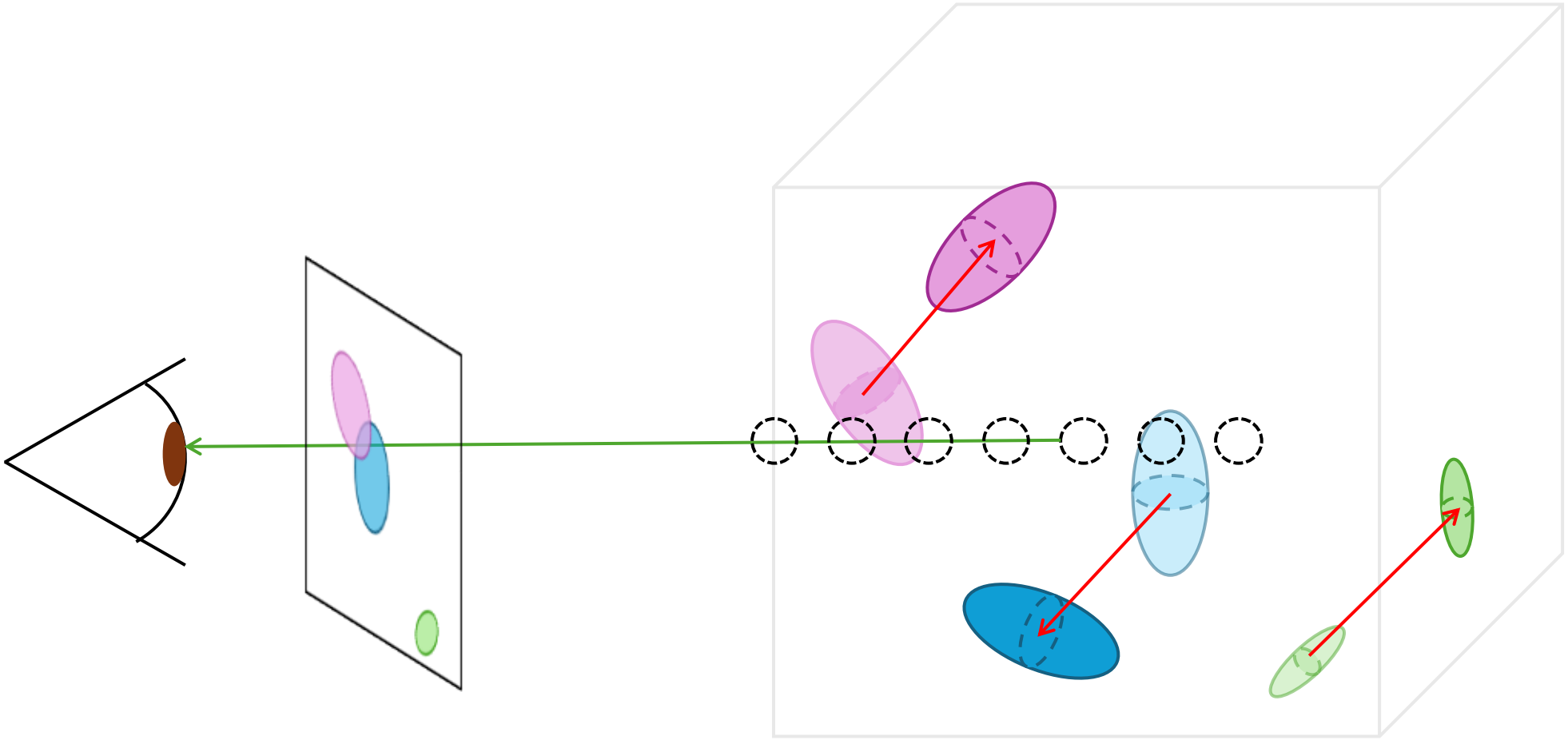
ID	Method	PSNR↑	SSIM↑	LPIPS↓	Train↓	FPS↑
a	DyNeRF [Li et al. 2022b]*†	29.58	-	0.08	1344 h**	0.015
b	StreamRF [Li et al. 2022a]	28.16	0.85	0.31	79 min	8.50
c	HyperReel [Attal et al. 2023]	30.36	0.92	0.17	9 h	2.00
d	NeRFPlayer [Song et al. 2023a]†	30.69	-	0.11	6 h	0.05
e	K-Planes [Fridovich-Keil et al. 2023]	30.73	0.93	0.07	190 min	0.10
f	MixVoxels [Wang et al. 2023b]	30.85	0.96	0.21	91 min	16.70
g	Deformable4DGS [Wu et al. 2023]	28.42	0.92	0.17	72 min	39.93
h	RealTime4DGS [Yang et al. 2024]	29.95	0.92	0.16	8 h	72.80
i	Ours	31.62	0.94	0.14	60 min	277.47



Results

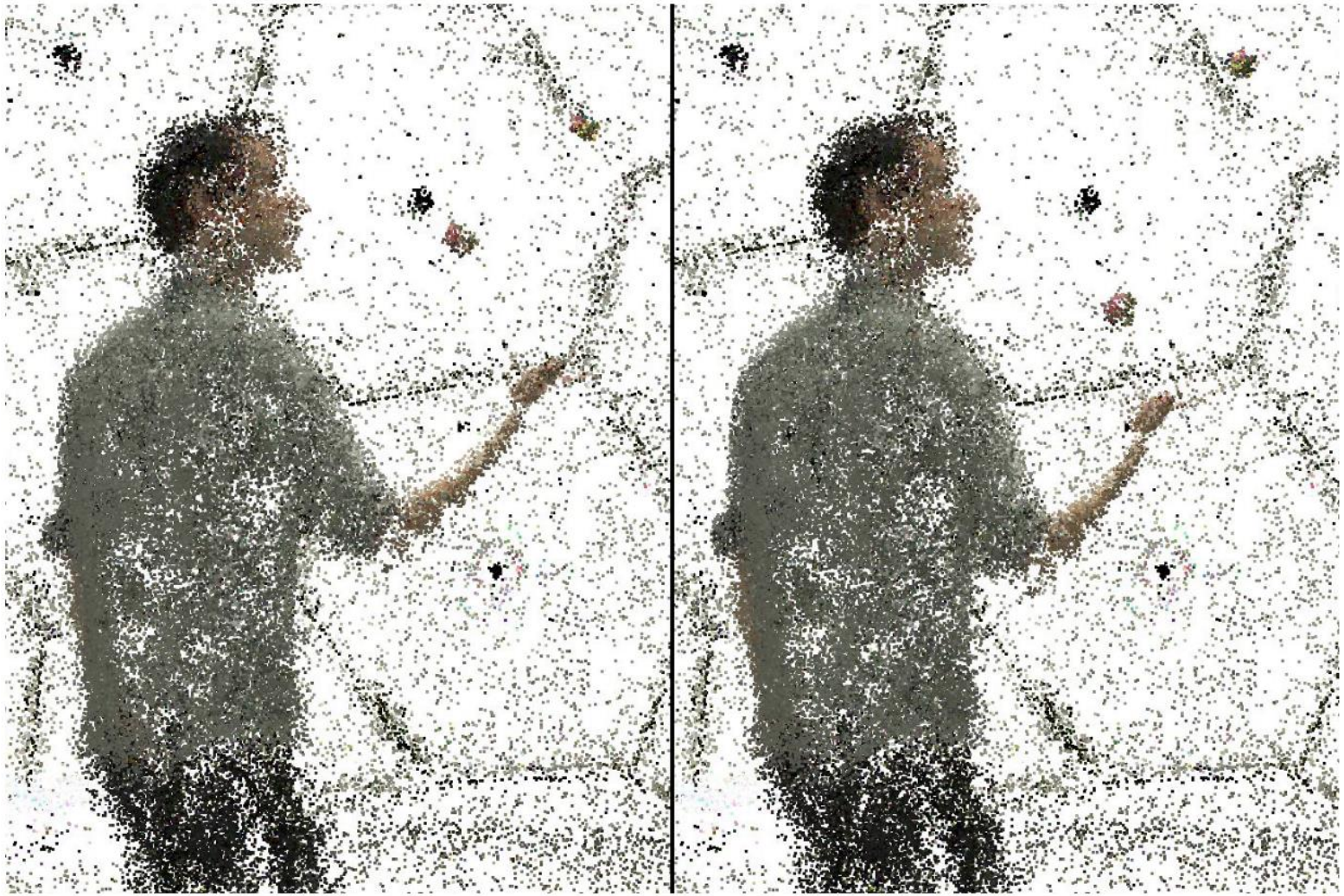
4D-Rotor Gaussian Splatting: Towards Efficient Novel View Synthesis for Dynamic Scenes. Duan et al.

Method 2: 3D GS and their Motions



Dynamic 3D Gaussians: Tracking by Persistent Dynamic View Synthesis

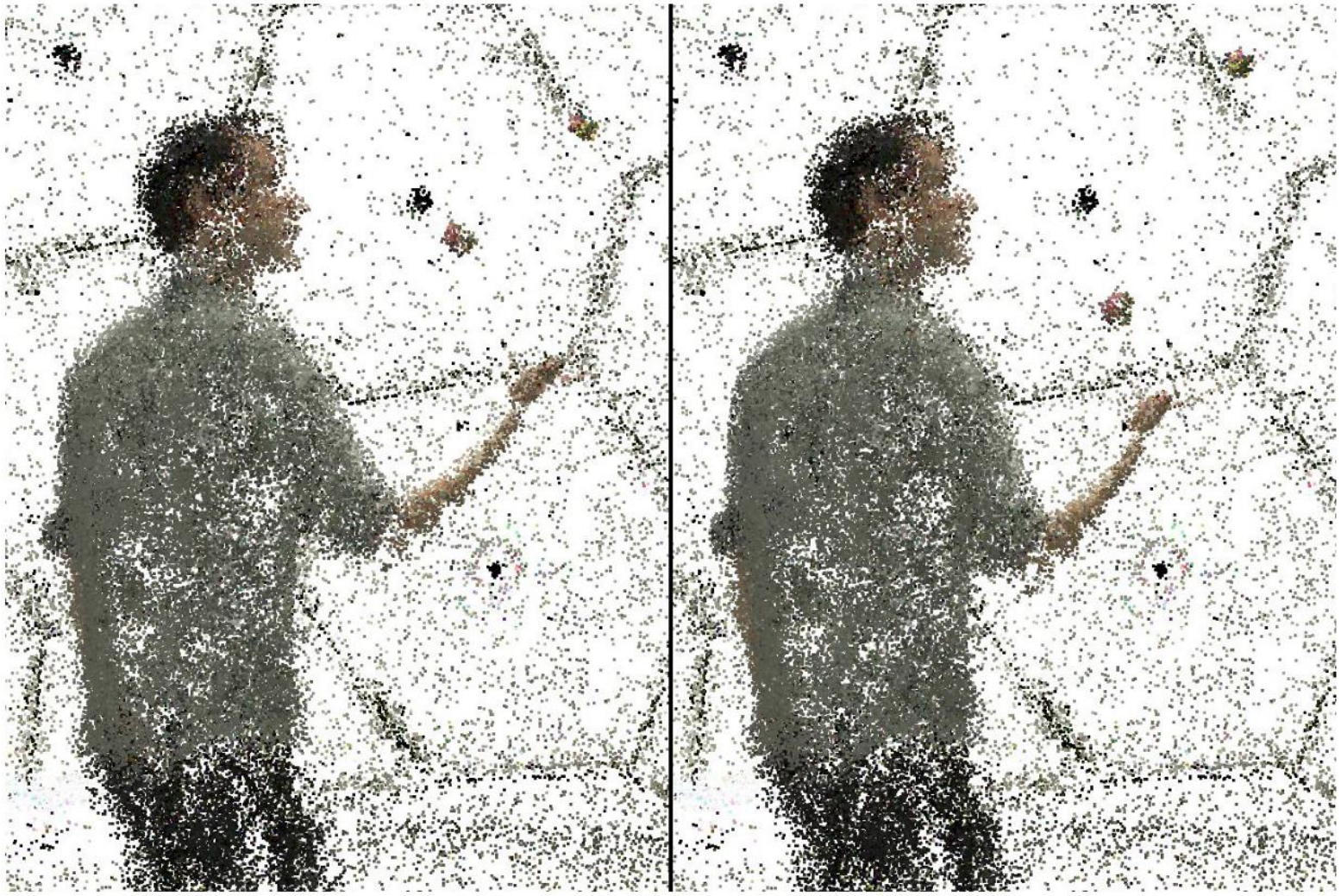
- Idea: Represent a dynamic scene with the same set of 3D Gaussians and their motions
- Particle-based dynamic perspective: Each Gaussian can be thought of as softly representing an area of 3D physical space which is occupied by solid matter



Dynamic 3D Gaussians: Tracking by Persistent Dynamic View Synthesis. Luiten et al

Dynamic 3D Gaussians: Tracking by Persistent Dynamic View Synthesis

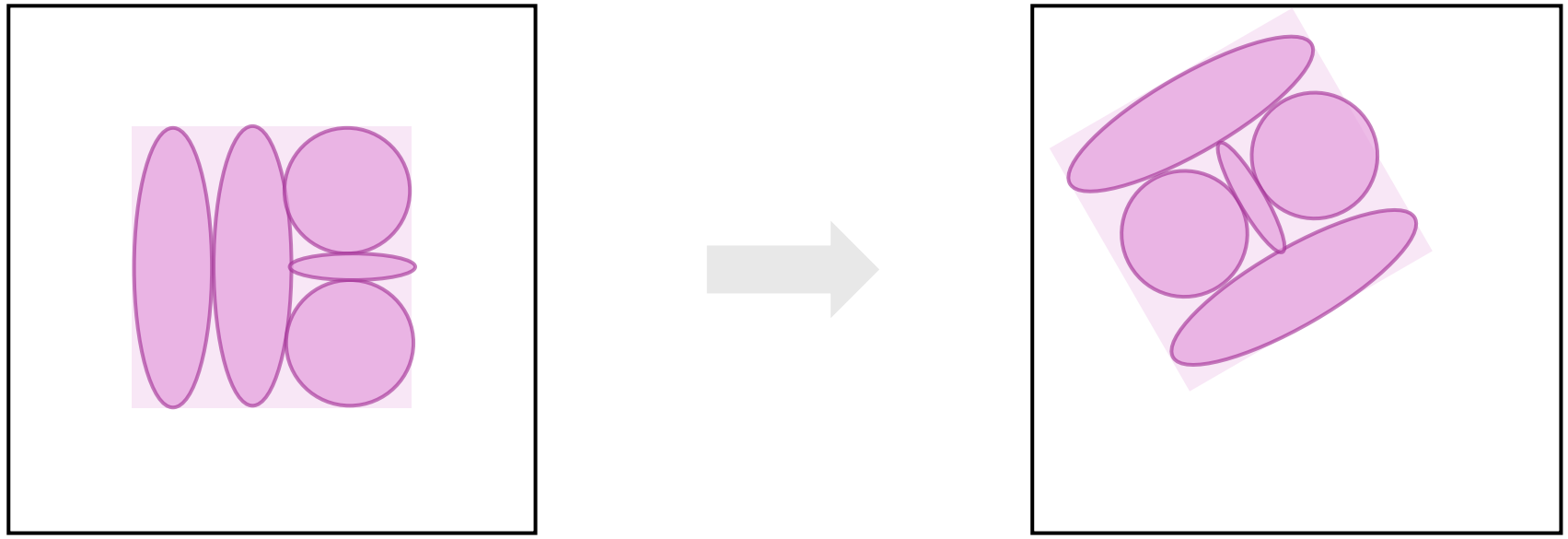
- Dynamic 3D GS:
  - A 3D center for each timestep (x_t, y_t, z_t)
  - A 3D rotation for each timestep (qw_t, qx_t, qy_t, qz_t)
  - A fixed 3D size across time (s_x, s_y, s_z)
  - A fixed color (r, g, b)
  - A fixed opacity o
  - A background indicator bg
- $7t + 8$ parameters in total



Dynamic 3D Gaussians: Tracking by Persistent Dynamic View Synthesis. Luiten et al

Dynamic 3D Gaussian Doesn’t Work Out-of-Box

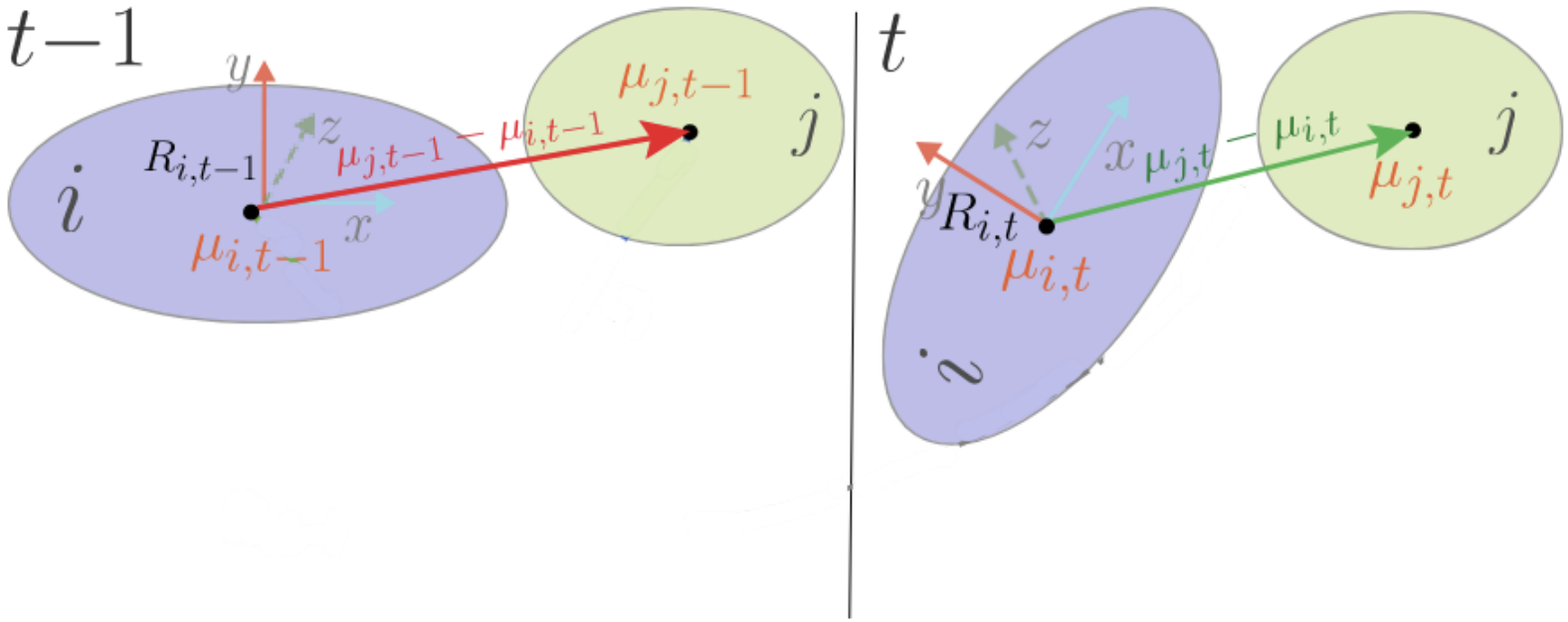
- **Problem**: Fixing color, opacity and size of Gaussians doesn’t work on long videos, especially across areas of the scene where there is a large area of near uniform color. 3D Gaussians move freely in the area of uniform color.



