From Paper to Progress: MASt3R-SLAM: Real-Time Dense SLAM with 3D Reconstruction Priors

Guangping Liu, Ravali Gangineni, Rubeena Sultana

Overview

01 **Introduction**

The motivation of this paper

02 **Results**

Videos

03 **Methodology**

04 **Improvement**

Introduction

What is SLAM?

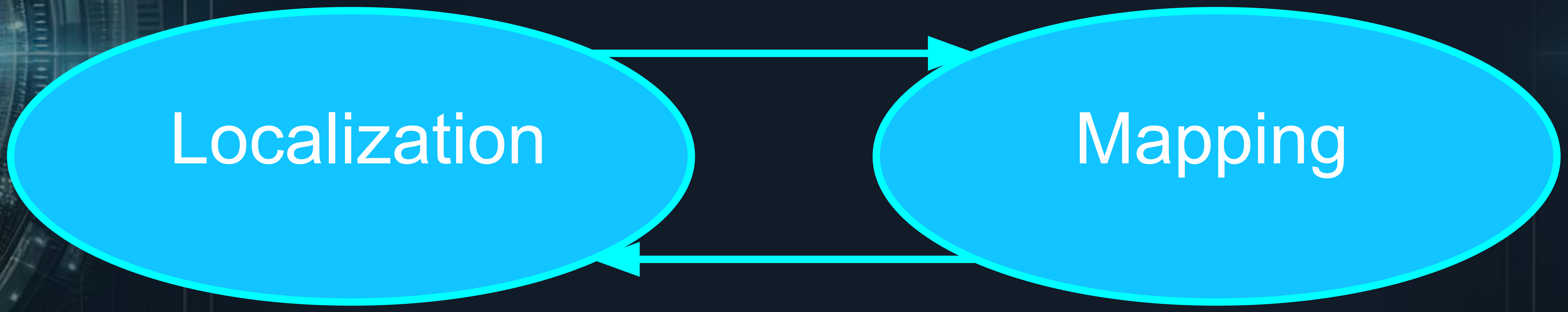
Simultaneous Localization and Mapping (SLAM) is a method used for mobile robots that lets you build a map and localize your vehicle in that map at the same time.

I am a new cleaner at SLU. How should I plan my work to clean the floor of the PLUS XII Library?

-I visited the library to plan my cleaning route.

Introduction

What is SLAM?



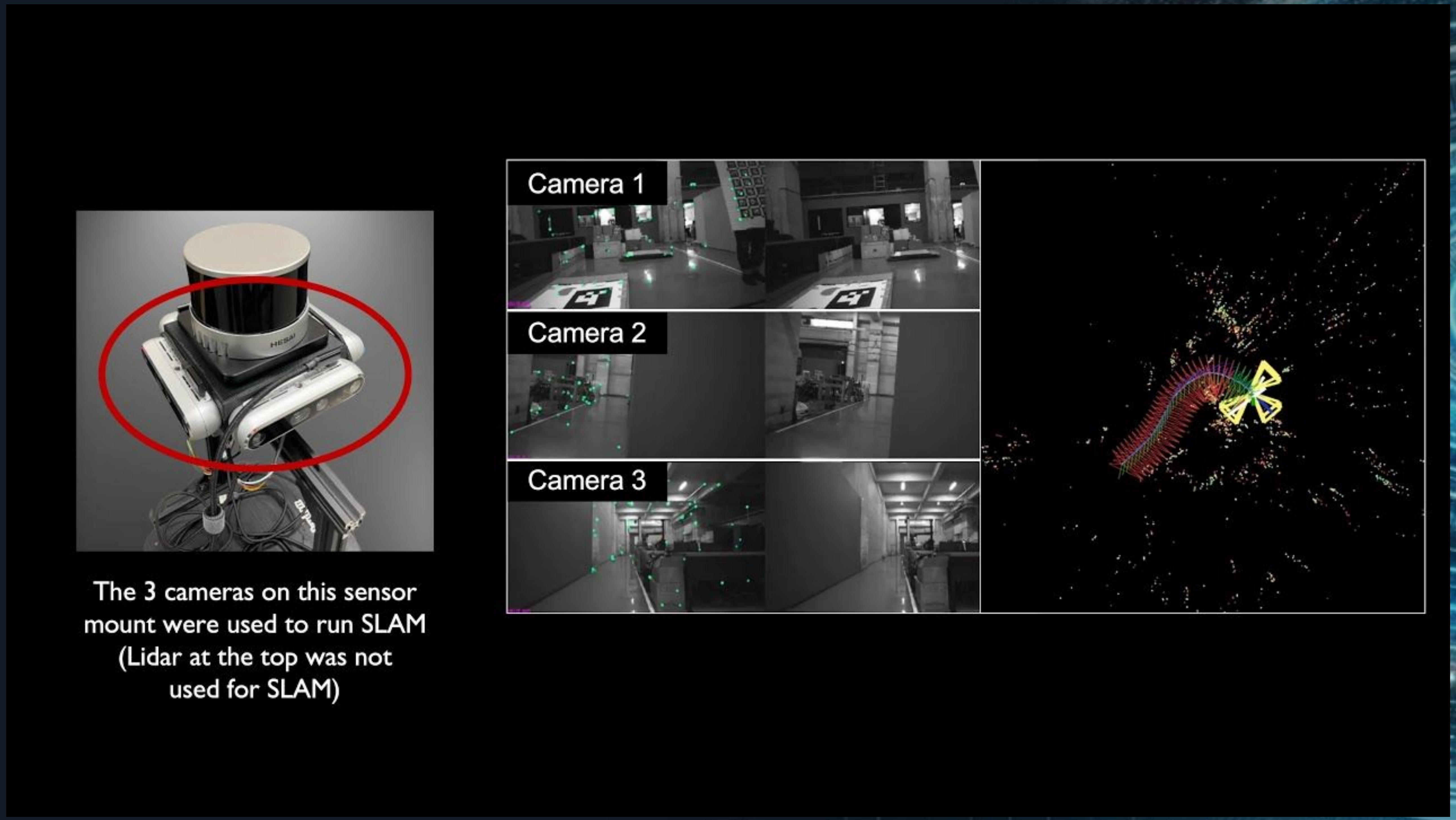
Where I am?