Dynamic 3D Gaussians: Tracking by Persistent Dynamic View Synthesis. Luiten et al

Dynamic 3D Gaussian Doesn't Work Out-of-Box

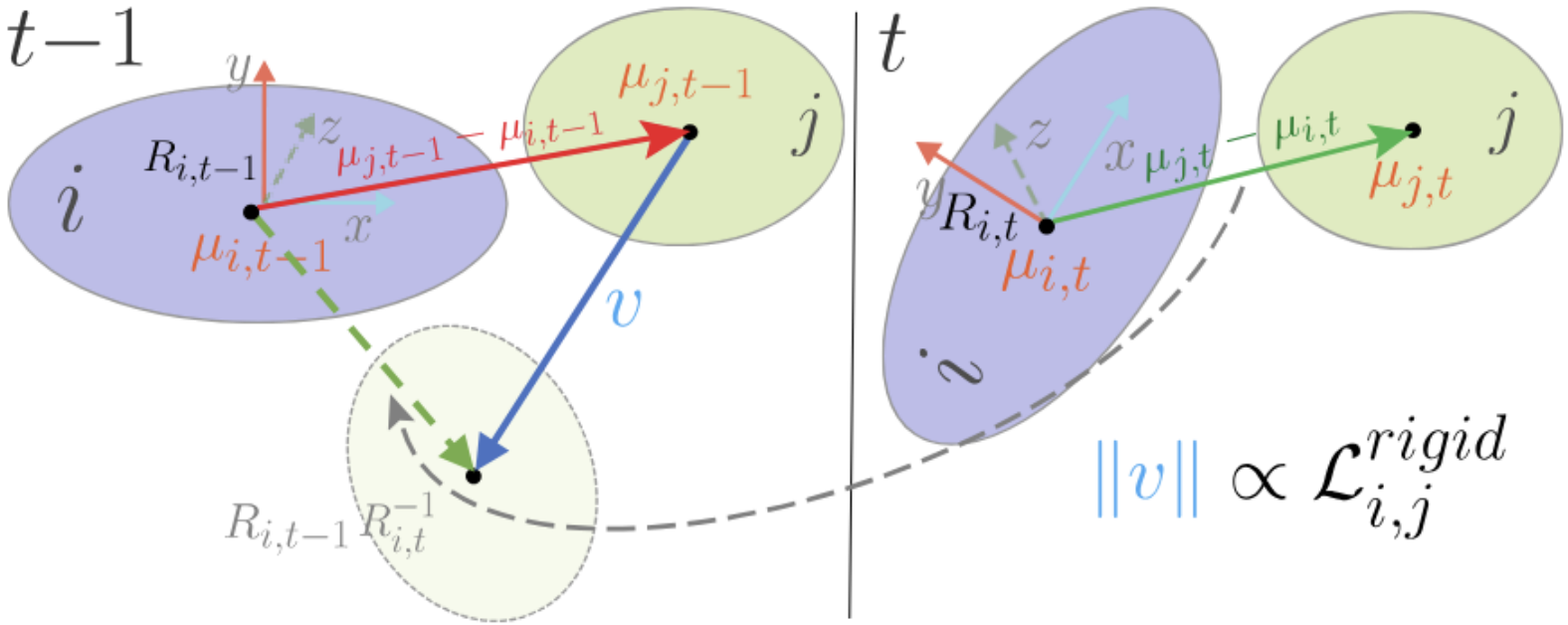
- **Solution**: Physically-Based Priors
  - ➢ Local-rigidity constraint: locally neighboring Gaussians should move like a rigid body



Dynamic 3D Gaussians: Tracking by Persistent Dynamic View Synthesis. Luiten et al

Dynamic 3D Gaussian Doesn't Work Out-of-Box

- **Solution**: Physically-Based Priors
  - ➢ Local-rigidity constraint: locally neighboring Gaussians should move like a rigid body



Dynamic 3D Gaussians: Tracking by Persistent Dynamic View Synthesis. Luiten et al

Dynamic 3D Gaussian Doesn't Work Out-of-Box

- Solution: Physically-Based Priors
  - Local-rigidity constraint: locally neighboring Gaussians should move like a rigid body

$$\mathcal{L}_{i,j}^{\text{rigid}} = w_{i,j} \left\| (\mu_{j,t-1} - \mu_{i,t-1}) - R_{i,t-1} R_{i,t}^{-1} (\mu_{j,t} - \mu_{i,t}) \right\|_2$$

$$\mathcal{L}^{\text{rigid}} = \frac{1}{k|\mathcal{S}|} \sum_{i \in \mathcal{S}} \sum_{j \in \text{knn}_{i;k}} \mathcal{L}_{i,j}^{\text{rigid}}$$

  - This loss predicted $\mu_{i,t}$ and $R_{i,t}$ to be geometrically consistent, facilitating more precise tracking

Dynamic 3D Gaussians: Tracking by Persistent Dynamic View Synthesis. Luiten et al

Learning Dynamic 3D Gaussian is Slow

- Solution:
  - Local-orientation loss: Enforce neighboring Gaussians to have the same rotation

$$\mathcal{L}^{\text{rot}} = \frac{1}{k|\mathcal{S}|} \sum_{i \in \mathcal{S}} \sum_{j \in \text{knn}_{i;k}} w_{i,j} \left\| \hat{q}_{j,t} \hat{q}_{j,t-1}^{-1} - \hat{q}_{i,t} \hat{q}_{i,t-1}^{-1} \right\|_2 \quad w_{i,j} = \exp\left(-\lambda_w \left\| \mu_{j,0} - \mu_{i,0} \right\|_2^2\right)$$

  - Local-translation loss: Enforce neighboring Gaussians to have the same translation

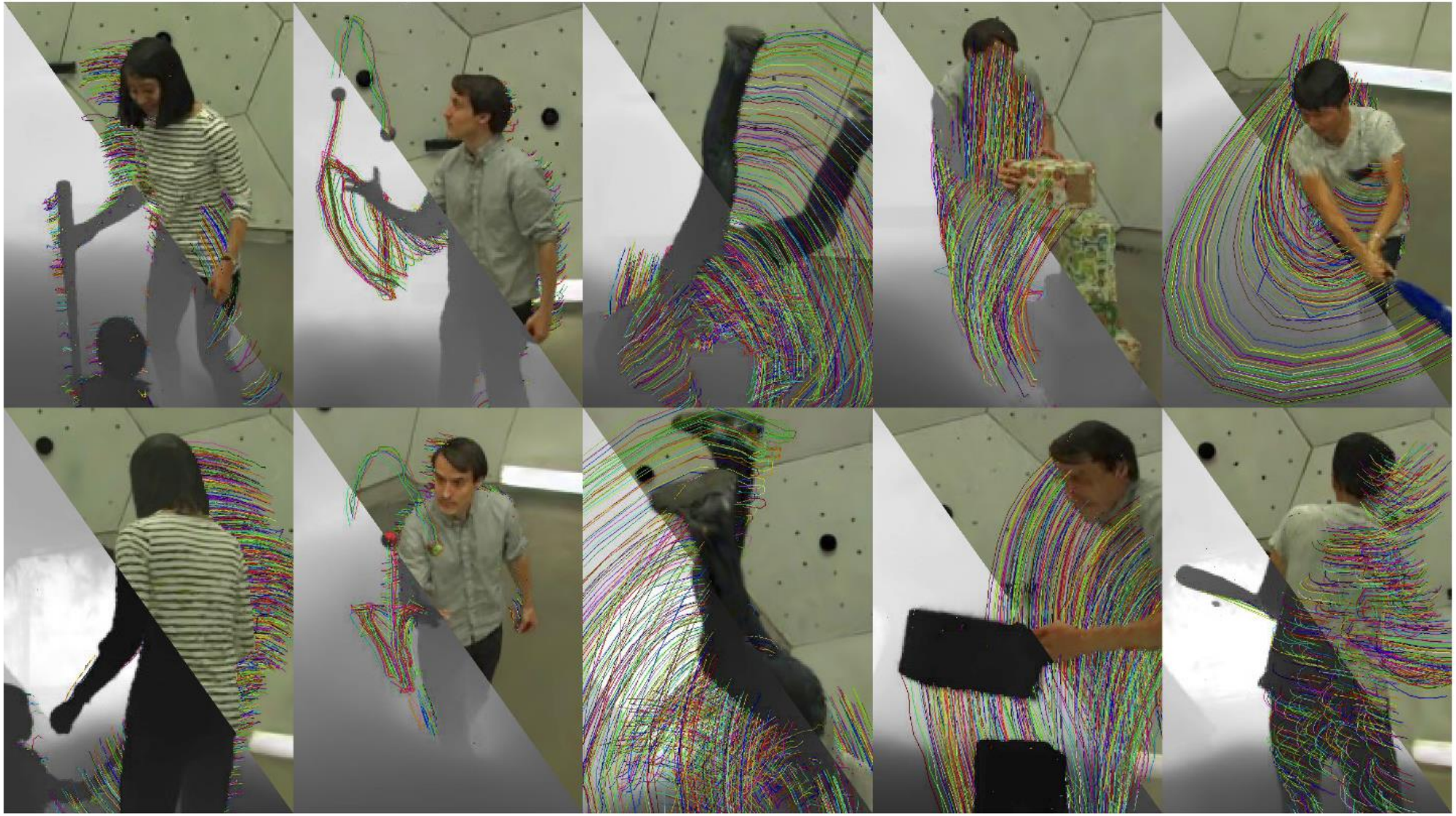
$$\mathcal{L}^{\text{iso}} = \frac{1}{k|\mathcal{S}|} \sum_{i \in \mathcal{S}} \sum_{j \in \text{knn}_{i;k}} w_{i,j} \left| \left\| \mu_{j,0} - \mu_{i,0} \right\|_2 - \left\| \mu_{j,t} - \mu_{i,t} \right\|_2 \right|$$

Dynamic 3D Gaussians: Tracking by Persistent Dynamic View Synthesis. Luiten et al

Results

Dynamic 3D Gaussians: Tracking by Persistent Dynamic View Synthesis. Luiten et al

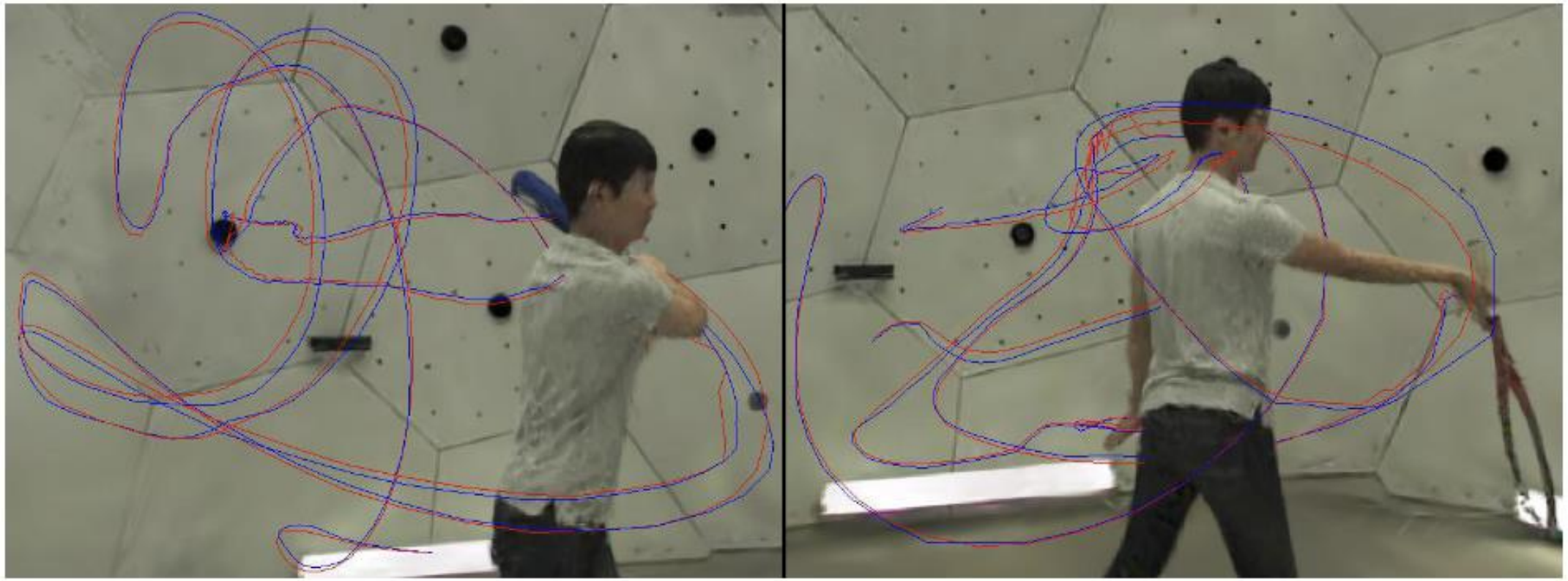
Results



Dynamic 3D Gaussians: Tracking by Persistent Dynamic View Synthesis. Luiten et al

Emerging Tracking from Image Rendering

Red curve: ground-truth tracks; Blue curve: predicted tracks



Dynamic 3D Gaussians: Tracking by Persistent Dynamic View Synthesis. Luiten et al

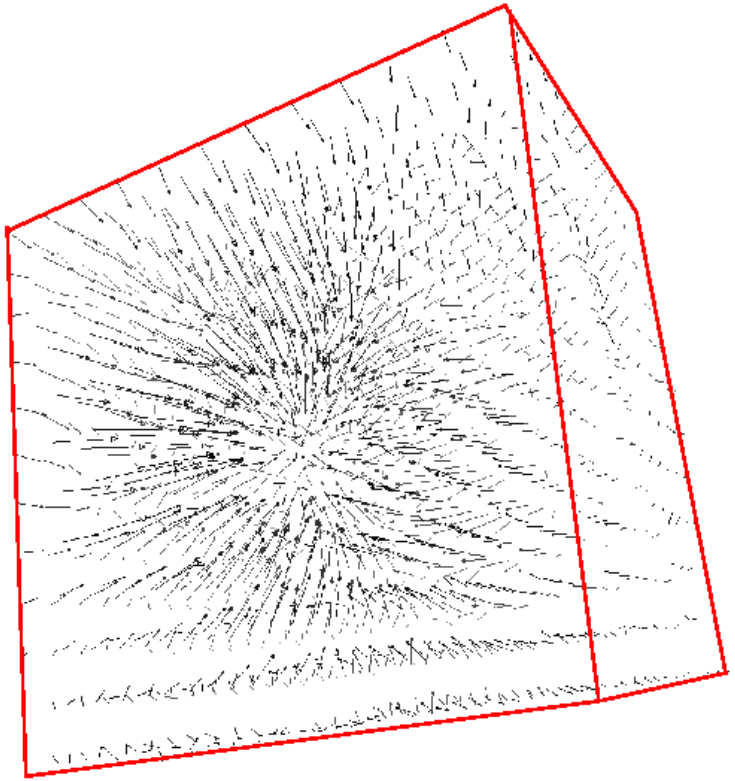
Applications: Augmented Reality

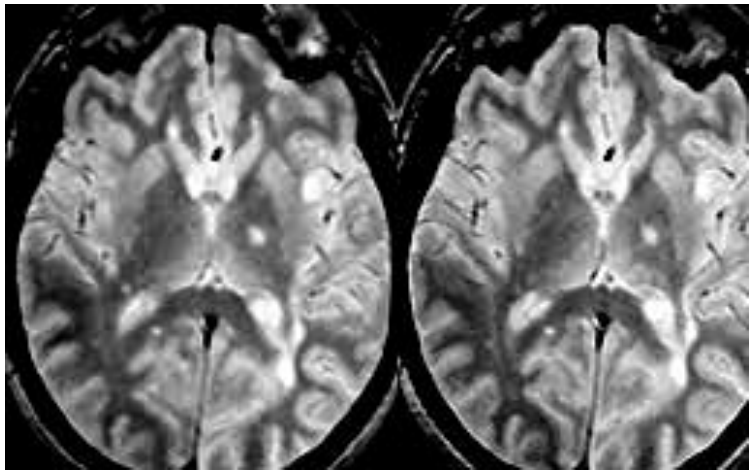
Dynamic 3D Gaussians: Tracking by Persistent Dynamic View Synthesis. Luiten et al

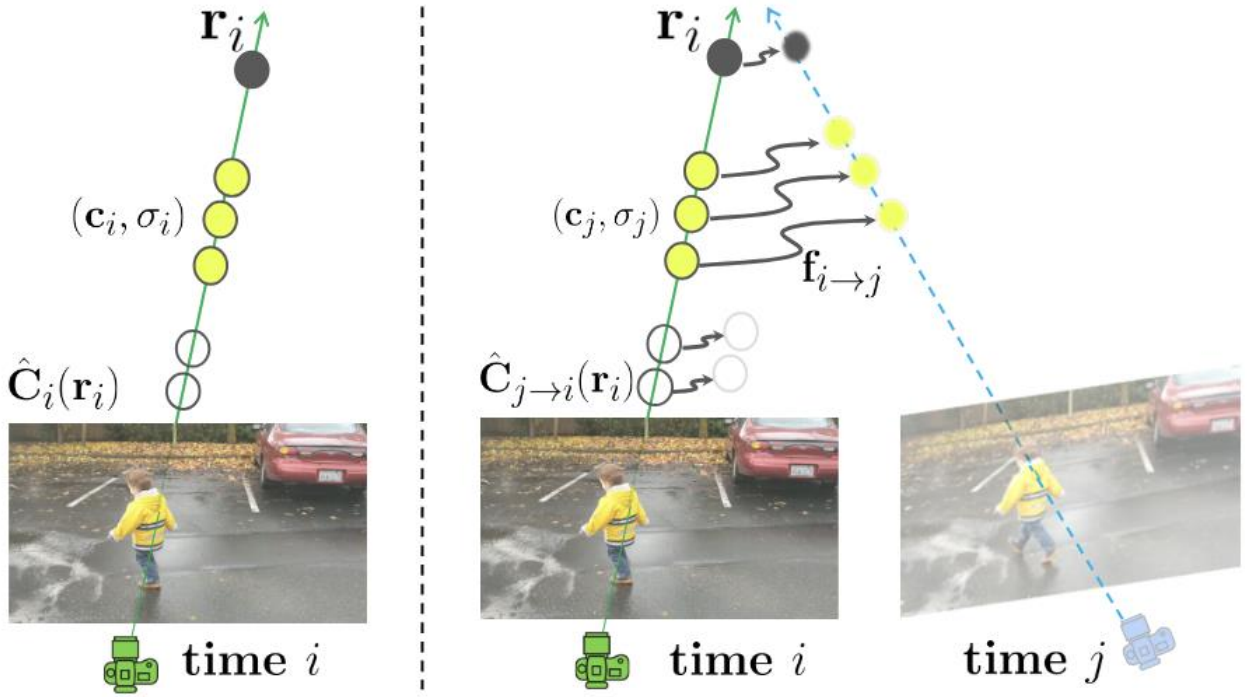
Content

- 2D Motion Representations
- 3D Motion Representations
  - ➢4D Volume Rendering with NERF
  - ➢Explicit Motion Field with NERF
  - ➢4D Volume Rendering with 3DGS
  - ➢Explicit Motion Field with 3DGS
  - ➢Implicit Motion Field with 3DGS
  - ➢4D Foundation Models
  - ➢Multi-view Video Generation (we'll cover in the next lecture)

Model a Vector Field

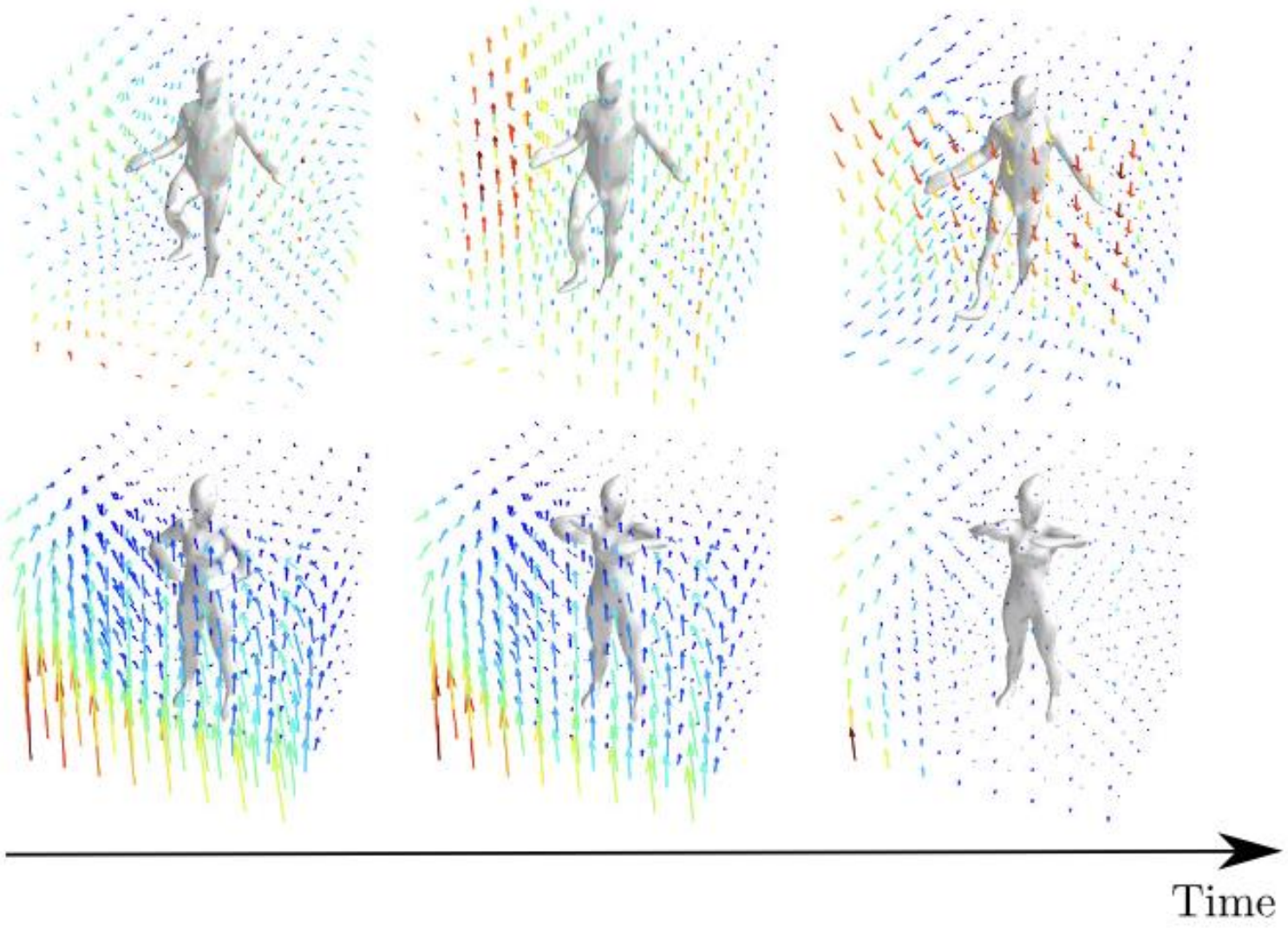






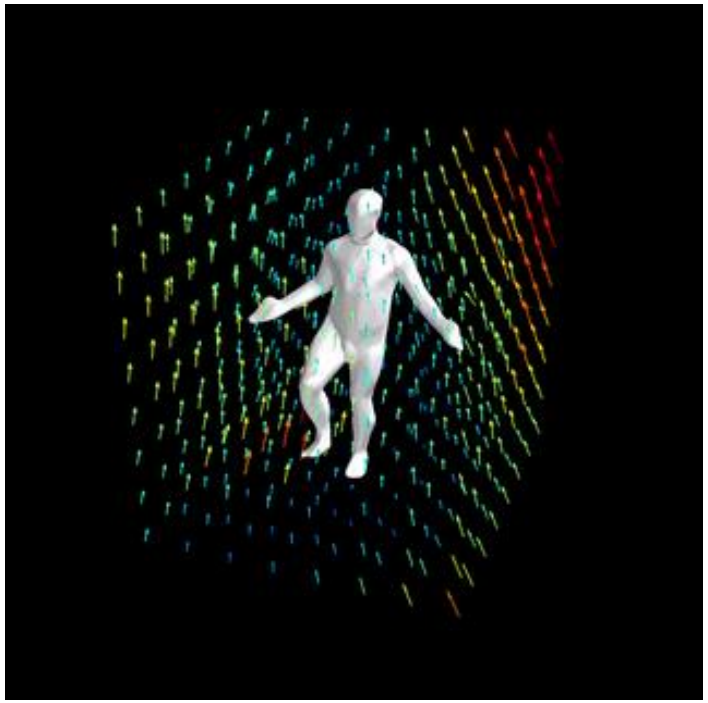
https://www-sop.inria.fr/epidaure/Demonstrations/ms_evolution/HTML/Field.html

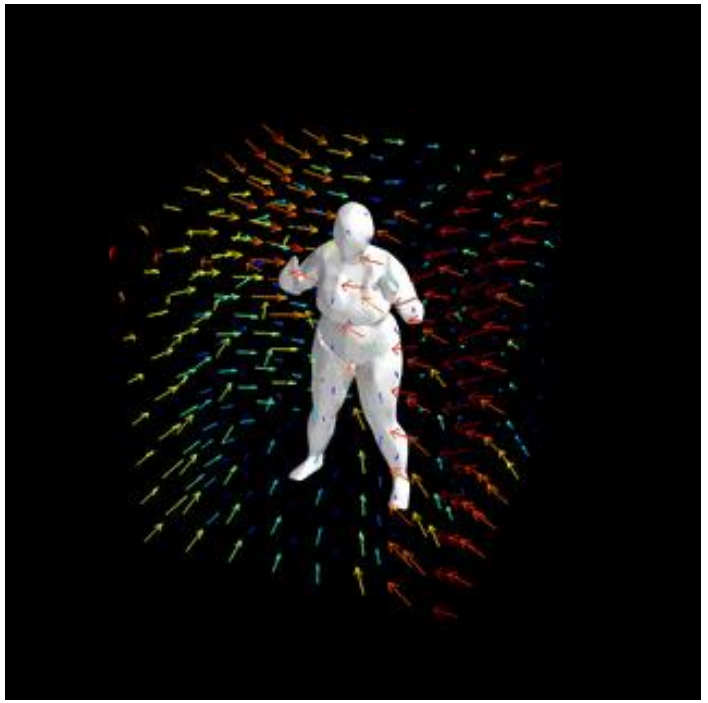
Decoupling Shape and Motion Representations



Occupancy Flow: 4D Reconstruction by Learning Particle Dynamics. Niemeyer et al.

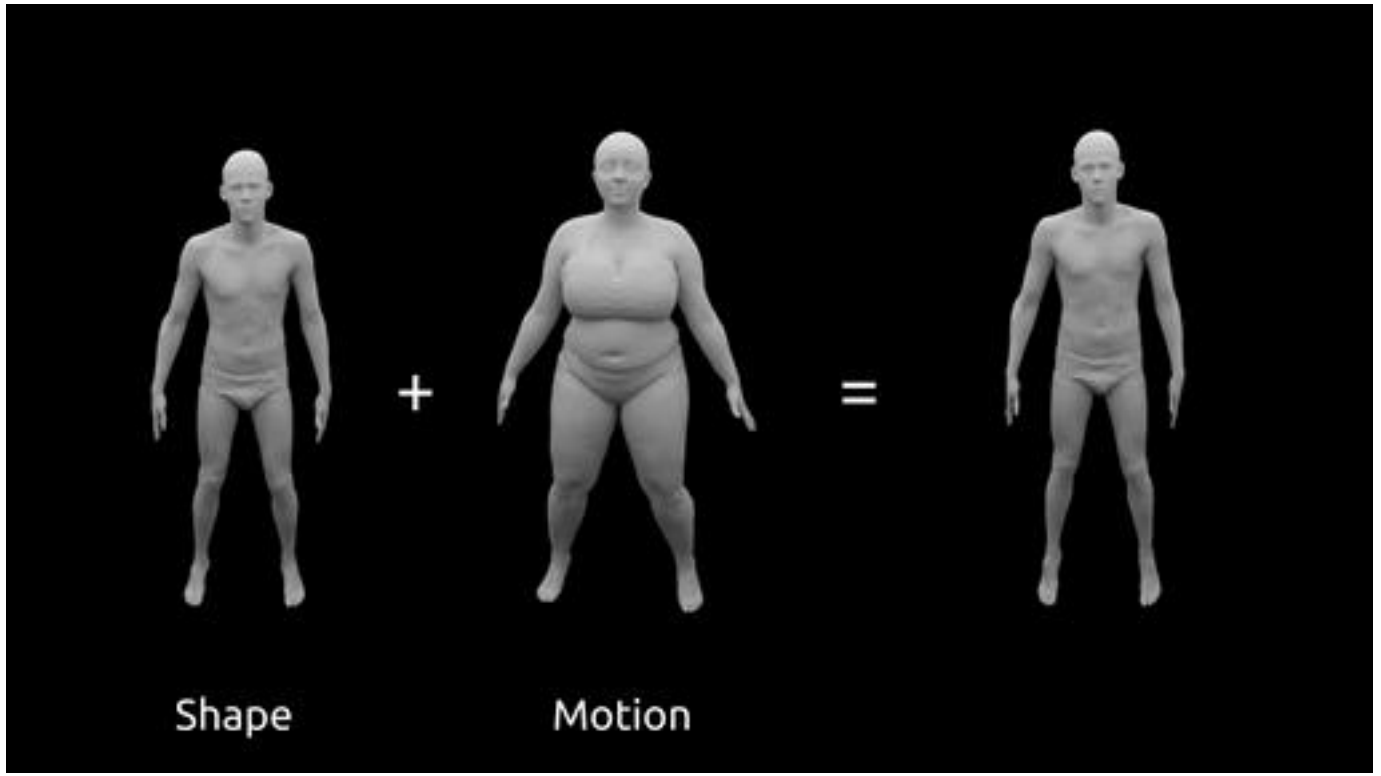
Modeling High-Resolution Motions with a Vector Field





Occupancy Flow: 4D Reconstruction by Learning Particle Dynamics. Niemeyer et al.

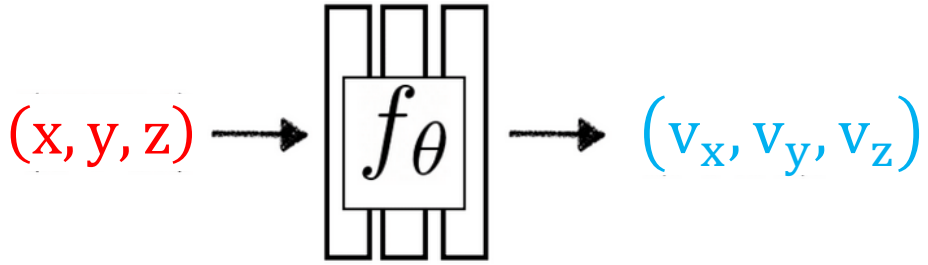
Transfer Motions to Other Instances



Occupancy Flow: 4D Reconstruction by Learning Particle Dynamics. Niemeyer et al.

How to Model a Dense Motion Field?

Prediction of a neural network



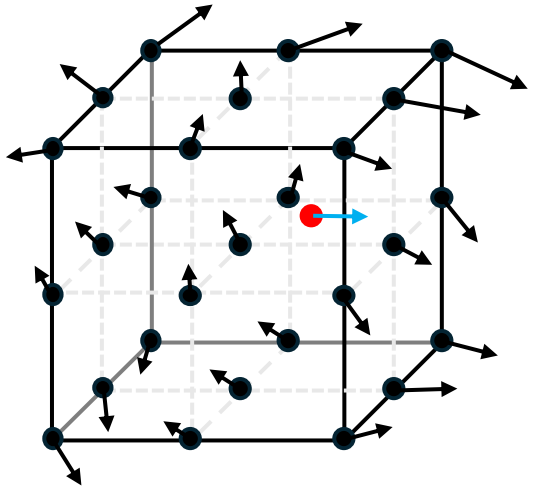
Pros:

- Capture high-frequency motions

Cons:

- Difficult to train

Interpolation of a grid



$$v_{xyz} = \sum_{i \in \mathcal{N}_{xyz}} w_i v_i$$

Pros:

- Share motions across particles (low rank approximation of motions)

Cons:

- Incapable of capturing high-frequency motions

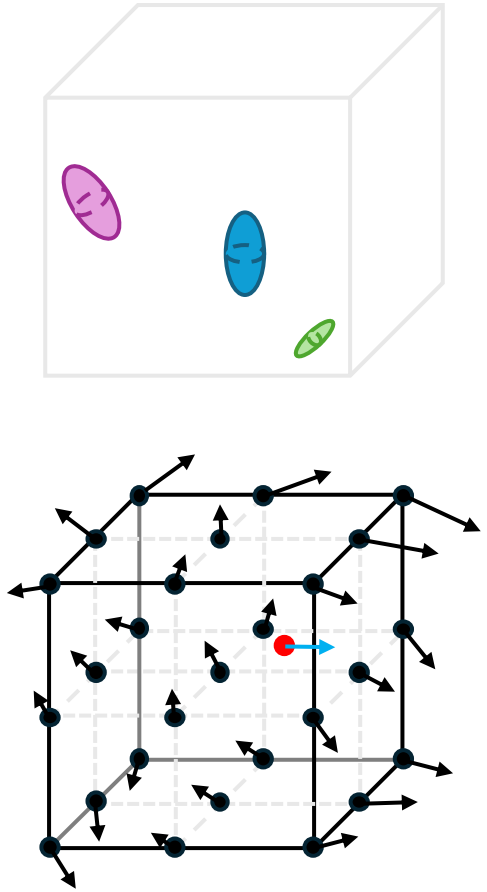
Common RGB Videos Contain only Low-frequency Motions

Low-frequency motions

Interpolating motions within a grid is a good choice on common RGB videos

High-frequency motions

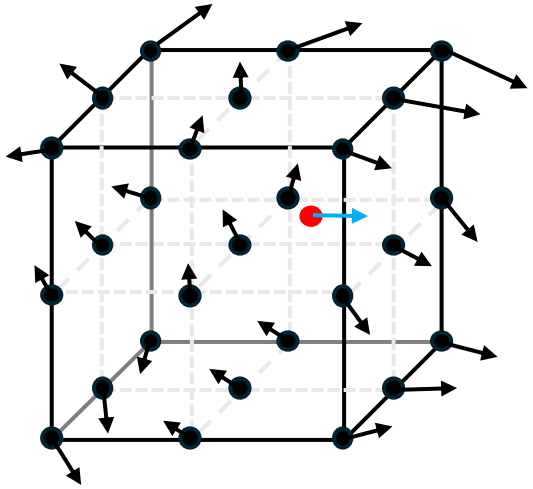
Goal: Infer 3D GS and the Motion Field from RGB Videos for 4D Reconstruction



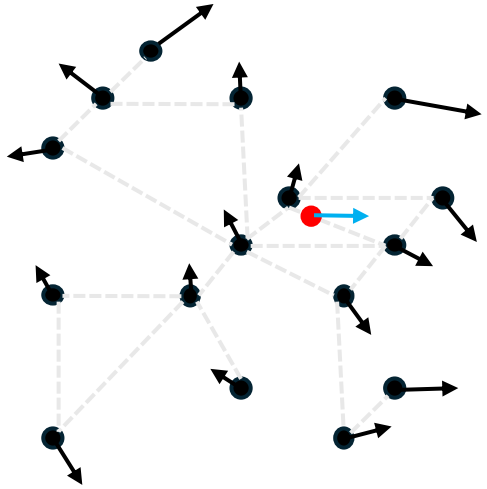
Video source: Shape of Motion: 4D Reconstruction from a Single Video

We Can Extend Grid-based Interpolation to Graph-based Interpolation

Interpolation of a grid



Interpolation of a graph



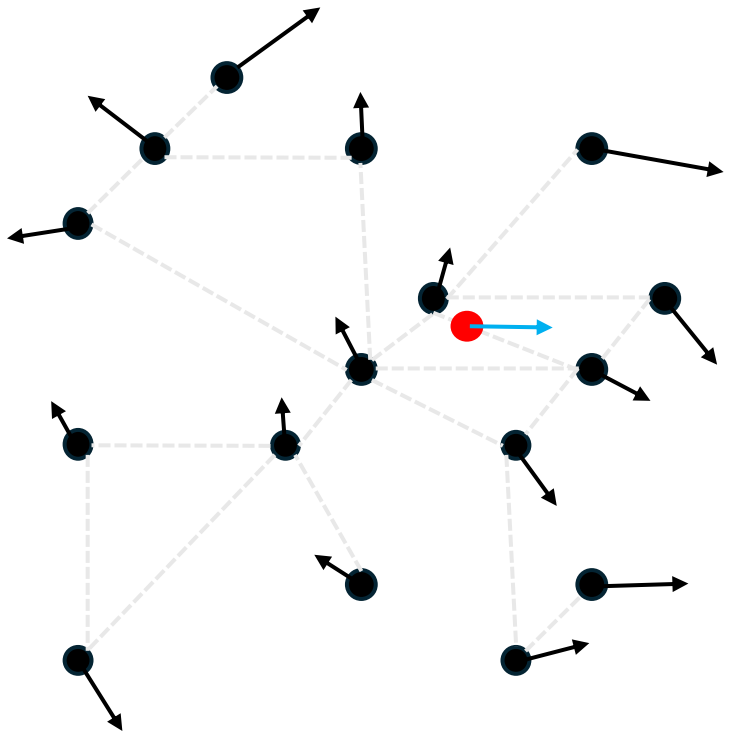
$$v_{xyz} = \sum_{i \in \mathcal{N}_{xyz}} w_i v_i$$

Graph-based Interpolation

- **Rigid transformation of 3D GS**: Transform a 3D GS with parameters (μ, Σ) using a rigid transformation $Q = [R;\ t]$

$$\mu' = R\mu + t; \qquad \Sigma' = R\Sigma$$

- **Motivation**: Real-world motion typically behaves rigidly, smoothly, and compactly, meaning that the actual solution is low-rank and driven by a few key "eigen" motions.
- **4D Motion Scaffold**: Encodes these local "eigen" motions and interpolates the dense deformation field with a structured graph $(\mathcal{V}, \mathcal{E})$



MoSca: Dynamic Gaussian Fusion from Casual Videos via 4D Motion Scaffolds. Lei et al.

Graph-based Interpolation

- Graph nodes $\mathcal{V} = \{\mathrm{v}^{(\mathrm{m})}\}$: each node is a 6-DoF rigid motion trajectory

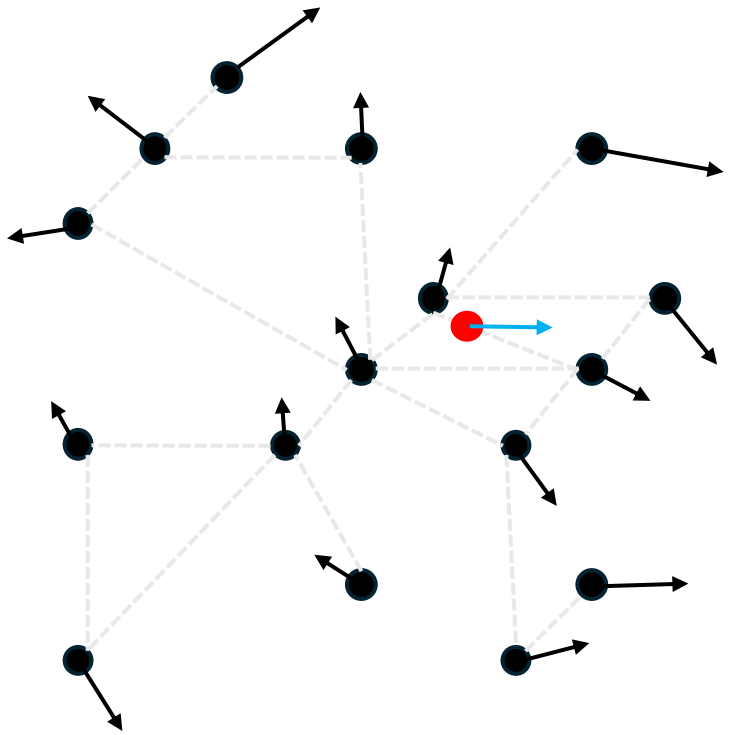
$$\mathrm{v}^{(\mathrm{m})} = \left(\left[\mathrm{Q}_1^{(\mathrm{m})}, \mathrm{Q}_2^{(\mathrm{m})}, \cdots, \mathrm{Q}_\mathrm{T}^{(\mathrm{m})}\right], \mathrm{r}^{(\mathrm{m})}\right)$$

  where $\mathrm{Q} = [\mathrm{R};\ \mathrm{t}]$ and $\mathrm{r}^{(\mathrm{m})}$ denotes control radius

- Graph edge:

$$\mathcal{E}(m) = \mathrm{KNN}_{n \in \{1,\ldots,M\}} \left[D_{\mathrm{curve}}(m,n)\right],$$

$$D_{\mathrm{curve}}(m,n) = \max_{t=1,2,\ldots,T} \|\mathbf{t}_t^{(m)} - \mathbf{t}_t^{(n)}\|,$$



MoSca: Dynamic Gaussian Fusion from Casual Videos via 4D Motion Scaffolds. Lei et al.