What's around me?

Introduction

What is SLAM?

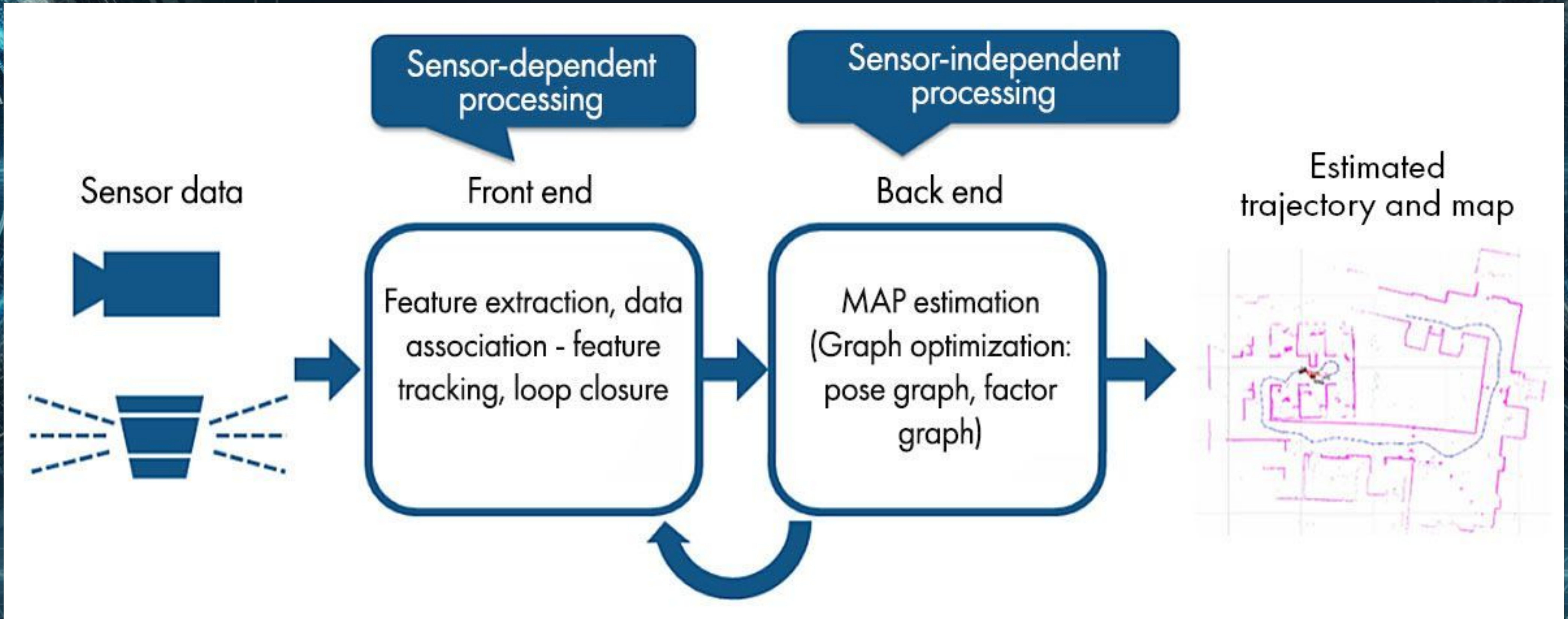
Lidar-based SLAM



Visual SLAM

Introduction

What is SLAM?



Introduction

Gaps in Previous SLAM Systems

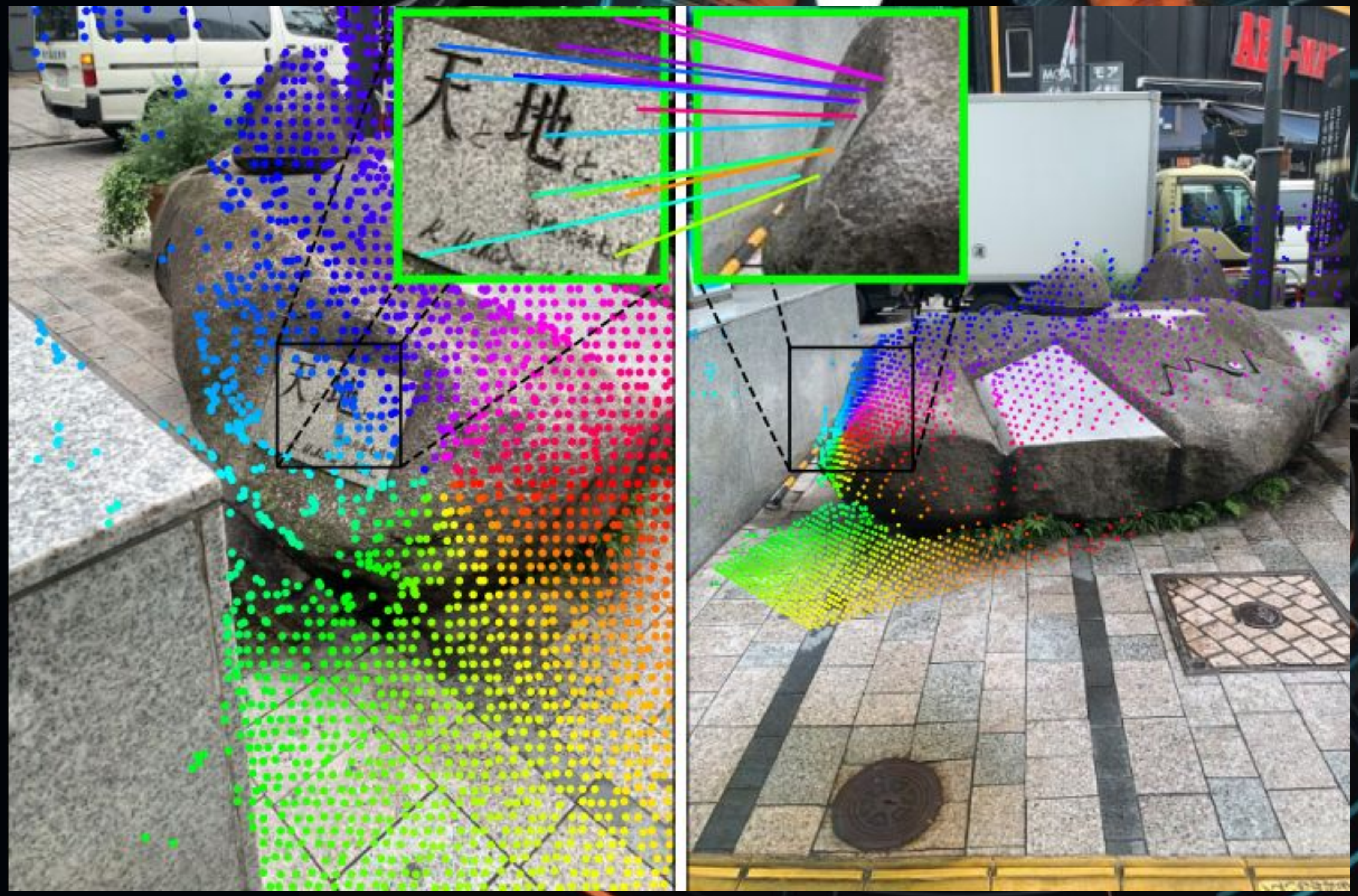
- **Multiple sensors:** Traditional SLAM relies on additional sensors such as IMU to estimate pose and trajectories.
- **Camera Requirements:** Needed Known Camera Settings (Camera Calibration), Fixed Camera Models: Geometry Prediction
- **Dense SLAM:** Many Systems Couldn’t scale to real-time dense 3D mapping
- **Runtime:** learning based systems like MASt3r-sfm were offline and slow