Graph-based Interpolation

- Graph nodes $\mathcal{V} = \{\mathrm{v}^{(\mathrm{m})}\}$: each node is a 6-DoF rigid motion trajectory

$$\mathrm{v}^{(\mathrm{m})} = \left(\left[\mathrm{Q}_1^{(\mathrm{m})}, \mathrm{Q}_2^{(\mathrm{m})}, \cdots, \mathrm{Q}_\mathrm{T}^{(\mathrm{m})}\right], \mathrm{r}^{(\mathrm{m})}\right)$$

where $\mathrm{Q} = [\mathrm{R};\ \mathrm{t}]$ and $\mathrm{r}^{(\mathrm{m})}$ denotes control radius

- Graph edge:

$$\mathcal{E}(m) = \mathrm{KNN}_{n \in \{1,\ldots,M\}} \left[D_{\mathrm{curve}}(m,n)\right],$$
$$D_{\mathrm{curve}}(m,n) = \max_{t=1,2,\ldots,T} \|\mathbf{t}_t^{(m)} - \mathbf{t}_t^{(n)}\|,$$

- Deformation field $\mathcal{W}$:

$$\mathcal{W}(\mathbf{x}, \mathbf{w}; t_{\mathrm{src}}, t_{\mathrm{dst}}) = \mathrm{DQB}\left(\{w_i, \Delta\mathbf{Q}^{(i)}\}_{i \in \mathcal{E}(m^*)}\right)$$

$$\mathrm{DQB}(\{(w_i, \mathbf{Q}_i)\}_{i=1}^L) = \frac{\sum_{i=1}^L w_i \hat{\mathbf{q}}_i}{\|\sum_{i=1}^L w_i \hat{\mathbf{q}}_i\|_{DQ}} \in SE(3),$$

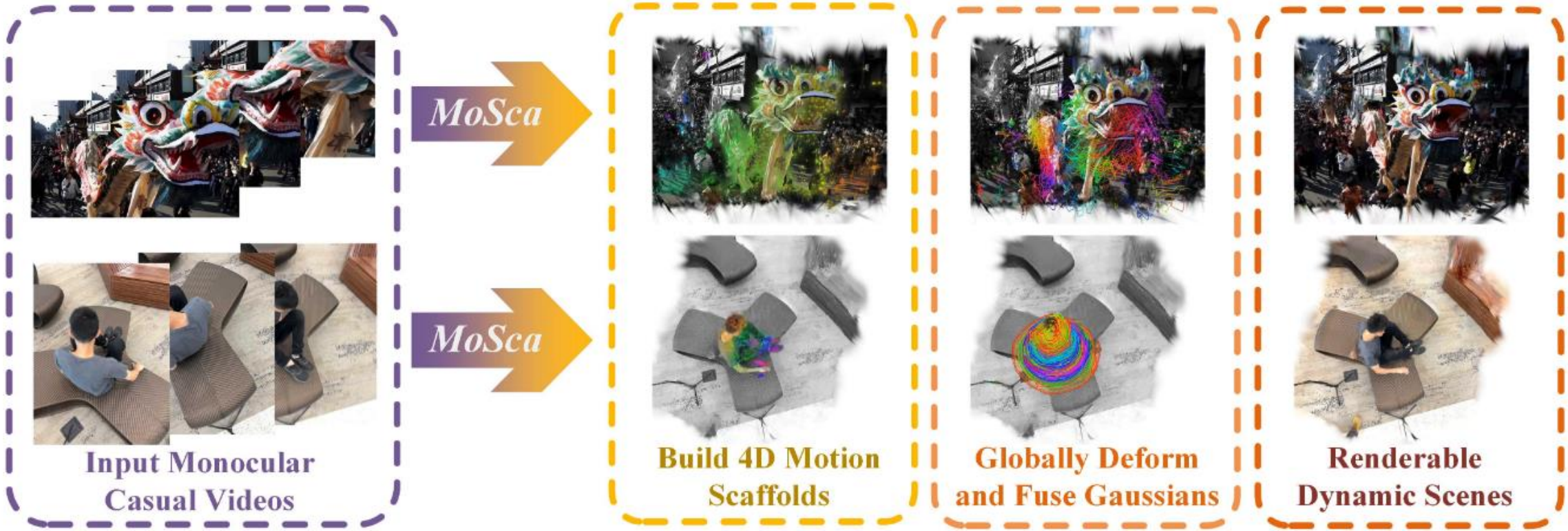
$$\Delta\mathbf{Q}^{(i)} = \mathbf{Q}_{t_{\mathrm{dst}}}^{(i)} \left(\mathbf{Q}_{t_{\mathrm{src}}}^{(i)}\right)^{-1}$$

$$w_i(\mathbf{x}, t_{\mathrm{src}}) = \exp\left(-\|\mathbf{x} - \mathbf{t}_{t_{\mathrm{src}}}^{(i)}\|_2^2 / 2r^{(i)}\right) \in \mathbb{R}^+.$$

MoSca: Dynamic Gaussian Fusion from Casual Videos via 4D Motion Scaffolds. Lei et al.

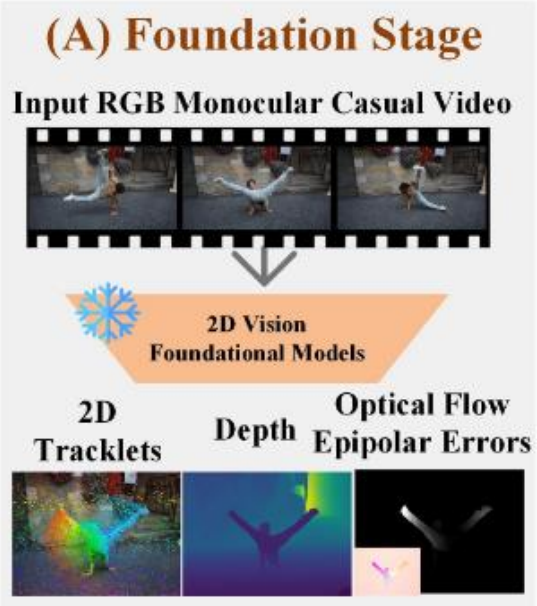
MoSca: Dynamic Gaussian Fusion from Casual Videos via 4D Motion Scaffolds

Jiahui Lei[1] Yijia Weng[2] Adam W. Harley[2] Leonidas Guibas[2] Kostas Daniilidis[1,3]
[1] University of Pennsylvania [2] Stanford University [3] Archimedes, Athena RC
{leijh, kostas}@cis.upenn.edu, {yijiaw, aharley, guibas}@cs.stanford.edu



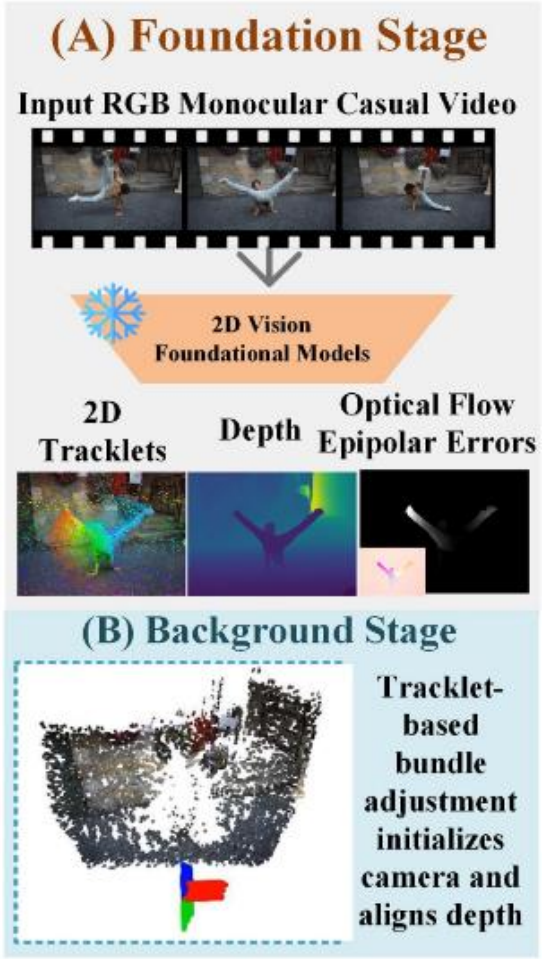
MoSca: Dynamic Gaussian Fusion from Casual Videos via 4D Motion Scaffolds. Lei et al.

Stage 1: Extract 2D Tracklets, Depth and Optical Flow Epipolar Errors from 2D Foundation Models



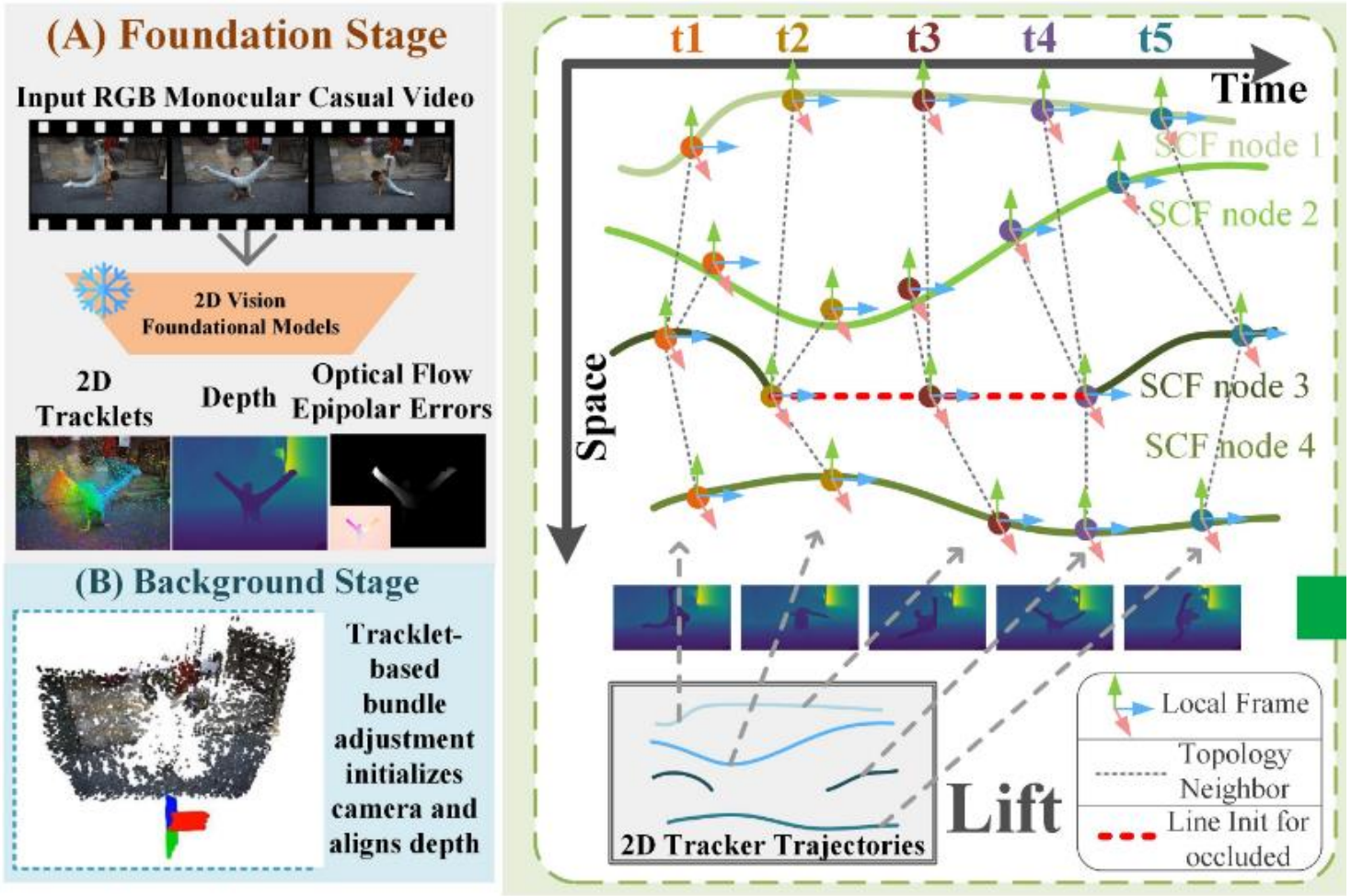
MoSca: Dynamic Gaussian Fusion from Casual Videos via 4D Motion Scaffolds. Lei et al.

Stage 2: Estimate Camera Motions by Separating Foreground and Background



MoSca: Dynamic Gaussian Fusion from Casual Videos via 4D Motion Scaffolds. Lei et al.

Stage 3: Initiate Graph Nodes $\mathcal{V}$ by Lifting 2D Tracklets to 3D



- Graph node

$$\mathbf{v}^{(m)} = \left(\left[\mathrm{Q}_1^{(m)}, \mathrm{Q}_2^{(m)}, \cdots, \mathrm{Q}_T^{(m)}\right], \mathrm{r}^{(m)}\right)$$

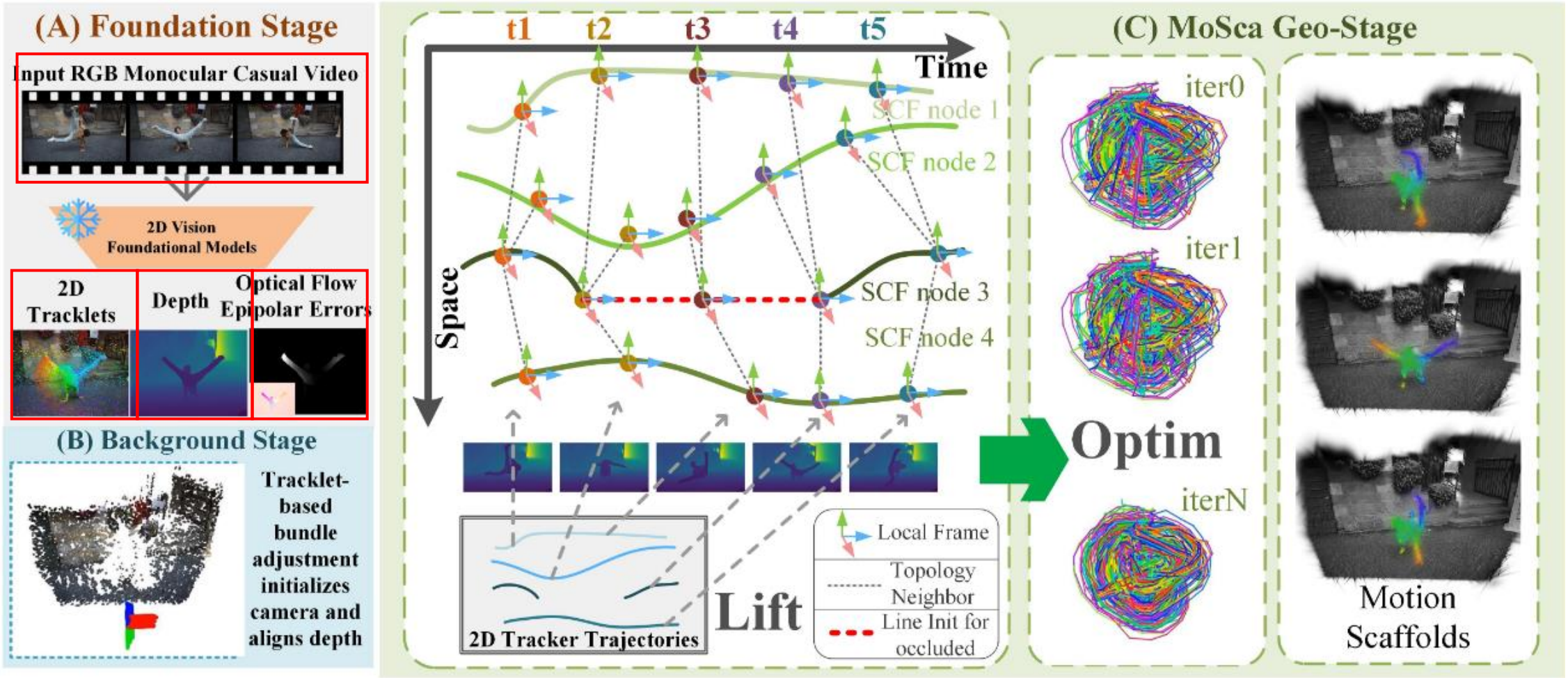
$$\mathrm{Q}_t^{(i)} = \left[\mathrm{I}, \mathrm{h}_t^{(i)}\right]$$

And $\mathbf{h}_t^{(i)}$ denotes the unprojected depth of 2D tracklet $\mathbf{i}$ at time $\mathbf{t}$

MoSca: Dynamic Gaussian Fusion from Casual Videos via 4D Motion Scaffolds. Lei et al.

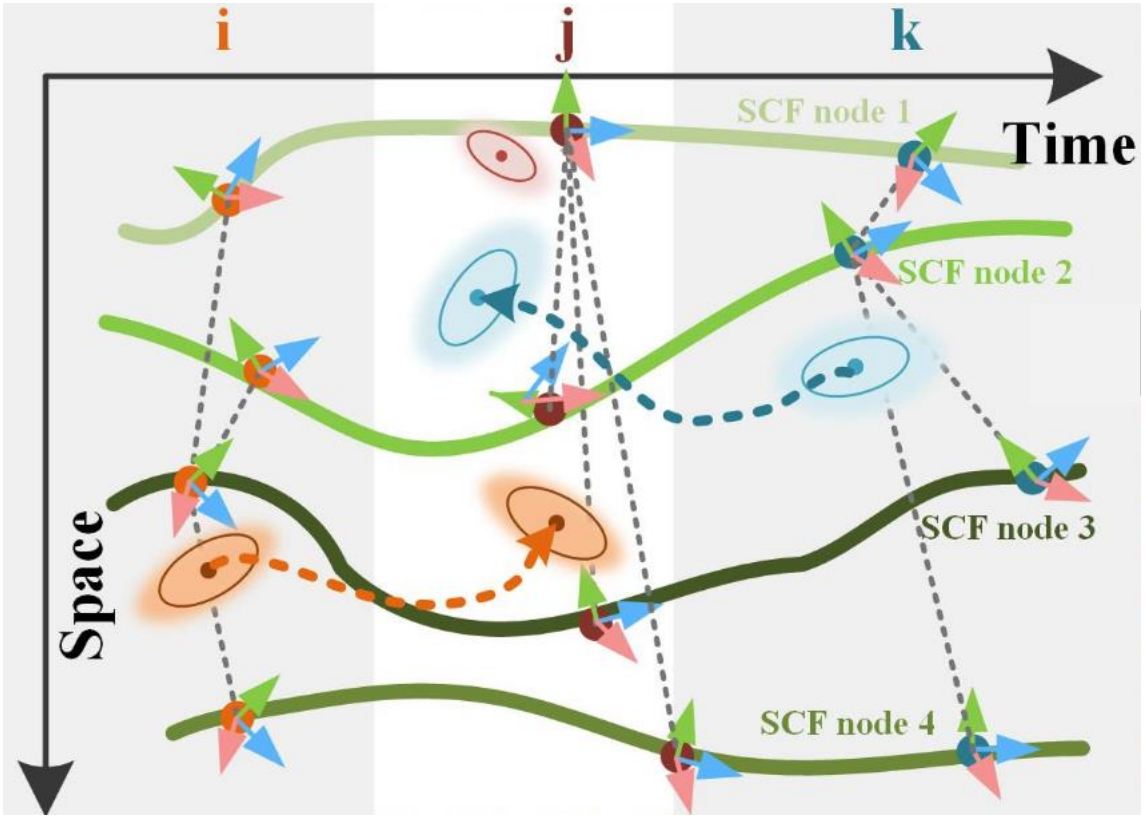
Stage 4: Optimize the 4D Motion Scaffolds

Supervise with RGB, depth, optical flow, tracking, smoothness and rigid-body losses.



MoSca: Dynamic Gaussian Fusion from Casual Videos via 4D Motion Scaffolds. Lei et al.

Why Motion Scaffolds are Good Motion Representations?

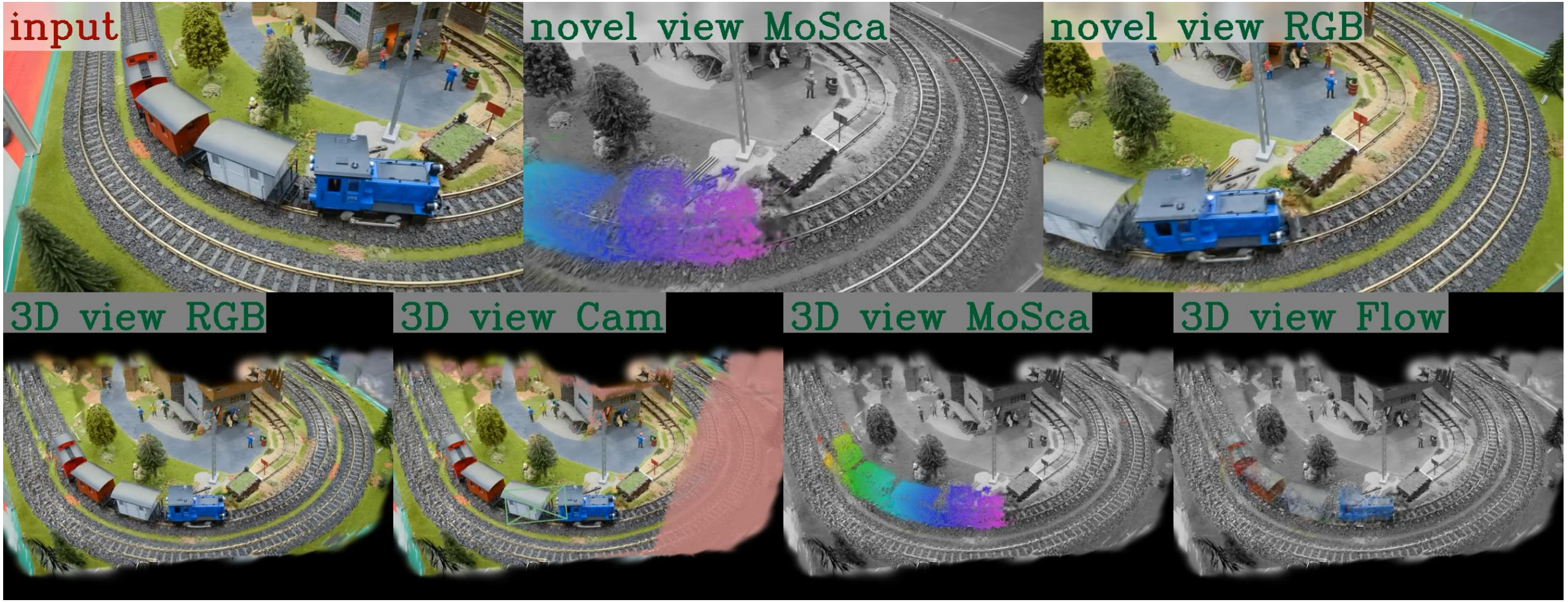


- The global deformation field $\boldsymbol{\mathcal{W}}$ can transform points at any time globally, enabling the fusion of all observed video frames into a single coherent representation.
- Let $\mathbf{t_j^{ref}}$ denote the time step when the $\mathbf{j}$-th GS is initiated and $\mathbf{\Delta w_j}$ be he per-Gaussian learnable skinning weight correction:

$$\mathcal{G}(t) = \{(\mathbf{T}_j(t)\mu_j, \mathbf{T}_j(t)R_j, s_j, o_j, c_j) \mid \mathbf{T}_j(t) = \mathcal{W}(\mu_j, \mathbf{w}(\mu_j, t_j^{\text{ref}}) + \Delta\mathbf{w}_j; t_j^{\text{ref}}, t)\}_{j=1}^{N}$$

MoSca: Dynamic Gaussian Fusion from Casual Videos via 4D Motion Scaffolds. Lei et al.