Introduction

Motivation

- Instead of using traditional sub-steps of matching and trangulation, it uses a prior, MASt3R, for an end-to-end prediction of the pointmap.

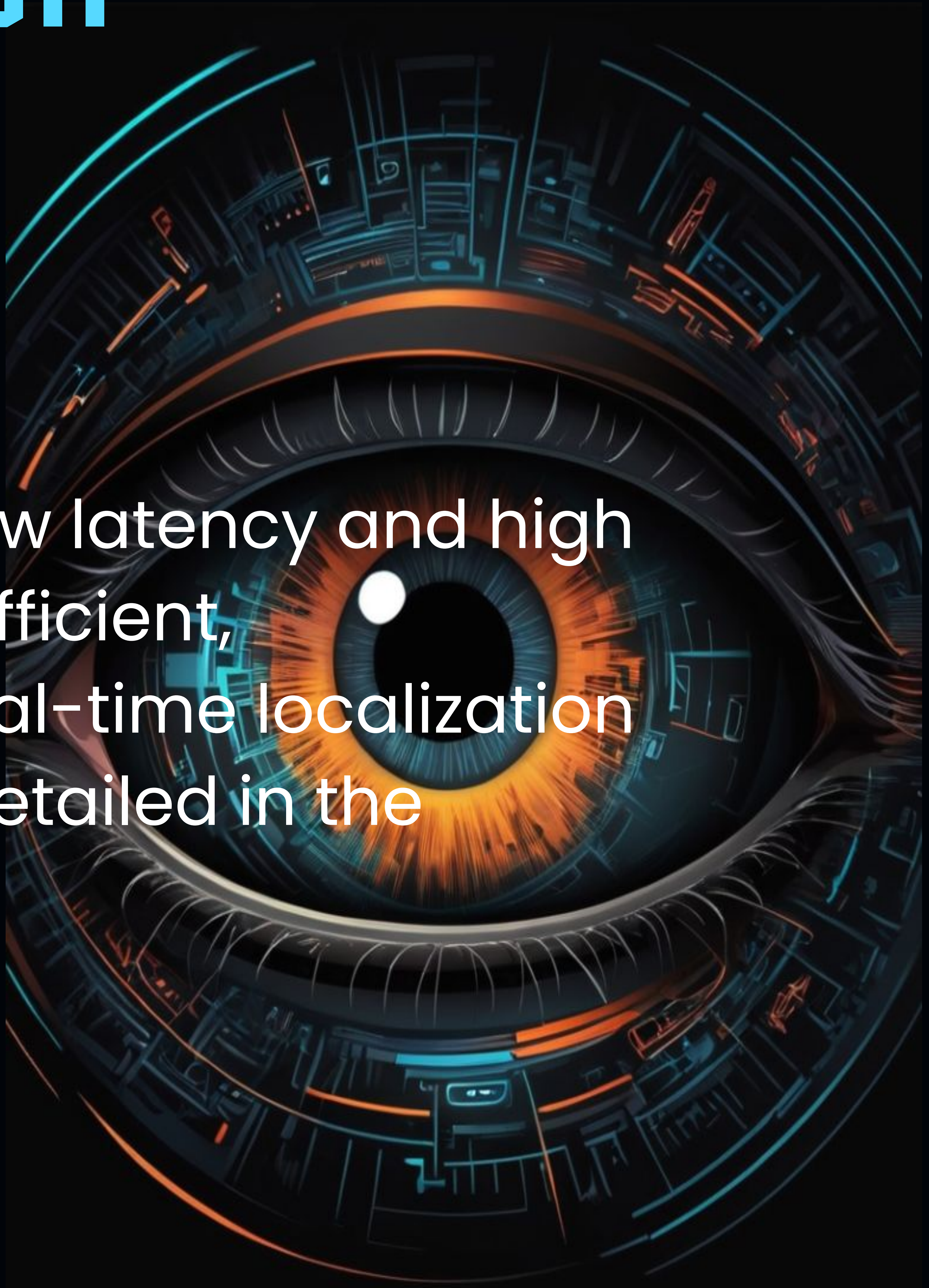
MASt3R, Multi-view Aggregation for Structure-from-Motion with Transformer, is a deep learning model built on the earlier method DUSt3R.