Results



MoSca: Dynamic Gaussian Fusion from Casual Videos via 4D Motion Scaffolds. Lei et al.

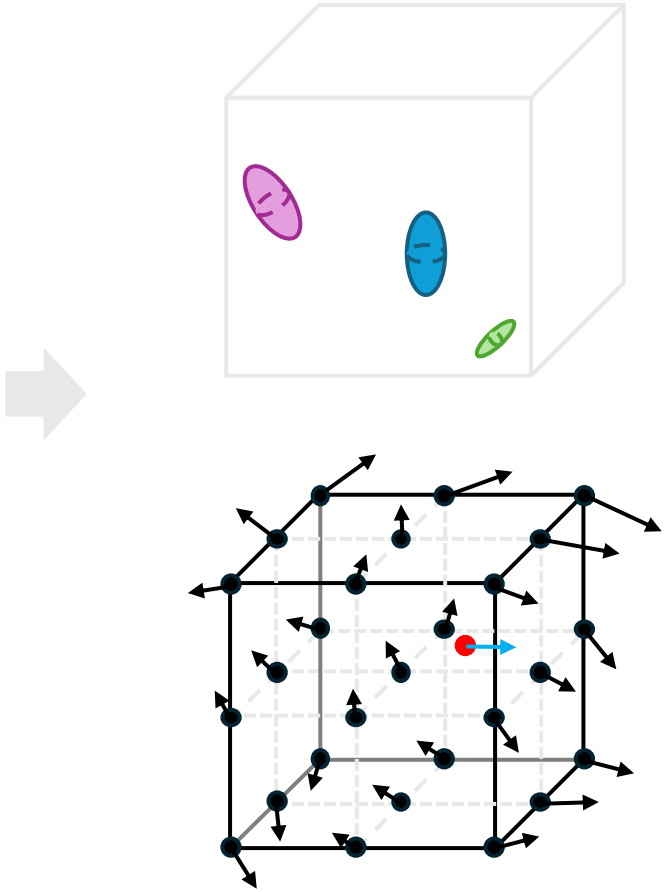
Results

MoSca: Dynamic Gaussian Fusion from Casual Videos via 4D Motion Scaffolds. Lei et al.

Content

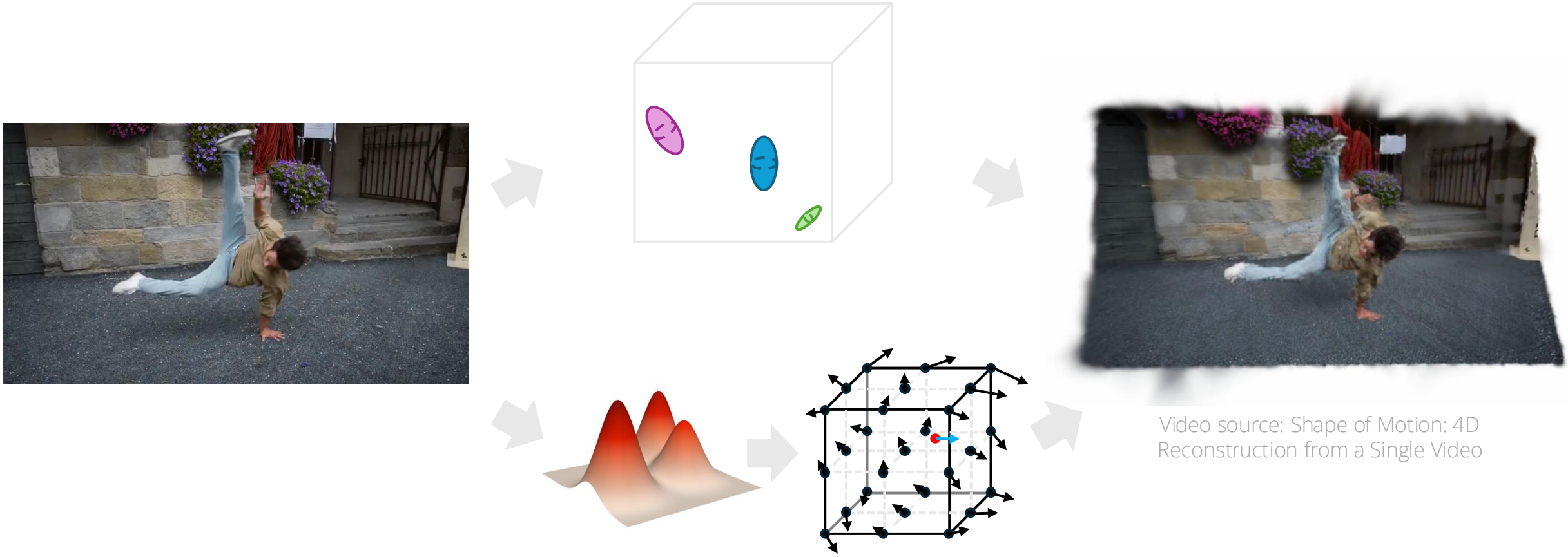
- 2D Motion Representations
- 3D Motion Representations
  - ➢4D Volume Rendering with NERF
  - ➢Explicit Motion Field with NERF
  - ➢4D Volume Rendering with 3DGS
  - ➢Explicit Motion Field with 3DGS
  - ➢Implicit Motion Field with 3DGS
  - ➢4D Foundation Models
  - ➢Multi-view Video Generation (we'll cover in the next lecture)

The Motion Field of the Input Video May Be High-Dimensional, which is difficult to infer…



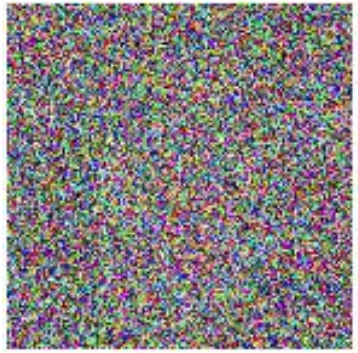
Video source: Shape of Motion: 4D Reconstruction from a Single Video

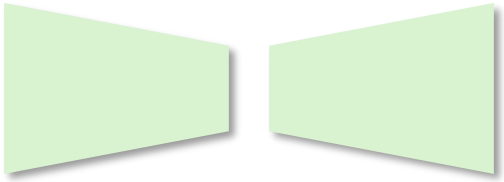
Idea: First Compress Motions into Compact Latent Features, then Decode Back to High-Dimensional Fields



Video source: Shape of Motion: 4D Reconstruction from a Single Video

We Have the Same Problem in Visual Generation



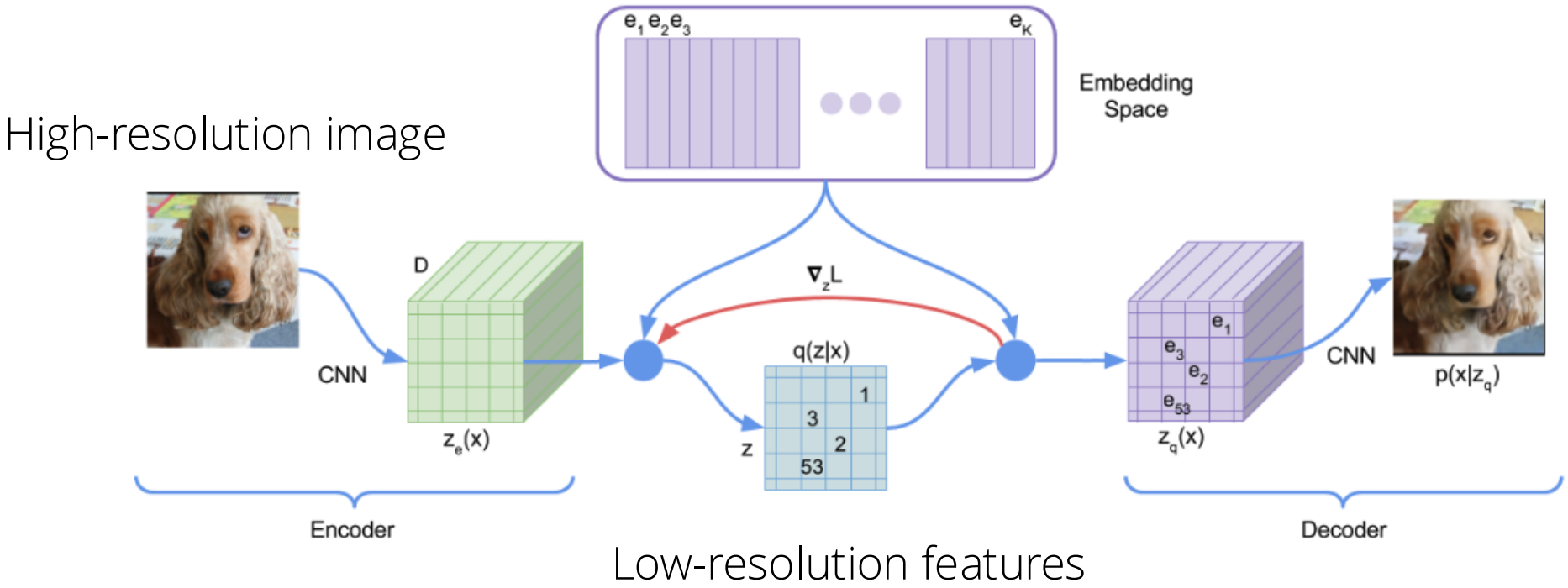


Generative model

- Generating rgb pixels directly is super expensive!
  - ➢ No structural prior: a 16x16 red patch can be denoted by much simpler representations
  - ➢ High computational cost

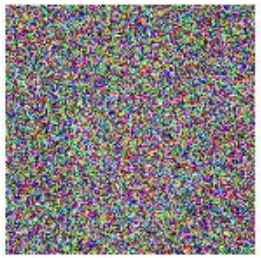
Idea: Encode Images in a Compact Feature Space

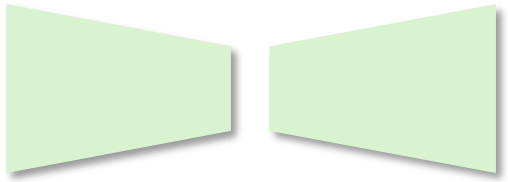
- Learn to encode images by auto-encoding
  - ➢ information bottleneck + reconstruction loss



Neural Discrete Representation Learning. Oord et al.

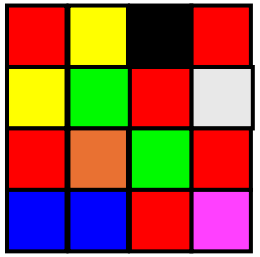
Idea: Generate Images in a Compact Feature Space

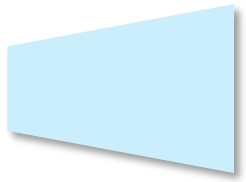




Generative model

Low-resolution features

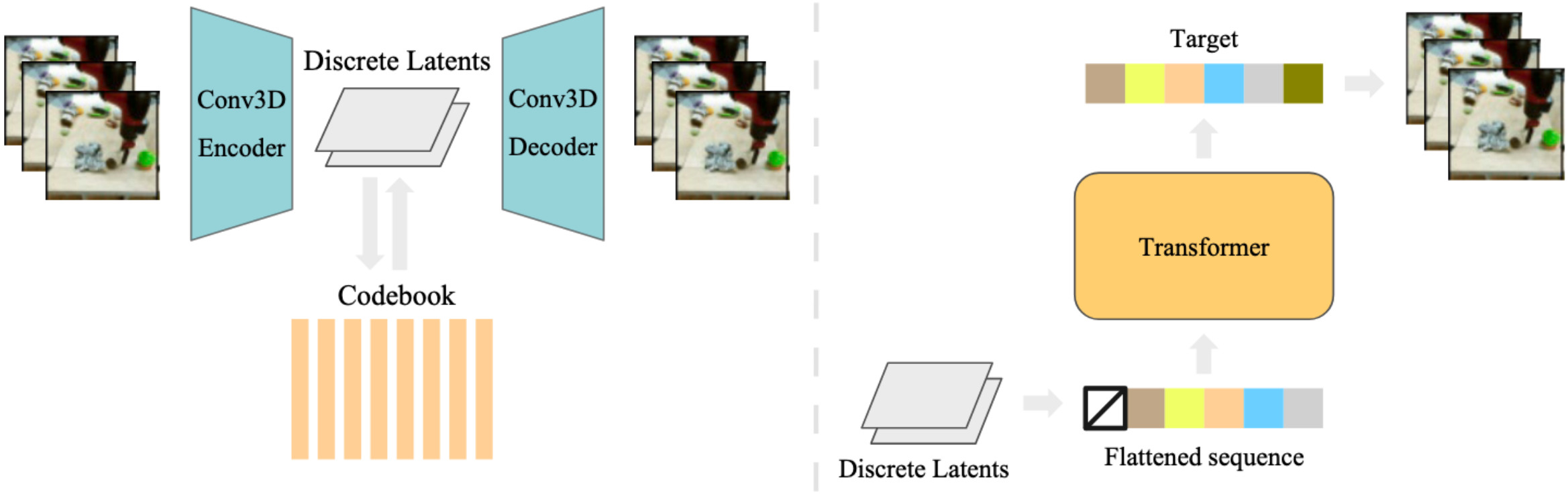




Decoder

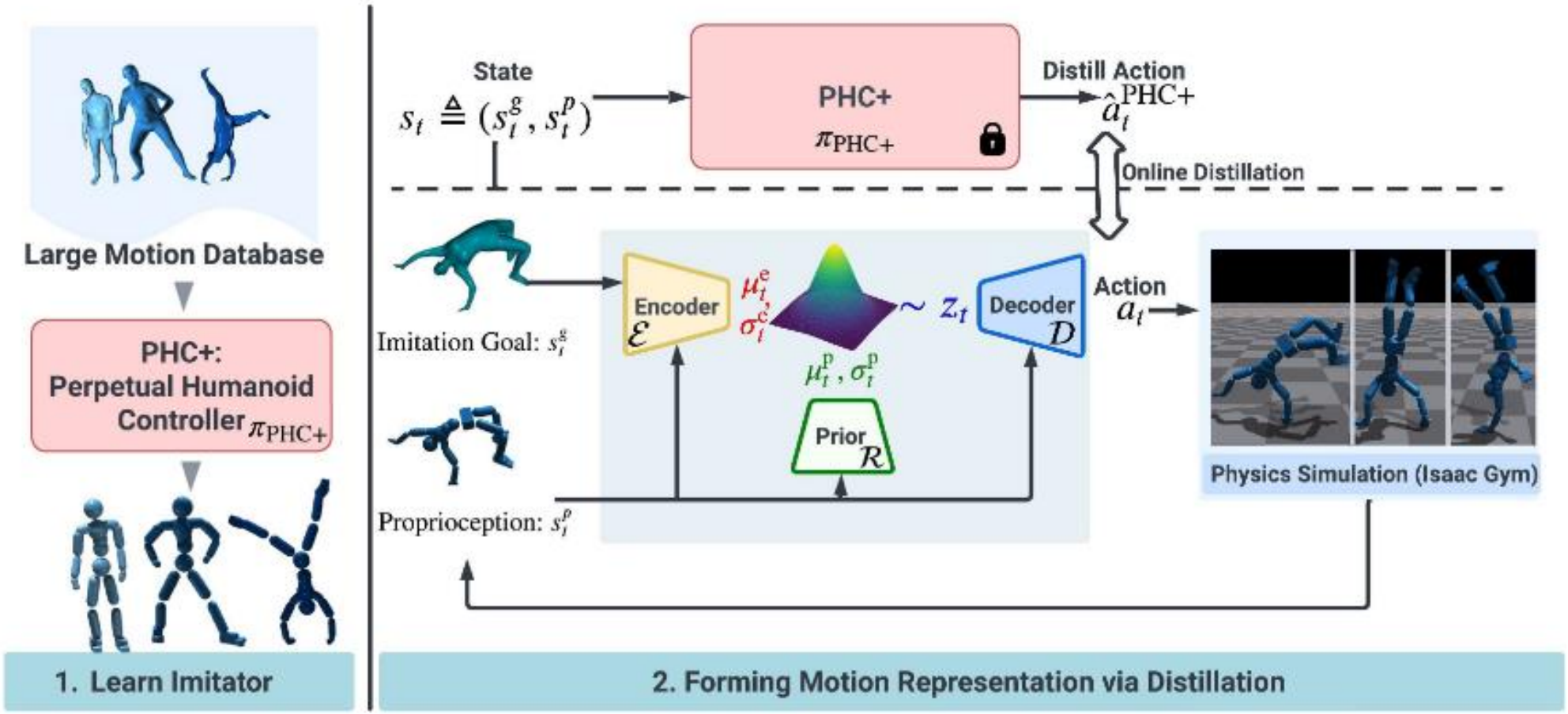
High-resolution image

Same Idea for Video Generation



VideoGPT: Video Generation using VQ-VAE and Transformers. Yan et al.

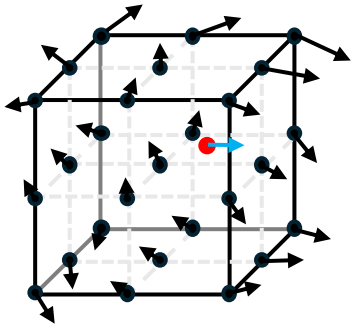
Same Idea for Human Motion Generation



We Need An AutoEncoder that Compresses a Motion Field

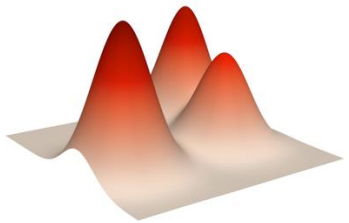
Static Object

Animated Object

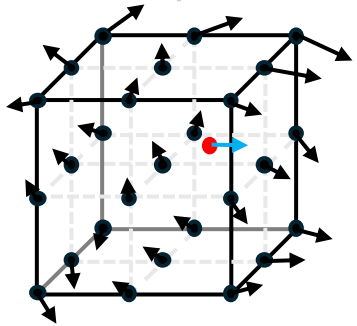


Motion Field

Encoder



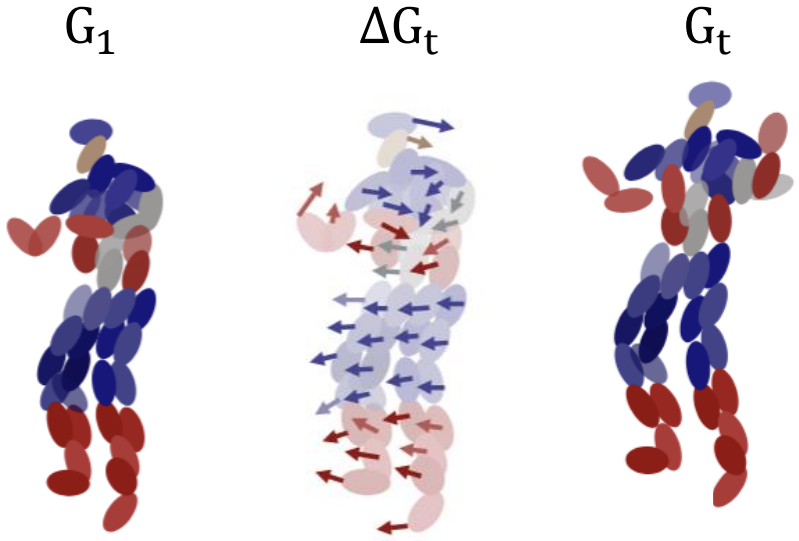
Decoder



Motion Field

Video generated by Gaussian Variation Field Diffusion for High-fidelity Video-to-4D Synthesis.

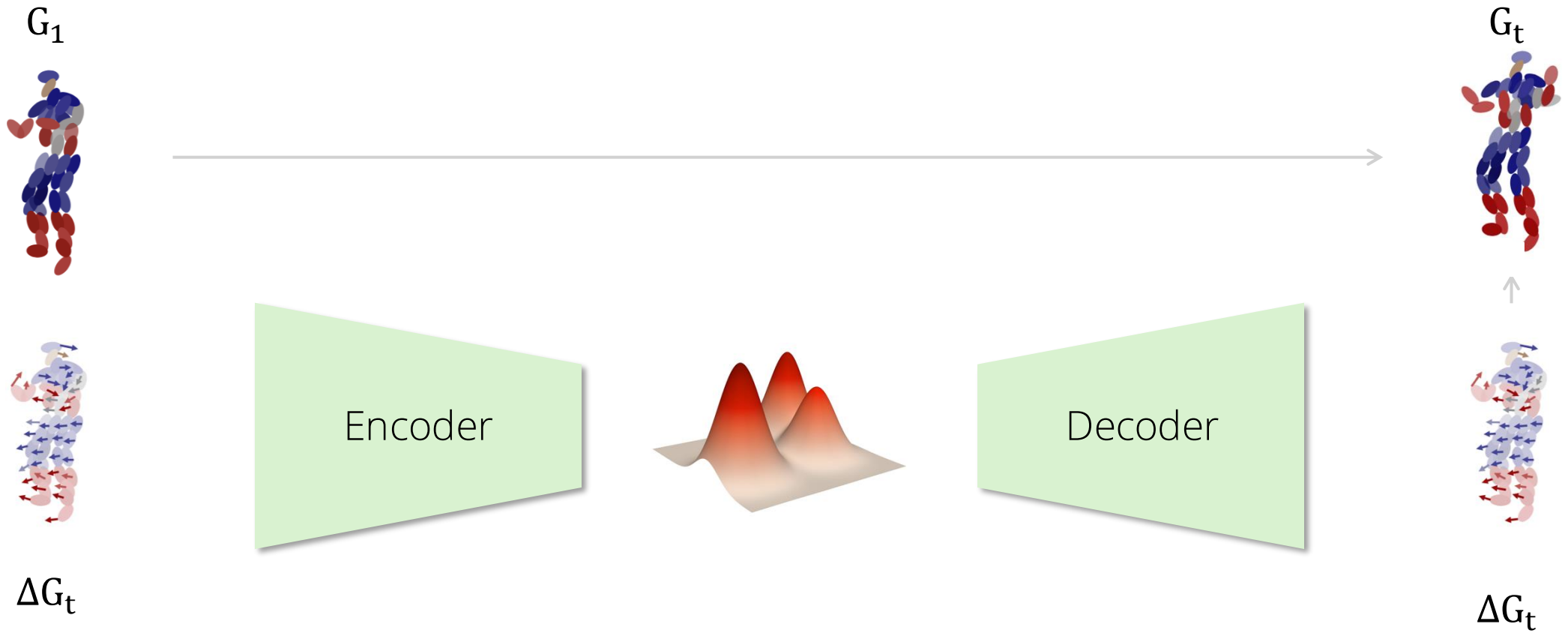
Gaussian Variation Field



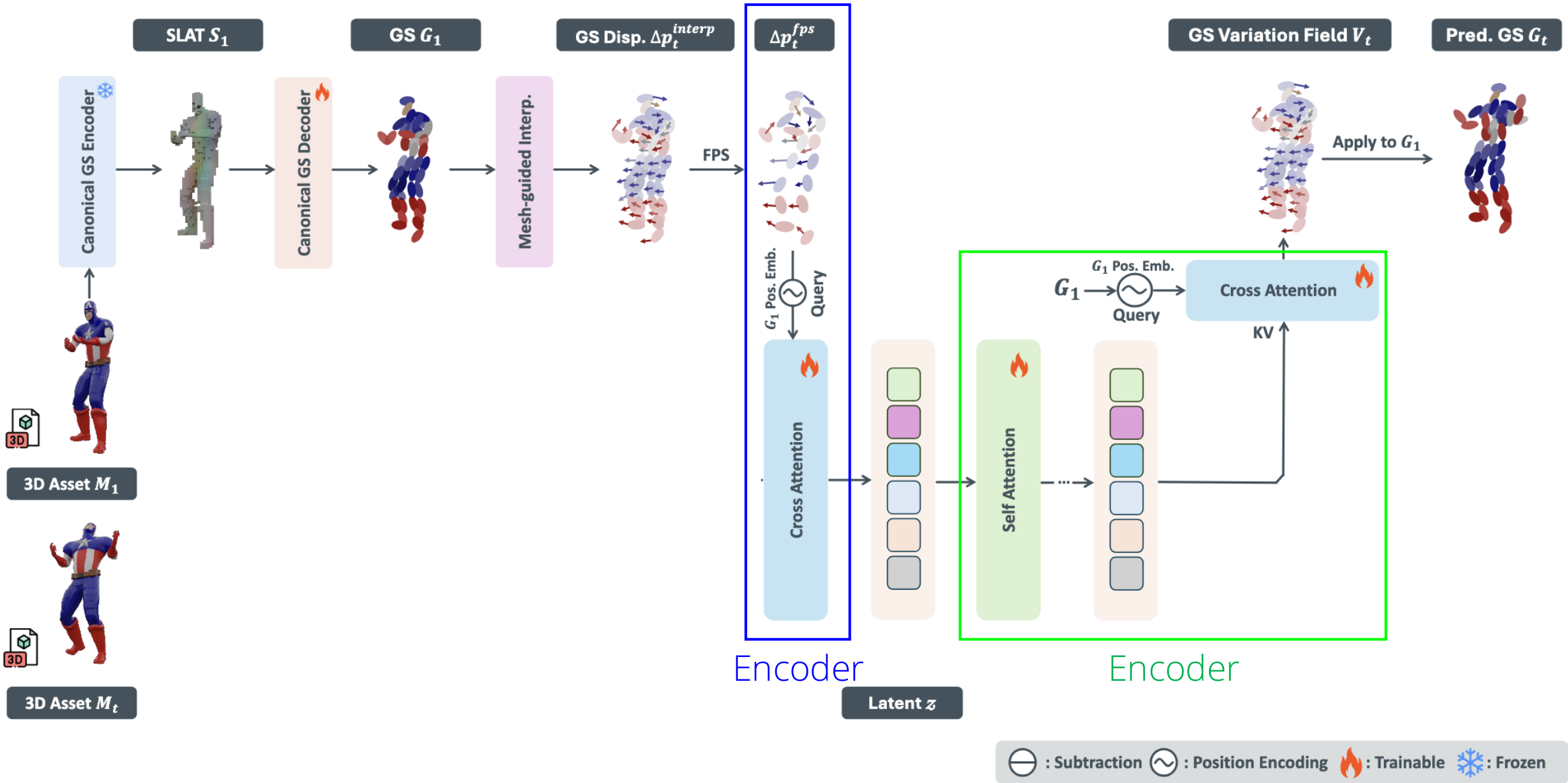
- Goal: generate a sequence of 3DGS models $\mathcal{G} = \{G_t\}_{t=1}^{T}$ that captures both the shape, appearance, and motion of the object.
- Gaussian Variation Field $\{\Delta G_t\}_{t=1}^{T}$ describes each Gaussian's attribute variations relative to G_1 over time.

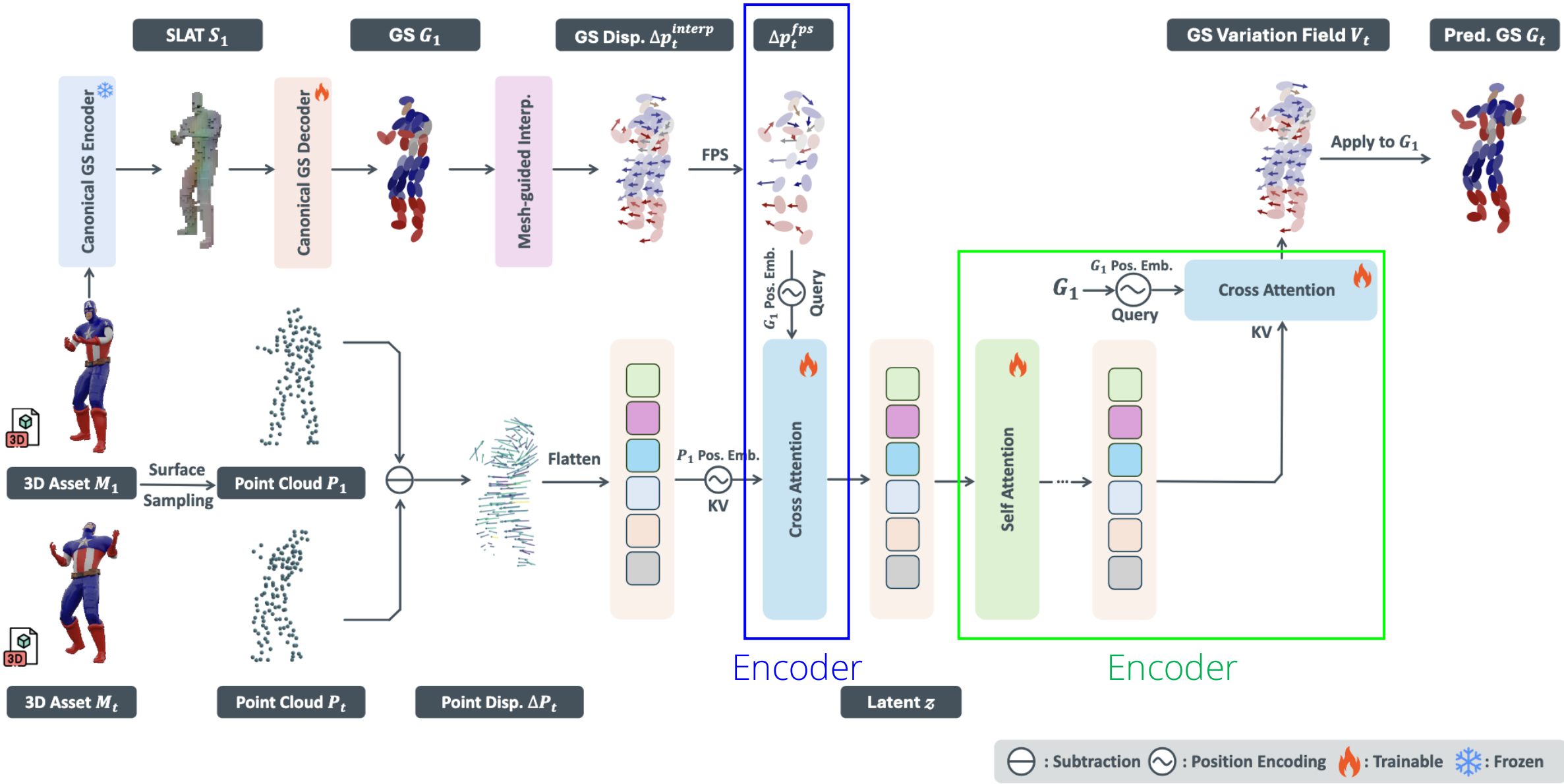
Gaussian Variation Field Diffusion for High-fidelity Video-to-4D Synthesis. Zhang et al.

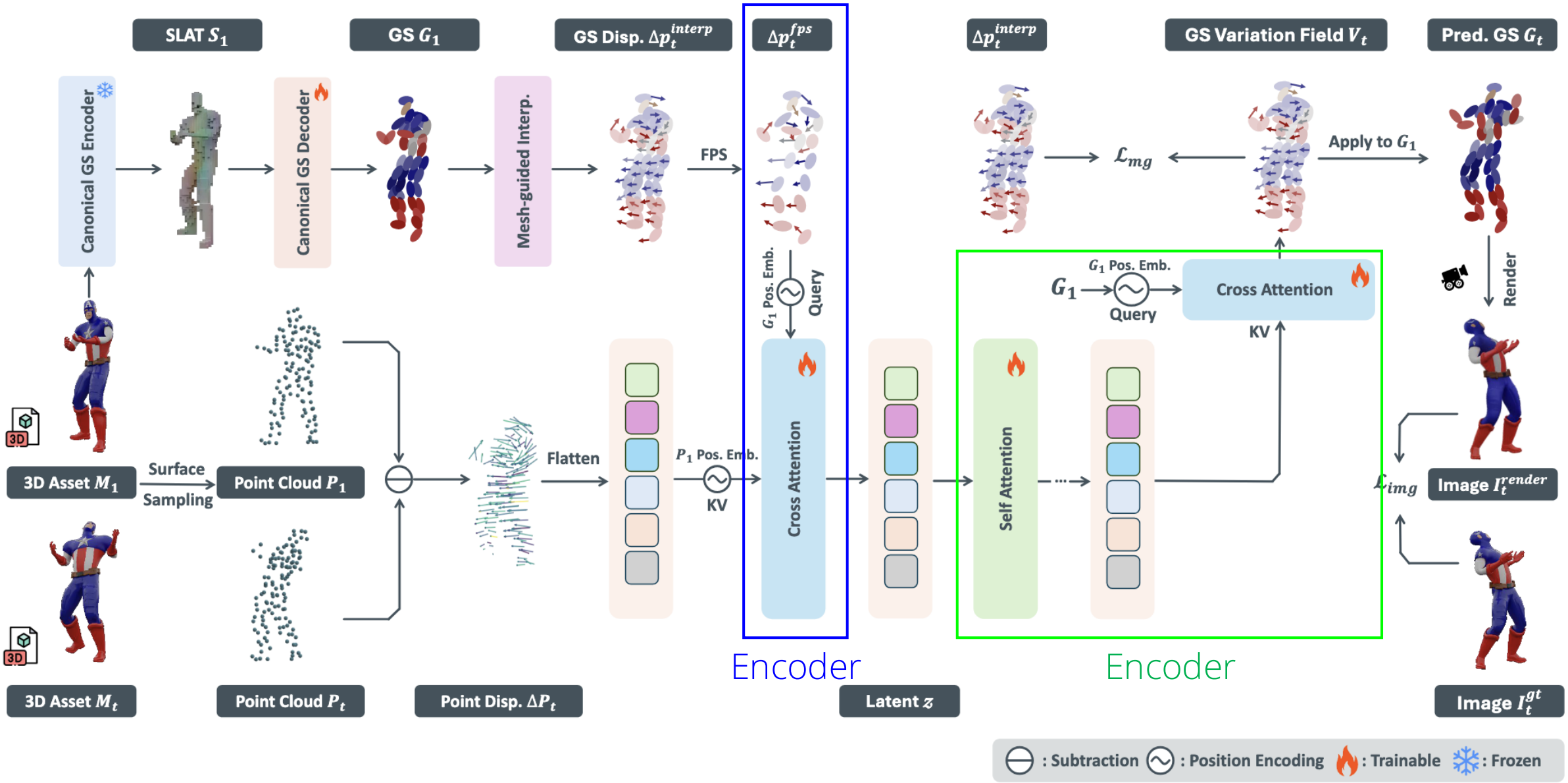
We Need An AutoEncoder that Compresses a Gaussian Variation Field



Gaussian Variation Field Diffusion for High-fidelity Video-to-4D Synthesis. Zhang et al.


SLAT S_1
GS G_1
GS Disp. Δp_t^{interp}
Δp_t^{fps}
GS Variation Field V_t
Pred. GS G_t
Canonical GS Encoder
Canonical GS Decoder
Mesh-guided Interp.
FPS
Apply to G_1
G_1 Pos. Emb.
Query
G_1
Cross Attention
KV
Self Attention
3D Asset M_1
3D Asset M_t
Latent z
Encoder
Encoder
: Subtraction
: Position Encoding
: Trainable
: Frozen


SLAT S_1
GS G_1
GS Disp. Δp_t^{interp}
Δp_t^{fps}
GS Variation Field V_t
Pred. GS G_t
Canonical GS Encoder
Canonical GS Decoder
Mesh-guided Interp.
FPS
Apply to G_1
G_1 Pos. Emb.
Query
G_1
Cross Attention
KV
3D Asset M_1
Surface Sampling
Point Cloud P_1
Flatten
P_1 Pos. Emb.
KV
Cross Attention
Self Attention
3D Asset M_t
Point Cloud P_t
Point Disp. ΔP_t
Latent z
Encoder
Encoder
: Subtraction
: Position Encoding
: Trainable
: Frozen


SLAT S_1
GS G_1
GS Disp. Δp_t^{interp}
Δp_t^{fps}
Δp_t^{interp}
GS Variation Field V_t
Pred. GS G_t
Canonical GS Encoder
Canonical GS Decoder
Mesh-guided Interp.
FPS
$\mathcal{L}_{mg}$
Apply to G_1
Render
G_1 Pos. Emb.
Query
G_1 Pos. Emb.
Query
Cross Attention
KV
3D Asset M_1
Surface Sampling
Point Cloud P_1
Flatten
P_1 Pos. Emb.
KV
Cross Attention
Self Attention
$\mathcal{L}_{img}$
Image I_t^{render}
3D Asset M_t
Point Cloud P_t
Point Disp. ΔP_t
Latent z
Image I_t^{gt}
Encoder
Encoder
: Subtraction
: Position Encoding
: Trainable
: Frozen

Results

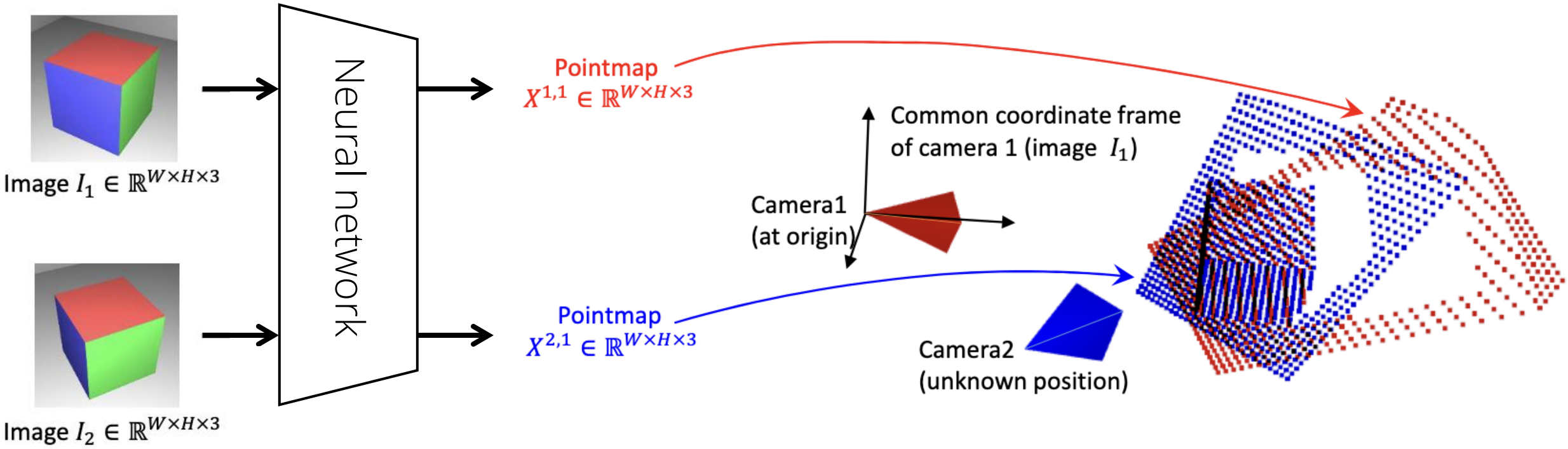
Input video | Consistent4D | STAG4D | L4GM | Ours

Gaussian Variation Field Diffusion for High-fidelity Video-to-4D Synthesis. Zhang et al.