Introduction

Motivation

Considering that SLAM requires low latency and high accuracy, the paper introduces efficient, camera-agnostic methods for real-time localization and map estimation, which are detailed in the Methodology section.



Results

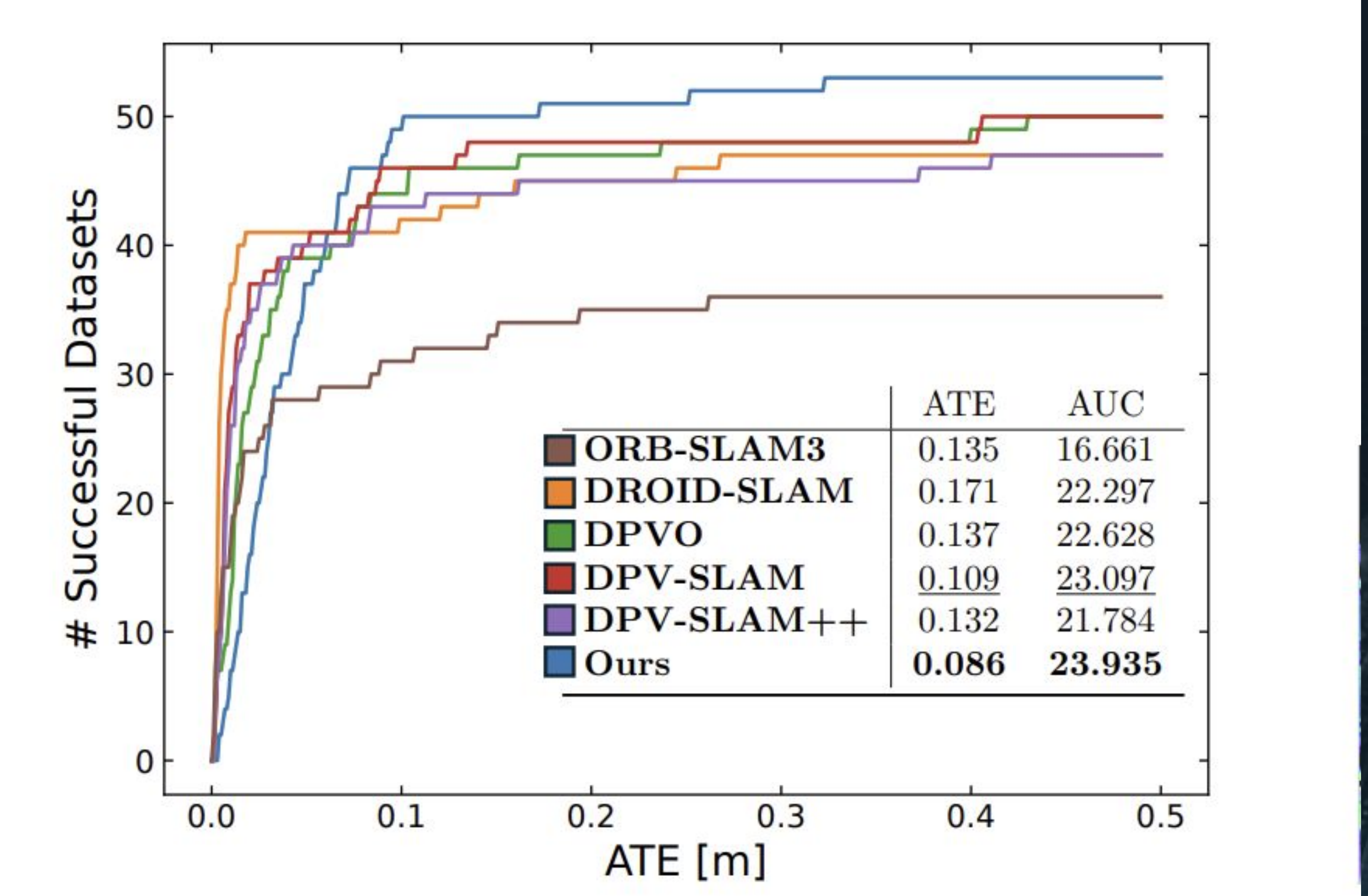
Results

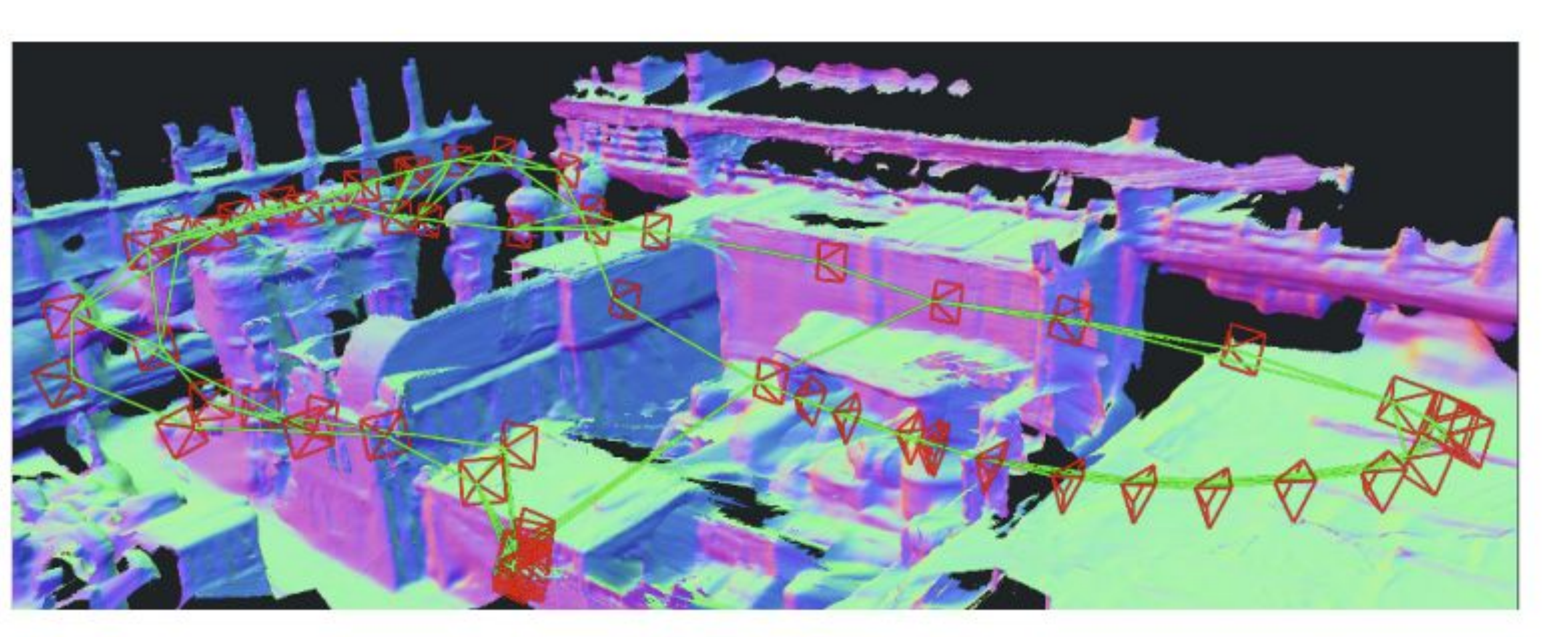
Evaluation metrics for results

- **ATE(Absolute Trajectory Error):** Measures how accurate the estimated camera path is compared to ground truth
- **Chamfer Distance:** Measures how close the reconstructed 3D geometry is to the real world scene
- **Accuracy, completion:** how accurately and completely the scene is reconstructed

What did they Evaluate?

They tested MASt3R- SLAM on 4 real world datasets TUM RGB-D, 7-Scenes, EuRoC MAV, ETH3D-SLAM





This 3D image(EuRoC machine hall) shows the reconstructed point cloud (colored surface) and camera poses (little red pyramids and green lines).

Results

Shows two consecutive frames (just 1 second apart) with extreme zoom change.
Despite no camera calibration, MASt3R-SLAM reconstructs a clean 3D model of the building.
Colored pyramids show camera positions over time.

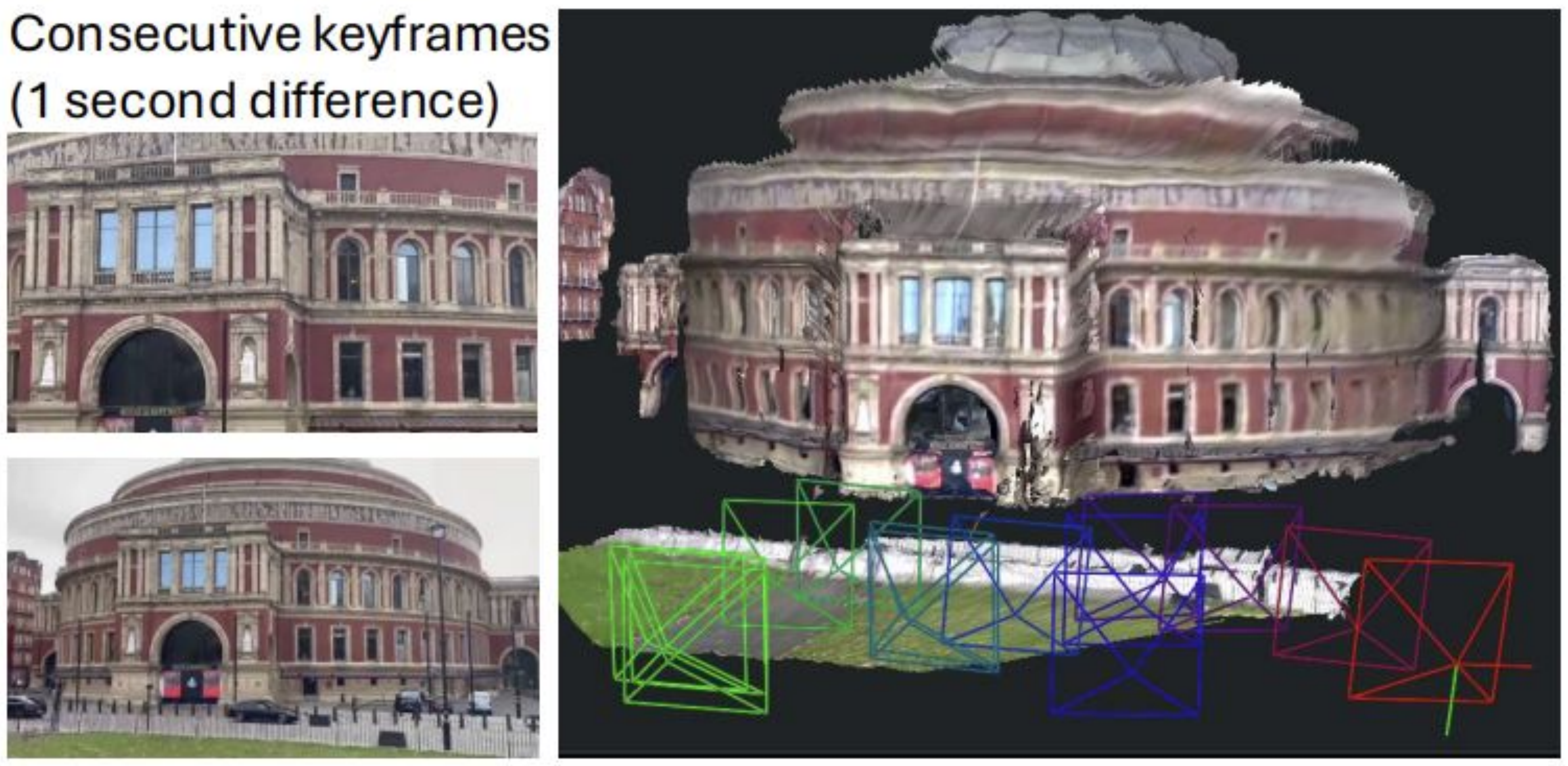
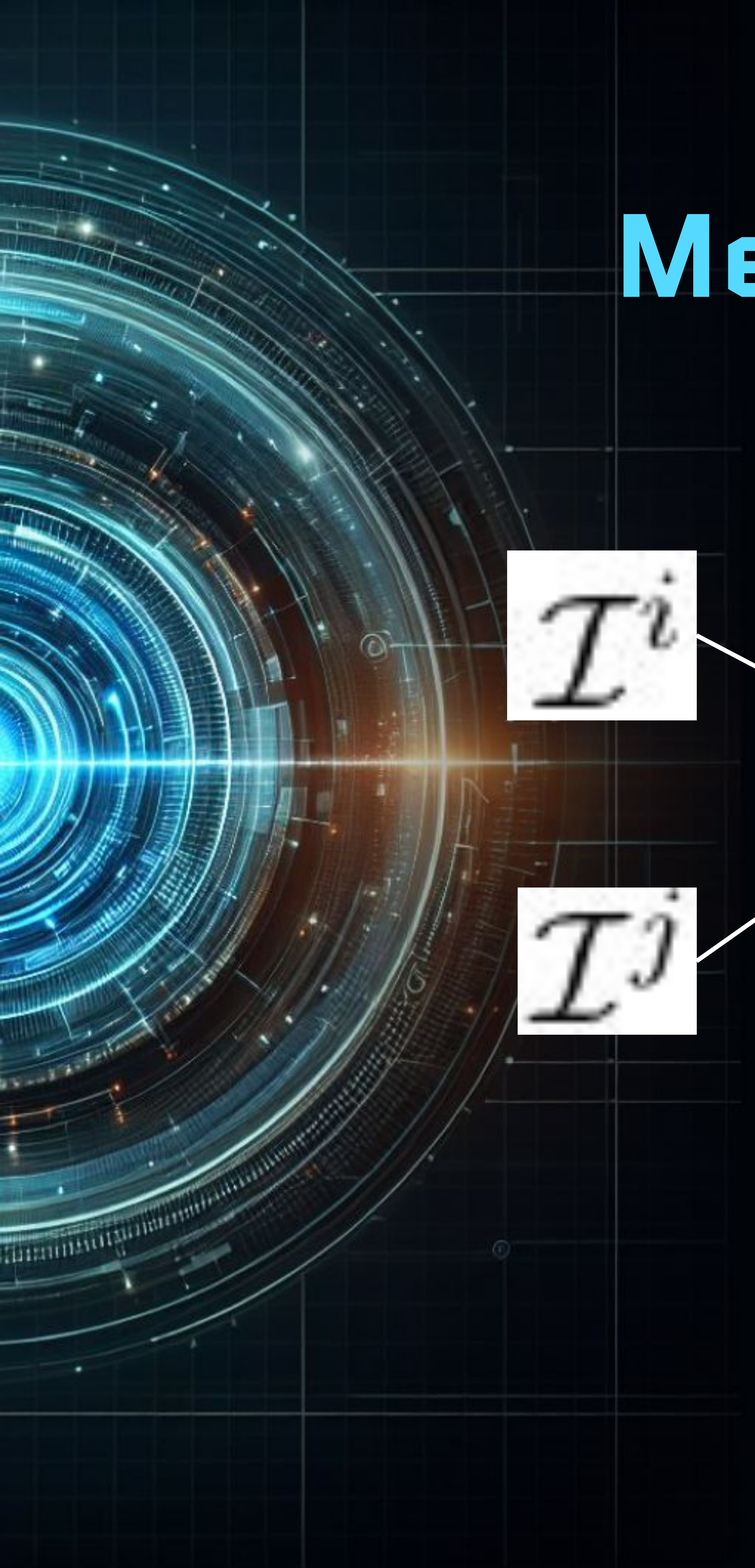


Figure 7. Dense uncalibrated SLAM with extreme zoom changes shown by two consecutive keyframes for an outdoor scene.