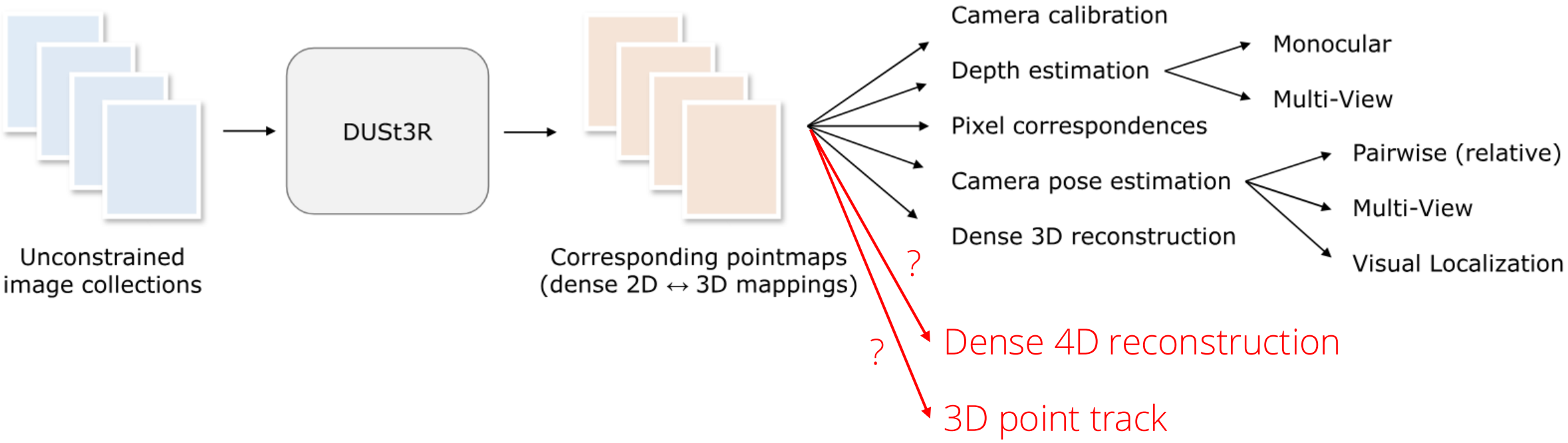
Content

- 2D Motion Representations
- 3D Motion Representations
  - ➢4D Volume Rendering with NERF
  - ➢Explicit Motion Field with NERF
  - ➢4D Volume Rendering with 3DGS
  - ➢Explicit Motion Field with 3DGS
  - ➢Implicit Motion Field with 3DGS
  - ➢4D Foundation Models
  - ➢Multi-view Video Generation (we'll cover in the next lecture)

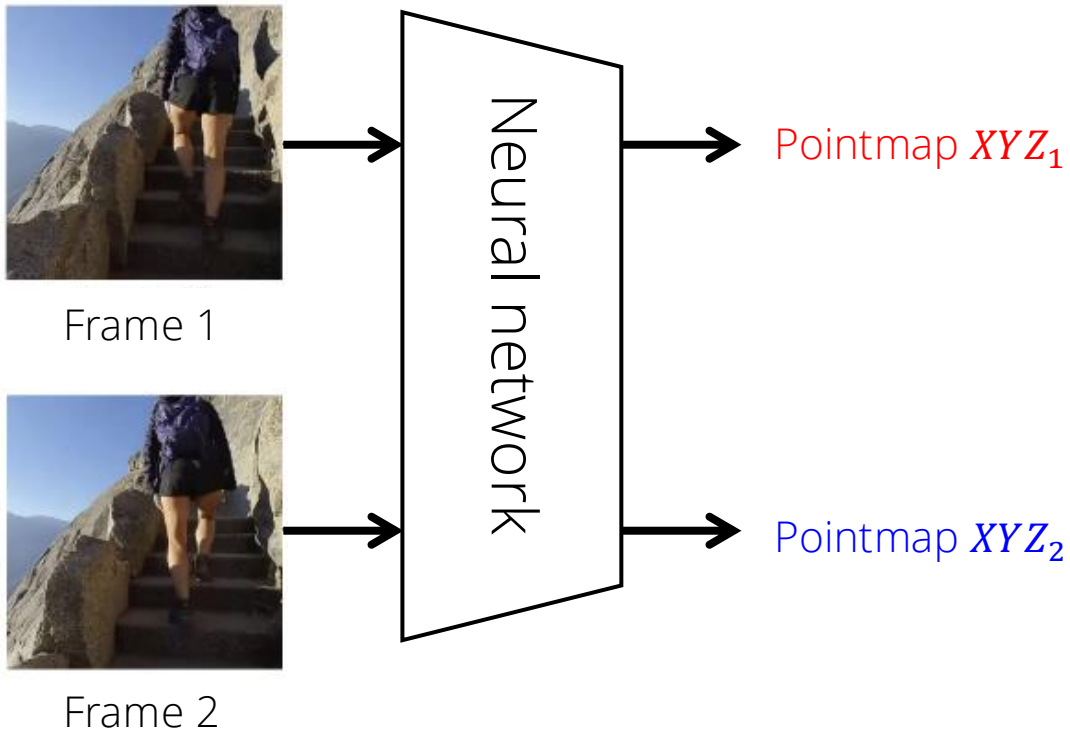
DUSt3R: Dense and Unconstrained Stereo 3D Reconstruction

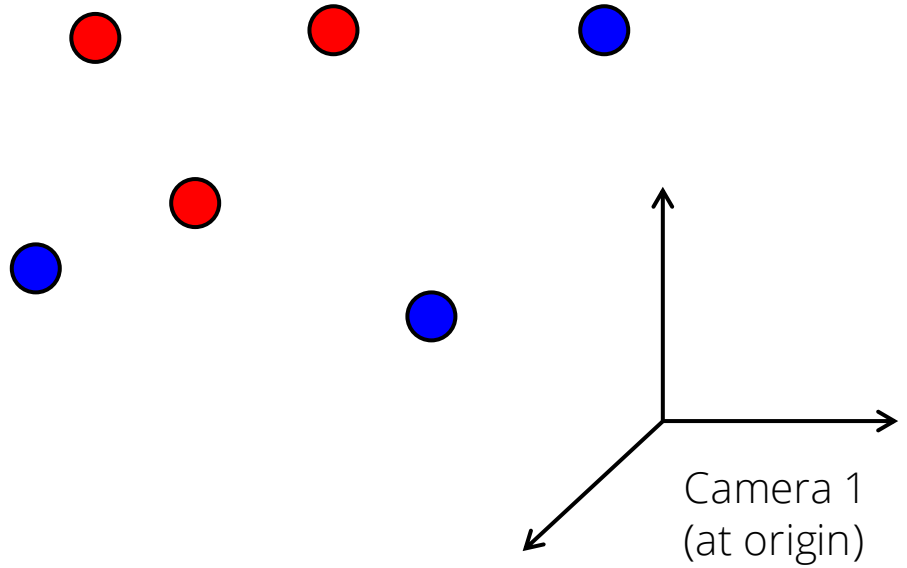


A Model that Solves 3D Reconstruction How About 4D Reconstruction?



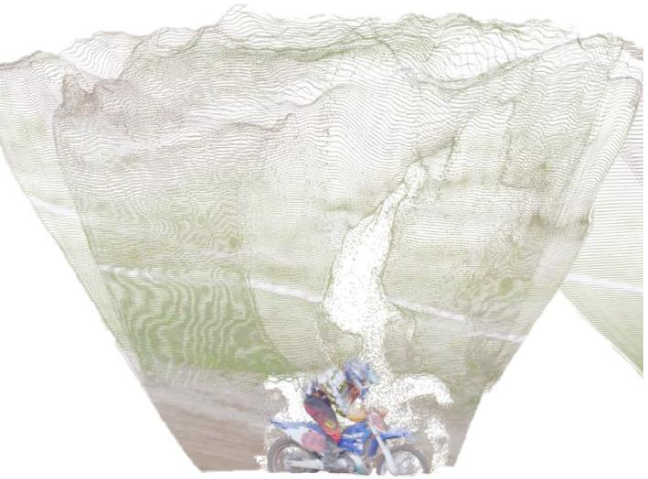
A Naïve Solution: Video Frames as Inputs





This Naïve Approach Doesn't Work Out-of-Box

Issue 1: Alignment based on dynamic object



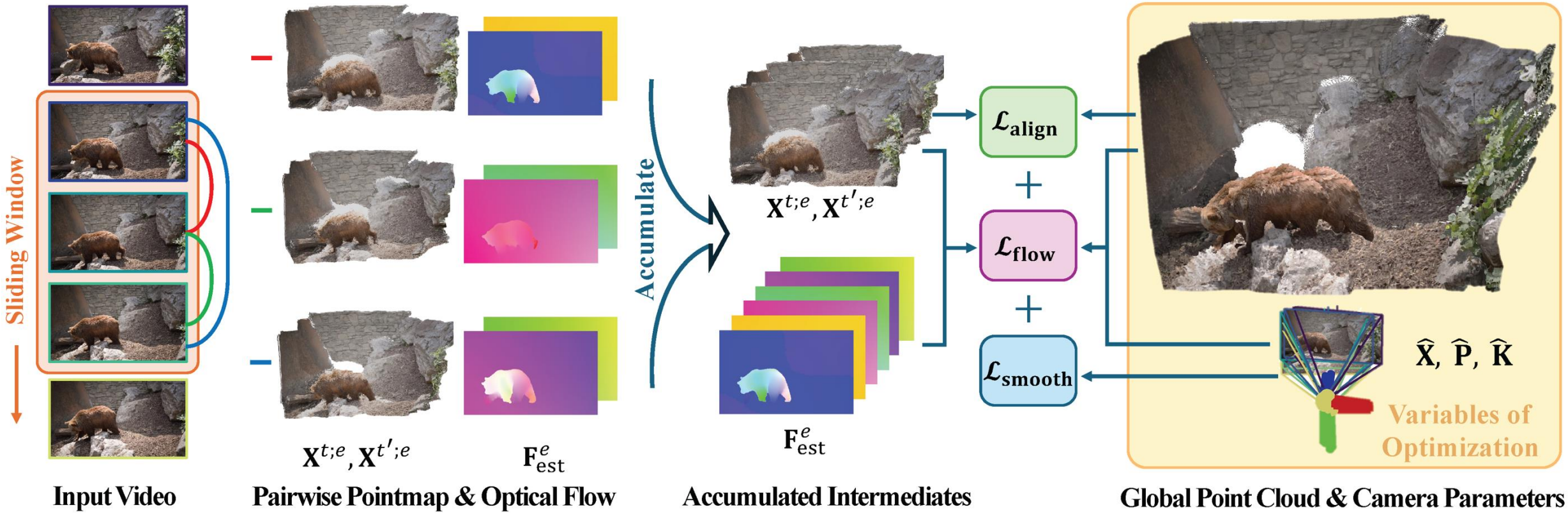
DUSt3R ✗

Issue 2: Foreground depth not estimated correctly

DUSt3R ✗

MonST3R: A Simple Approach for Estimating Geometry in the Presence of Motion. Zhang et al.

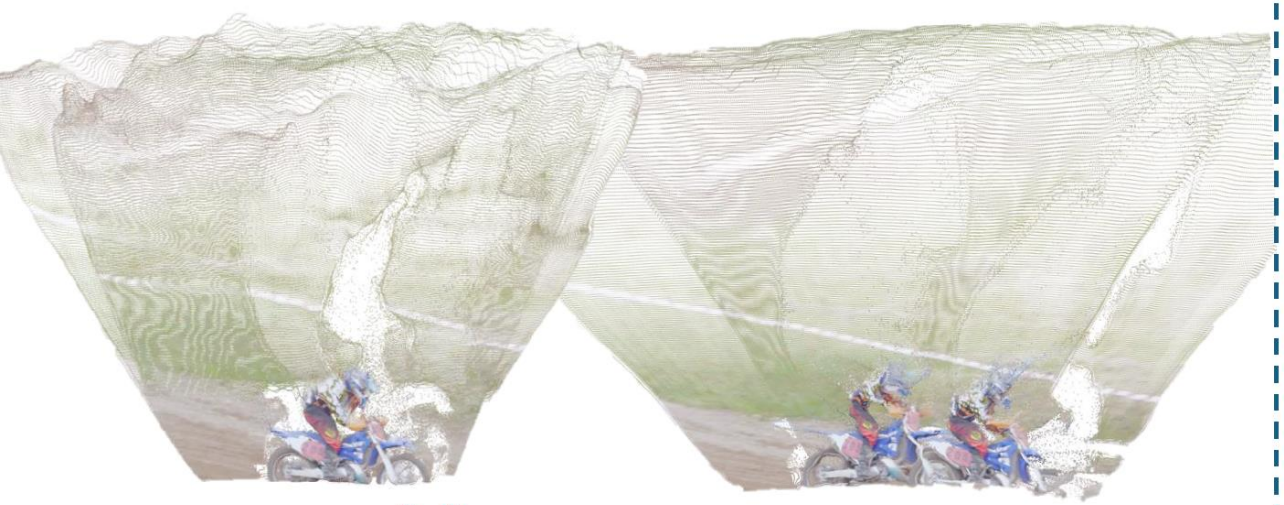
Idea: Globally Align Point Maps Based on Static Points



MonST3R: A Simple Approach for Estimating Geometry in the Presence of Motion. Zhang et al.

Results

Issue 1: Alignment based on dynamic object



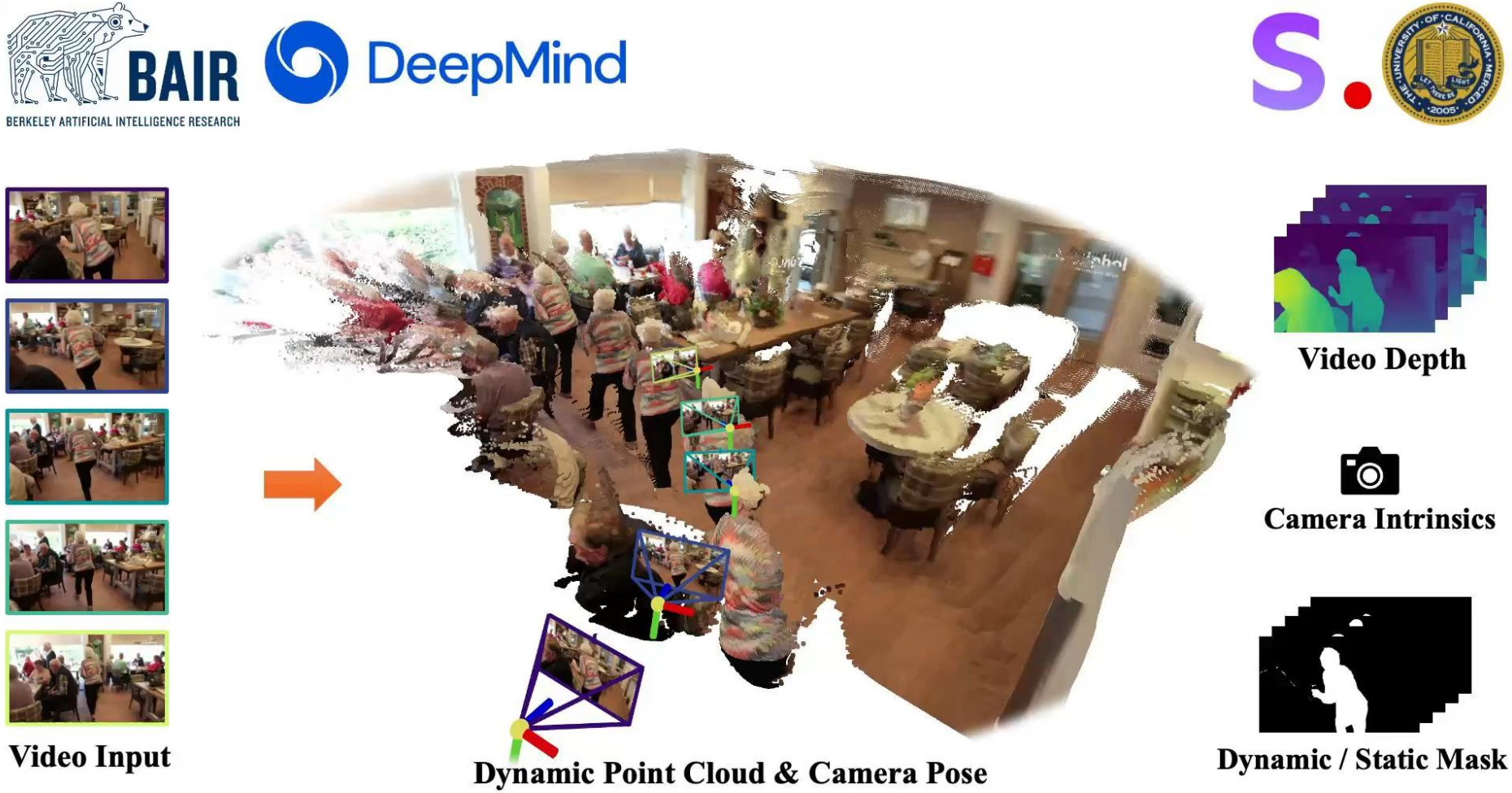
DUSt3R ✗ **MonST3R**

Issue 2: Foreground depth not estimated correctly

DUSt3R ✗ **MonST3R**

MonST3R: A Simple Approach for Estimating Geometry in the Presence of Motion. Zhang et al.

Results



MonST3R: A Simple Approach for Estimating Geometry in the Presence of Motion. Zhang et al.

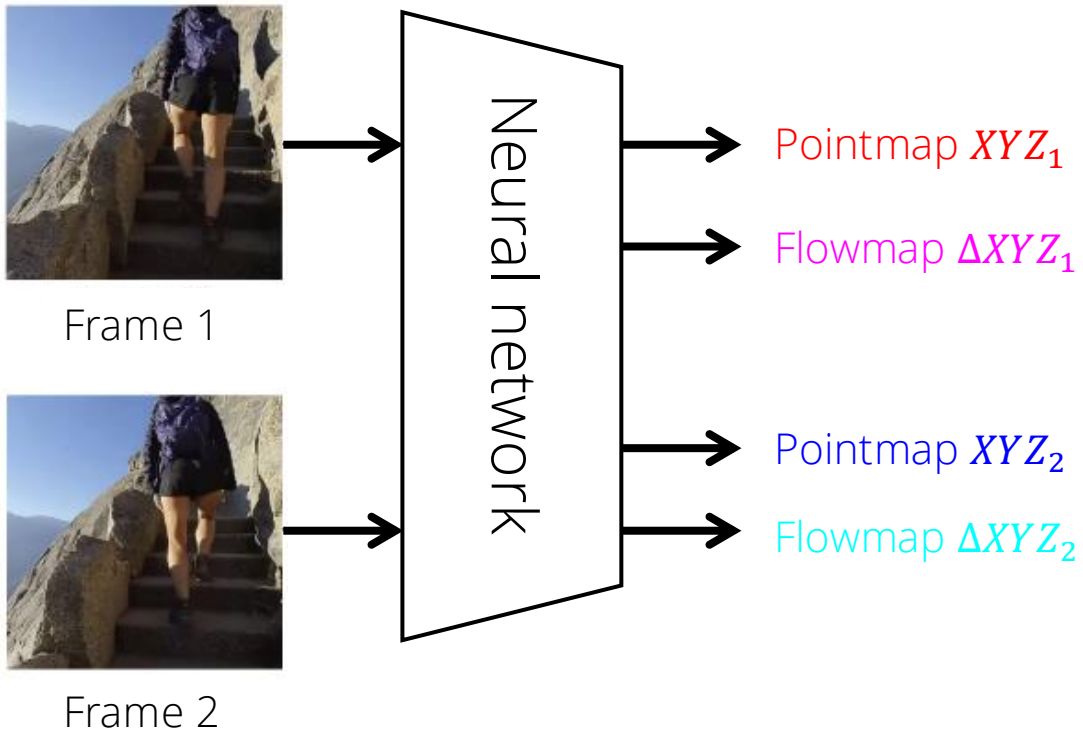
MONSt3R Has Two Problems

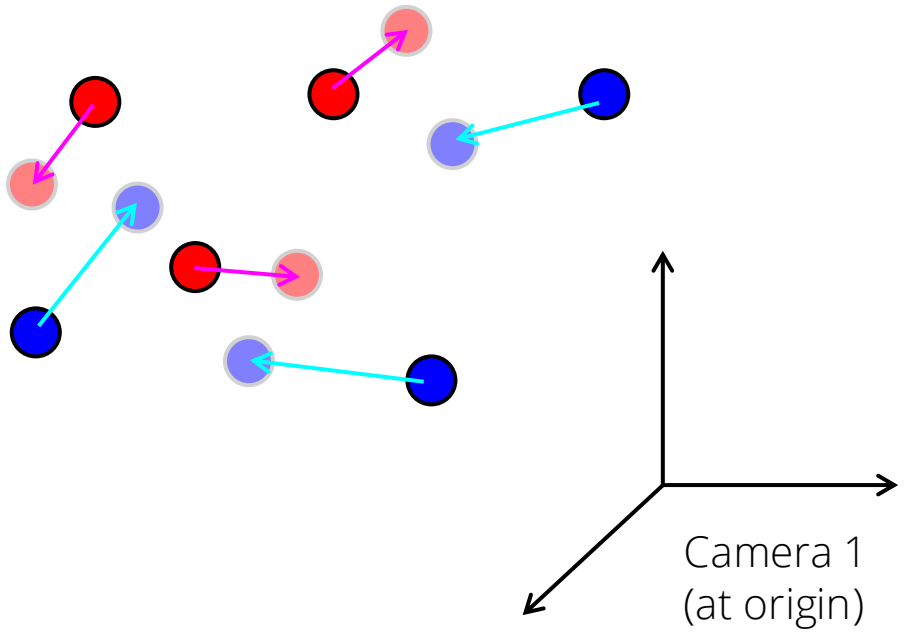
- Reconstructed 4D scenes lack correspondence
- Reconstructing a pair of video frames is inefficient