Methodology:Preliminaries

$\mathcal{I}^i$, $\mathcal{I}^j$ → MASt3R →

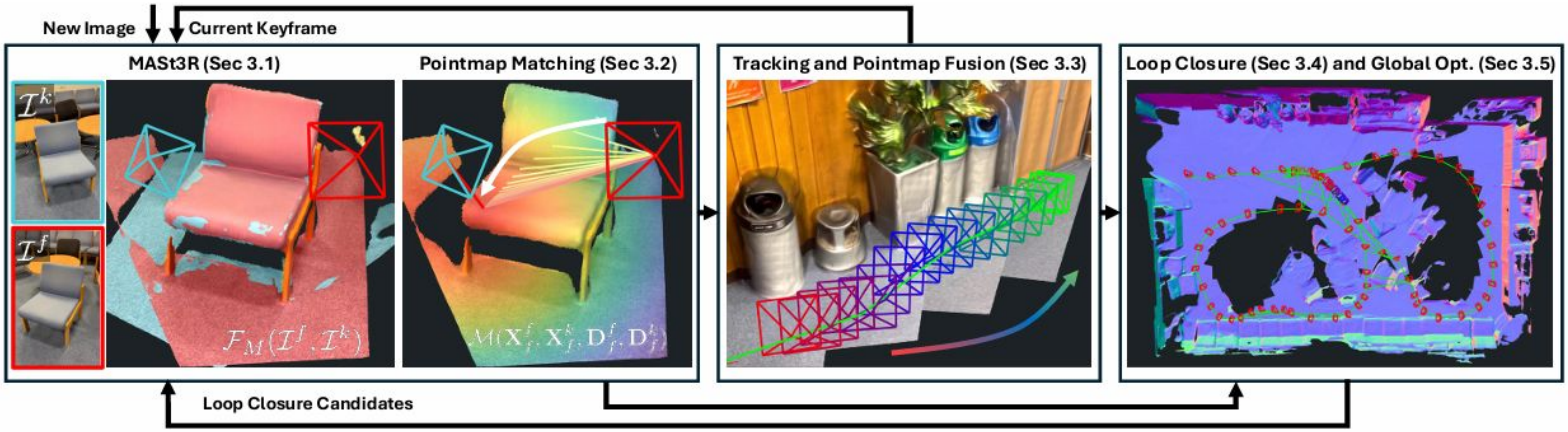
$$\mathbf{X}_i^i, \mathbf{X}_i^j \in \mathbb{R}^{H \times W \times 3}$$

$$\mathbf{C}_i^i, \mathbf{C}_i^j \in \mathbb{R}^{H \times W \times 1}$$

$$\mathbf{D}_i^i, \mathbf{D}_i^j \in \mathbb{R}^{H \times W \times d}$$

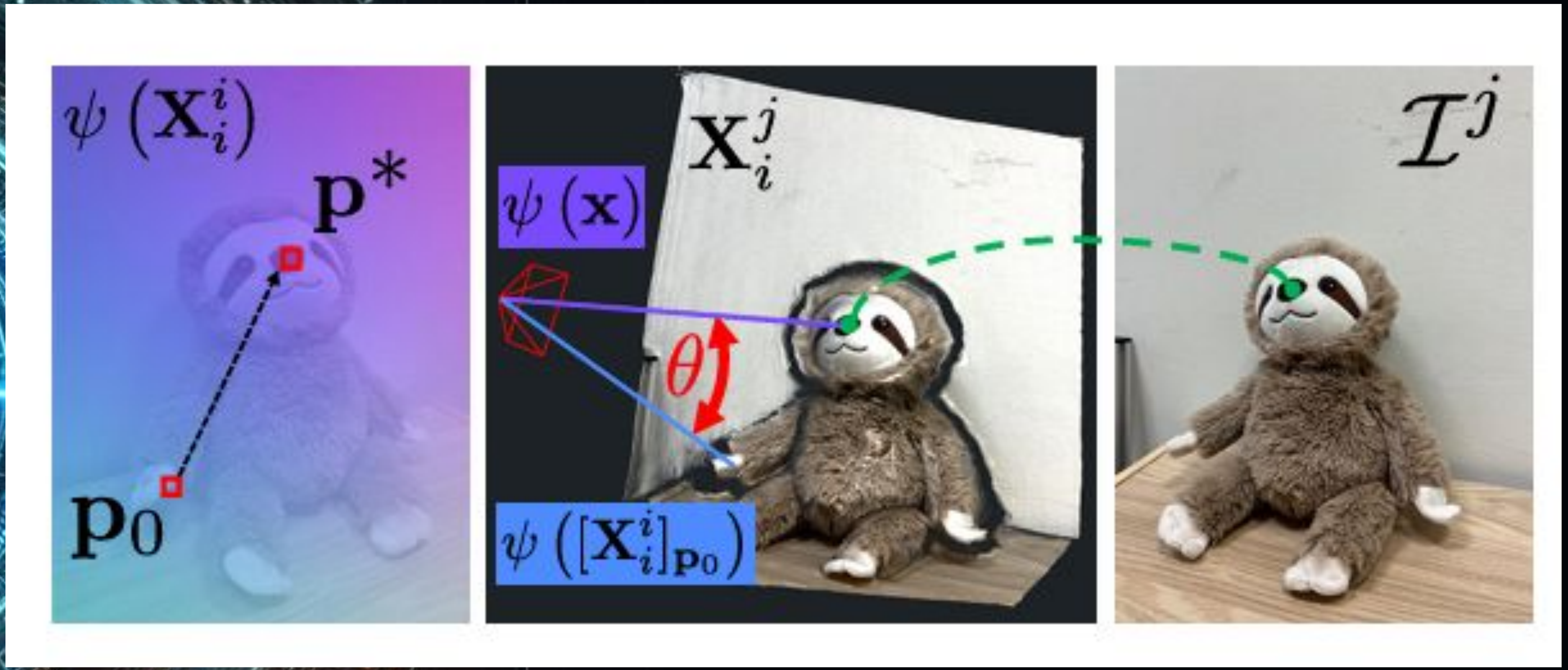
The forward pass of MASt3R is Fm[Ii, Ij]

Methodology:Preliminaries



Methodology:PointMap Matching

$$\mathbf{m}_{i,j} = \mathcal{M}(\mathbf{X}_i^i, \mathbf{X}_i^j, \mathbf{D}_i^i, \mathbf{D}_i^j)$$