MONSt3R Has Two Problems

- Reconstructed 4D scenes lack correspondence
- Reconstructing a pair of video frames is inefficient

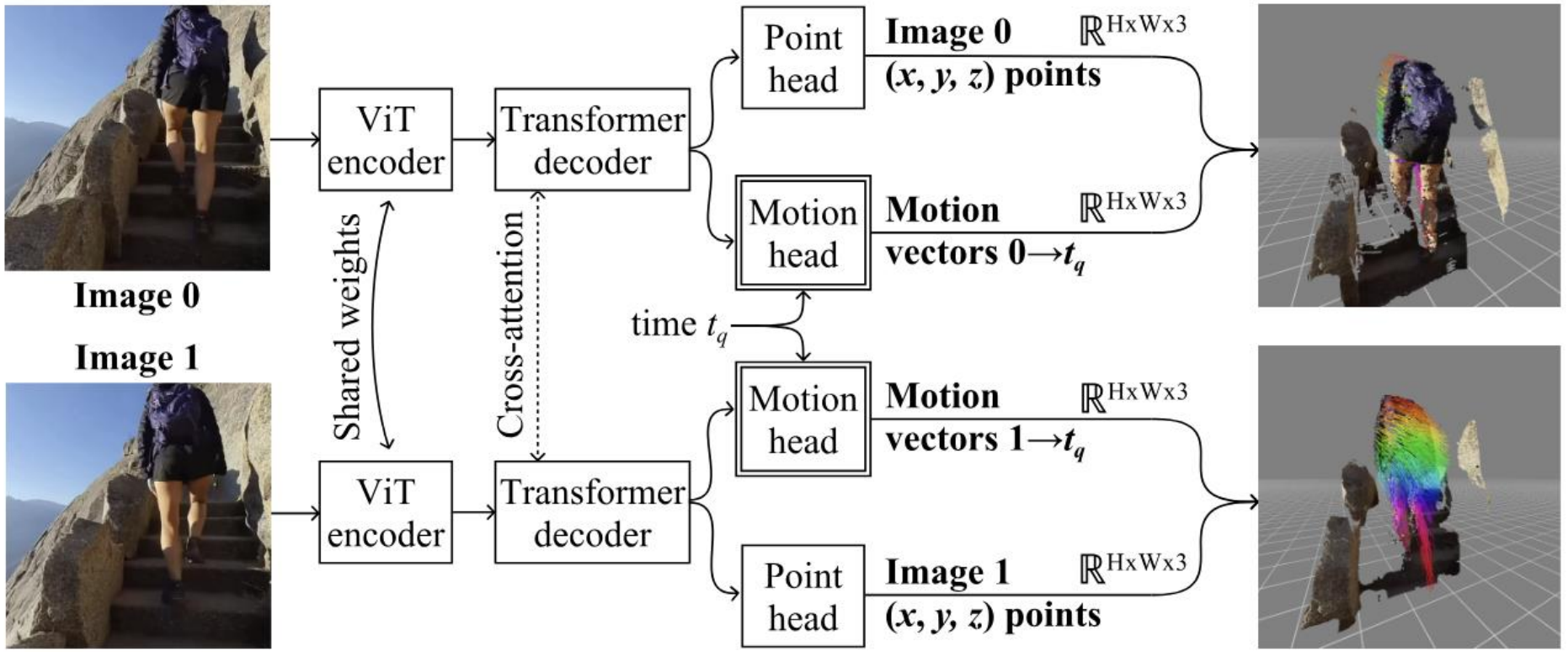
4D Reconstruction with Correspondence





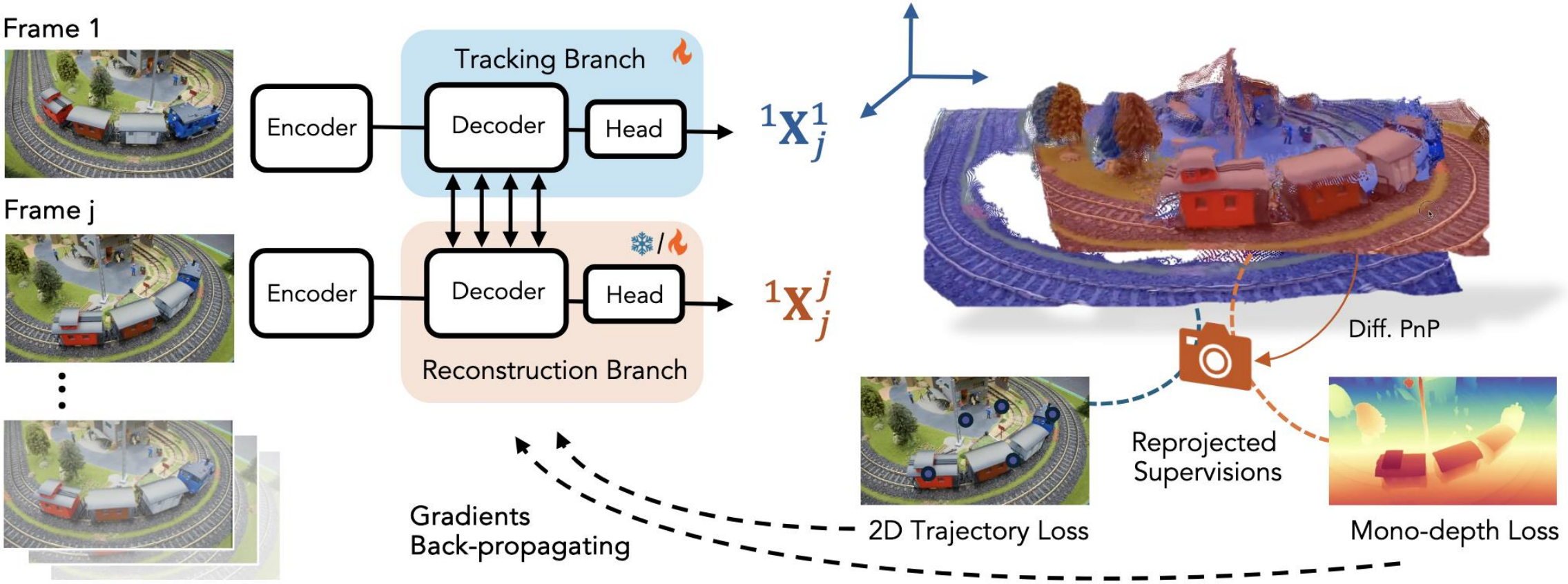
Everyone work on this direction...

For Example, Stereo4D



Stereo4D Learning How Things Move in 3D from Internet Stereo Videos . Jin et al.

For Example, St4RTrack



St4RTrack: Simultaneous 4D Reconstruction and Tracking in the World. Feng et al.

For Example, Dynamic Point Map

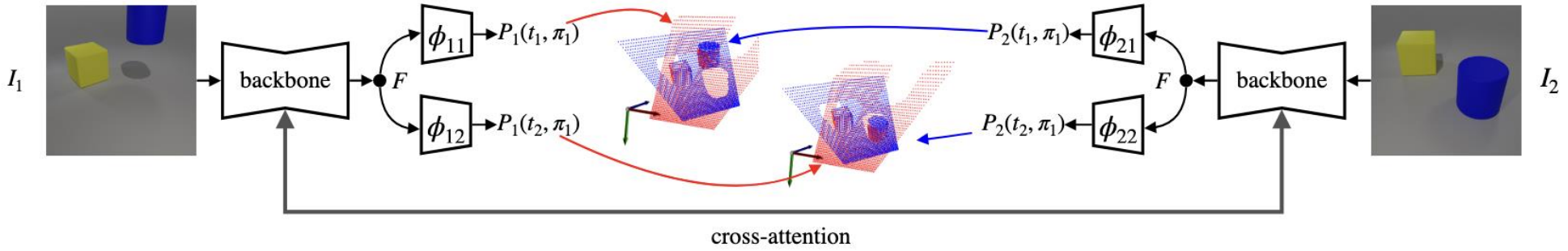


Figure 2. We propose to extend DUSt3R to predict **Dynamic Point Maps** (DPM). Each image in the pair is mapped to two point maps that correspond to the timestamps of the two images (pairs share the same colour in the figure). All points map are defined in the reference frame of image I_1, undoing the effect of viewpoint change. Scene flow and space-time correspondences can be inferred immediately.

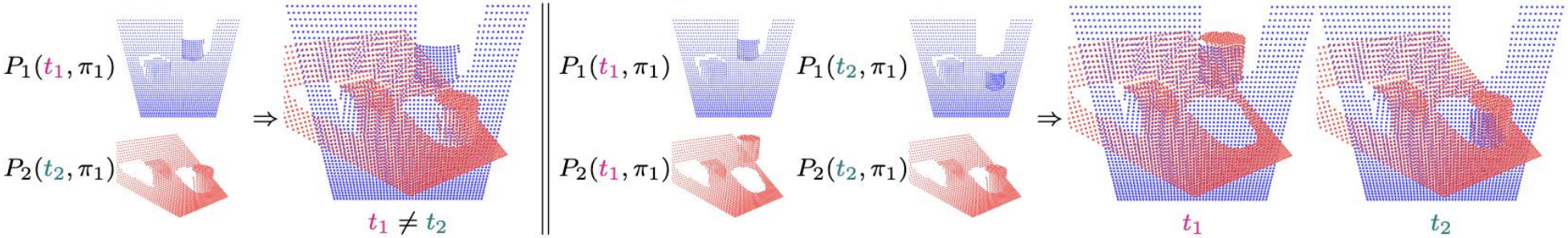


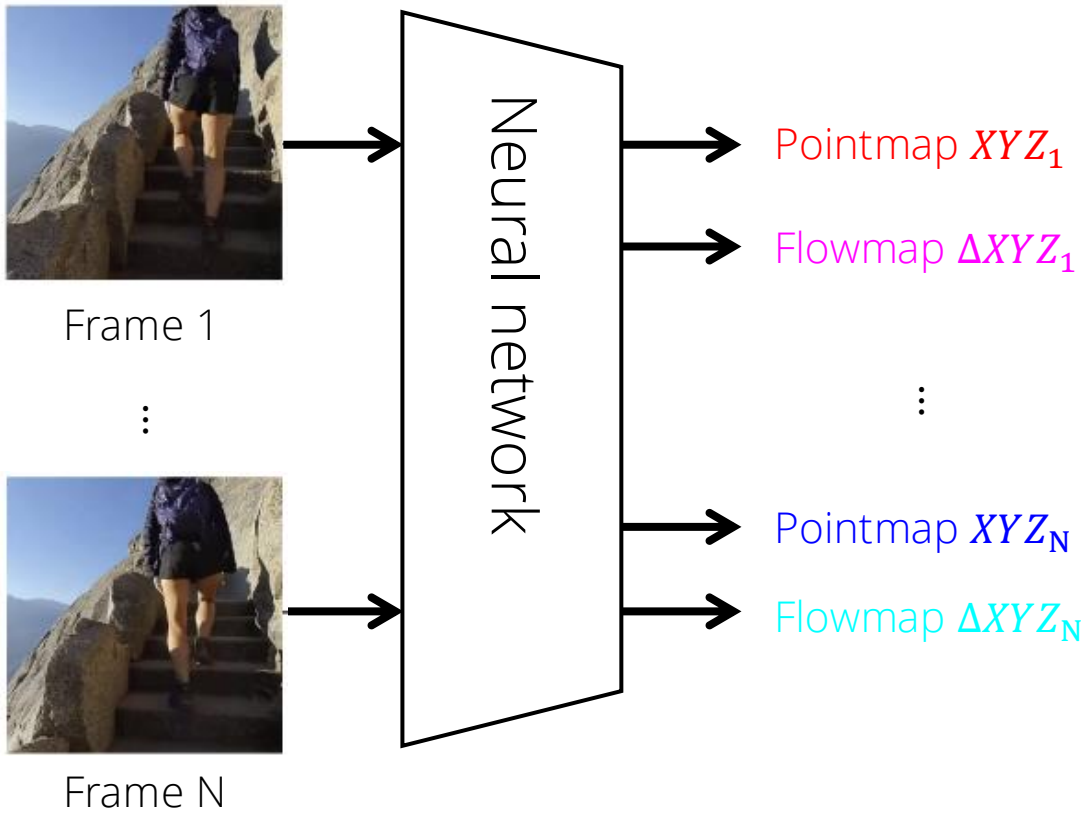
Figure 3. Left: Standard point maps applied to dynamic scenes as in MonST3R [80] fail to represent dynamics. The cylinder, which is moving downwards, breaks invariance when the point maps are overlaid. Right: Our Dynamic Point Maps correctly represent dynamics by also controlling time in addition to viewpoint. They allow to restore invariance while still representing the motion of the cylinder.

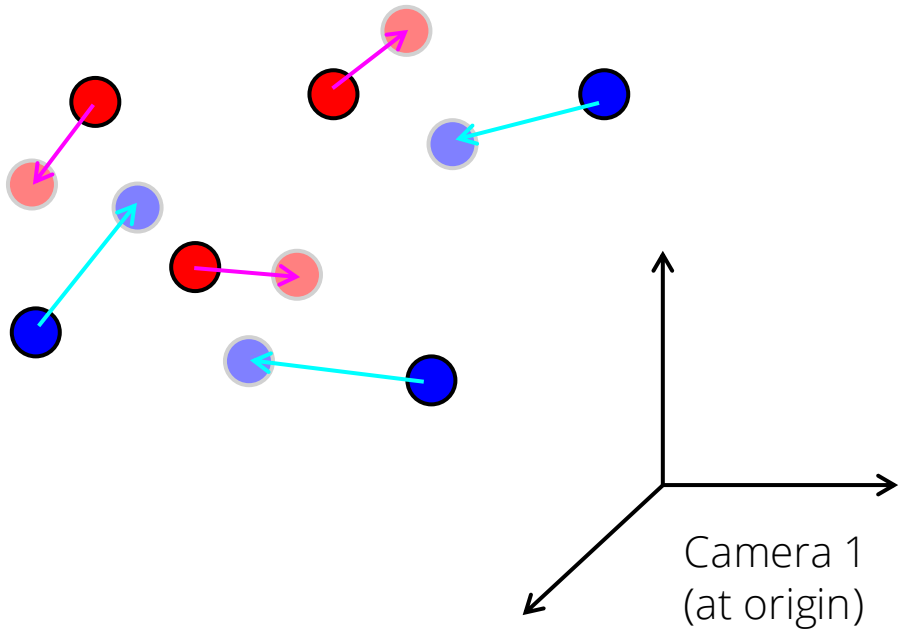
Dynamic Point Maps: A Versatile Representation for Dynamic 3D Reconstruction. Sucar et al.

MONSt3R Has Two Problems

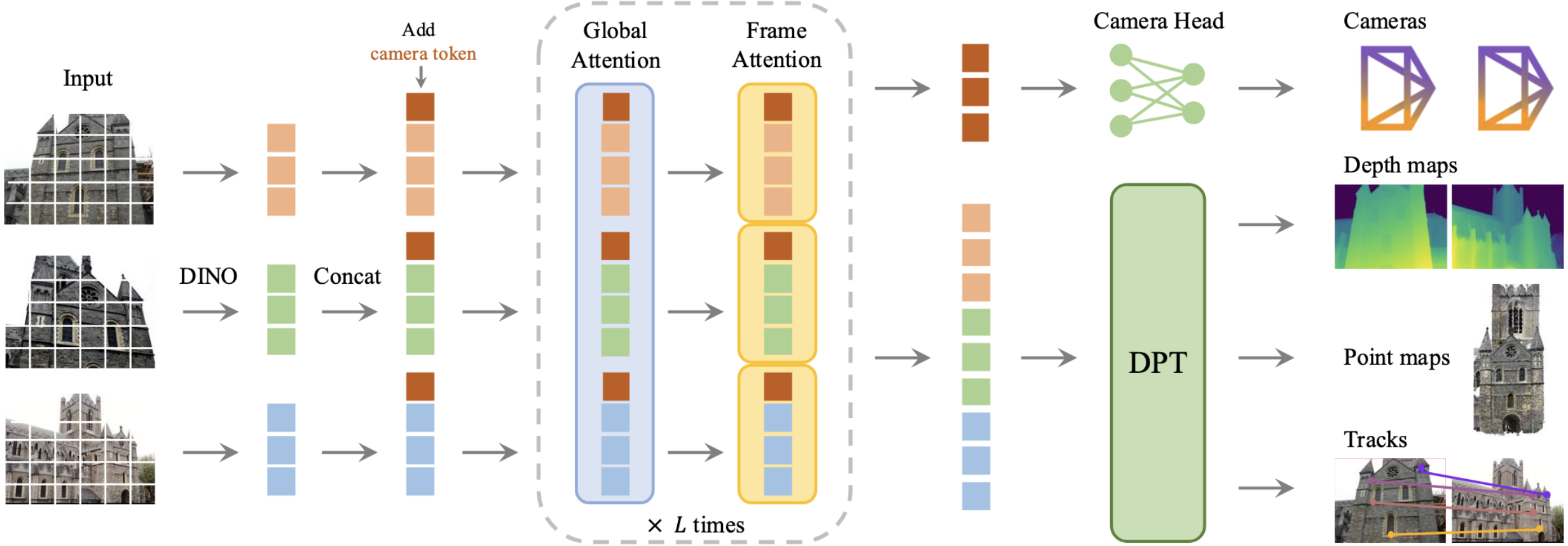
- Reconstructed 4D scenes lack correspondence
- Reconstructing a pair of video frames is inefficient

Idea: Use Large Visual Transformer

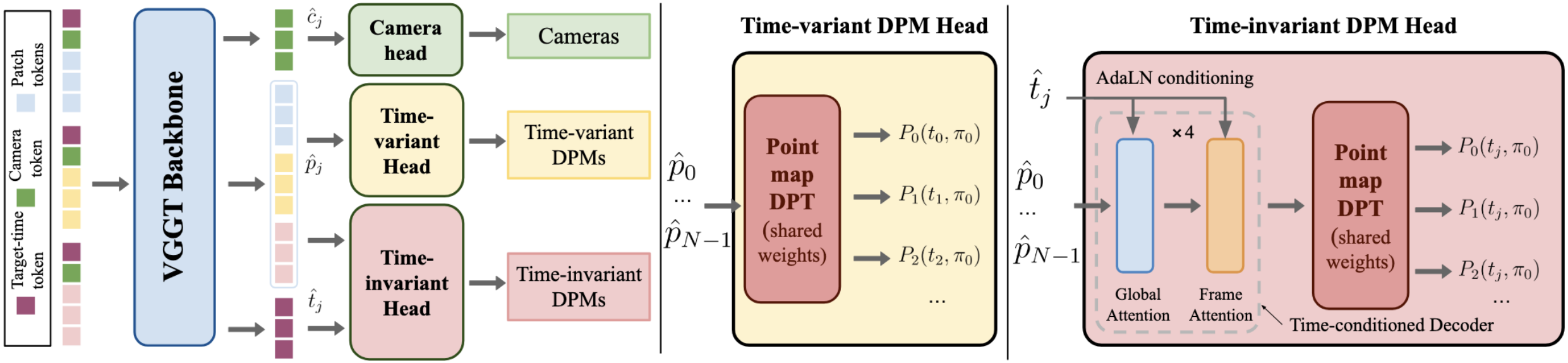




VGGT Goes Beyond Image Pairs



Append a Motion Head to VGGT



V-DPM: 4D Video Reconstruction with Dynamic Point Maps. Sucar et al.

Results

Tennis

Content

- 2D Motion Representations
- 3D Motion Representations
  - ➢4D Volume Rendering with NERF
  - ➢Explicit Motion Field with NERF
  - ➢4D Volume Rendering with 3DGS
  - ➢Explicit Motion Field with 3DGS
  - ➢Implicit Motion Field with 3DGS
  - ➢4D Foundation Models
  - ➢Multi-view Video Generation (we'll cover in the next lecture)

What We Will Cover Next Week

- Generative Modeling