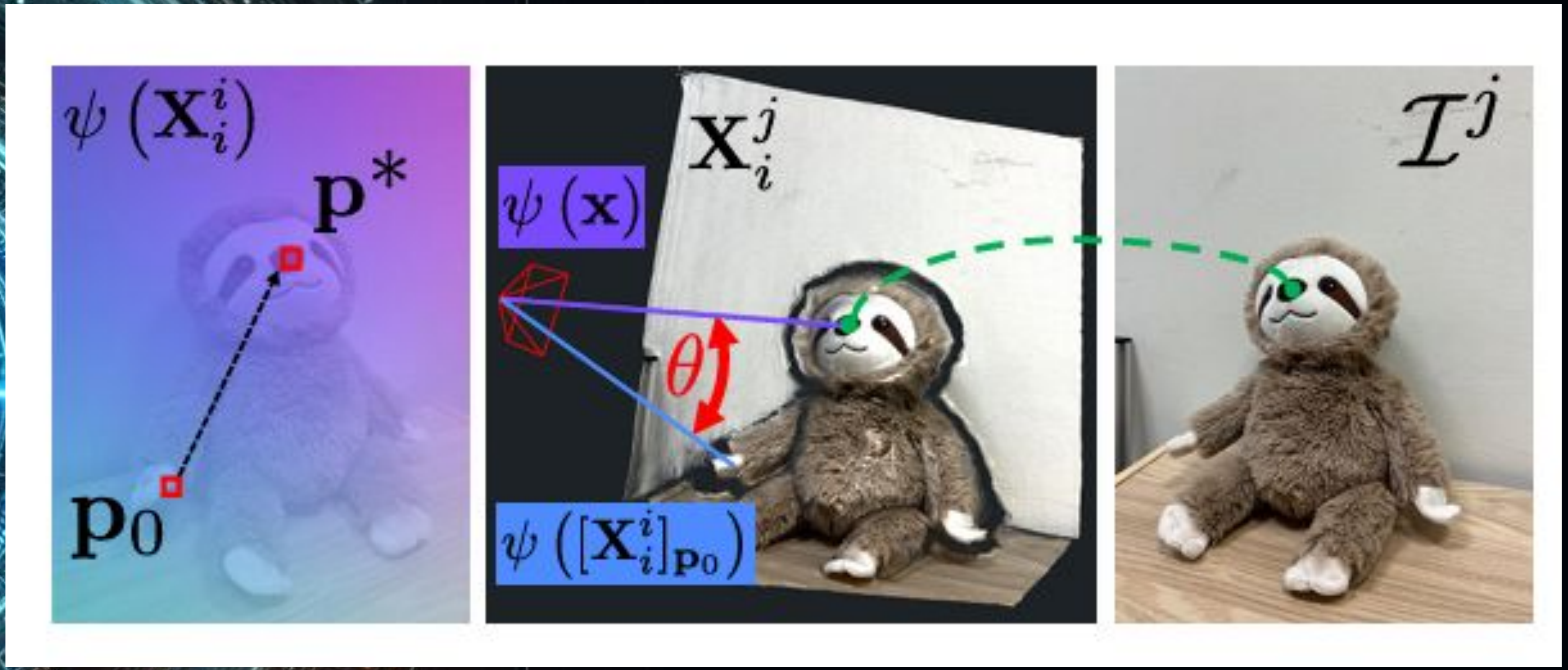


Cost Function:

$$\mathbf{p}^* = \arg\min_{\mathbf{p}} \left\| \psi\left([\mathbf{X}_i^i]_{\mathbf{p}}\right) - \psi\left(\mathbf{x}\right) \right\|^2$$

There is no closed-form projection.
Assumption: **Each frame has a unique camera center**

Methodology:PointMap Matching



Cost Function:

$$\mathbf{p}^* = \arg\min_{\mathbf{p}} \left\| \psi\left([\mathbf{X}_i^i]_{\mathbf{p}}\right) - \psi\left(\mathbf{x}\right) \right\|^2$$

How to represent the error? The difference of two rays can be converted into the theta by the following equation:

$$\|\psi_1 - \psi_2\|^2 = 2(1 - \cos\theta), \quad \cos\theta = \psi_1^T \psi_2.$$

Since it is not a linear equation, how can we minimize the error? The paper uses Levenberg-Marquardt which is for non-linear least square.

Methodology:PointMap Matching

What is Levenberg-Marquardt?

The primary application of the Levenberg–Marquardt algorithm is in the least-squares curve fitting problem: given a set of m empirical pairs (x_i, y_i) of independent and dependent variables, find the parameters $\boldsymbol{\beta}$ of the model curve $f(x, \boldsymbol{\beta})$ so that the sum of the squares of the deviations $S(\boldsymbol{\beta})$ is minimized:

$$\hat{\boldsymbol{\beta}} \in \operatorname{argmin}_{\boldsymbol{\beta}} S(\boldsymbol{\beta}) \equiv \operatorname{argmin}_{\boldsymbol{\beta}} \sum_{i=1}^{m} \left[y_i - f(x_i, \boldsymbol{\beta})\right]^2,$$ which is assumed to be non-empty.

Add a little increment

$$f(x_i, \boldsymbol{\beta} + \boldsymbol{\delta}) \approx f(x_i, \boldsymbol{\beta}) + \mathbf{J}_i \boldsymbol{\delta}$$

$$\mathbf{J}_i = \frac{\partial f(x_i, \boldsymbol{\beta})}{\partial \boldsymbol{\beta}}$$

Calculate the cost function with the increment

$$\begin{aligned} S(\boldsymbol{\beta} + \boldsymbol{\delta}) &\approx \left\|\mathbf{y} - \mathbf{f}(\boldsymbol{\beta}) - \mathbf{J}\boldsymbol{\delta}\right\|^2 \\ &= \left[\mathbf{y} - \mathbf{f}(\boldsymbol{\beta}) - \mathbf{J}\boldsymbol{\delta}\right]^{\mathrm{T}} \left[\mathbf{y} - \mathbf{f}(\boldsymbol{\beta}) - \mathbf{J}\boldsymbol{\delta}\right] \\ &= \left[\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})\right]^{\mathrm{T}} \left[\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})\right] - \left[\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})\right]^{\mathrm{T}} \mathbf{J}\boldsymbol{\delta} - (\mathbf{J}\boldsymbol{\delta})^{\mathrm{T}} \left[\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})\right] + \boldsymbol{\delta}^{\mathrm{T}} \mathbf{J}^{\mathrm{T}} \mathbf{J}\boldsymbol{\delta} \\ &= \left[\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})\right]^{\mathrm{T}} \left[\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})\right] - 2\left[\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})\right]^{\mathrm{T}} \mathbf{J}\boldsymbol{\delta} + \boldsymbol{\delta}^{\mathrm{T}} \mathbf{J}^{\mathrm{T}} \mathbf{J}\boldsymbol{\delta}. \end{aligned}$$

Finally , calculate the derivative

$$\left(\mathbf{J}^{\mathrm{T}}\mathbf{J}\right)\boldsymbol{\delta} = \mathbf{J}^{\mathrm{T}}\left[\mathbf{y} - \mathbf{f}(\boldsymbol{\beta})\right]$$

Methodology:PointMap Matching

With a cost function:

$$\min_{\mathbf{p}} \frac{1}{2} \|\mathbf{r}(\mathbf{p})\|^2$$

$$\mathbf{r}(\mathbf{p}^{(k)} + \delta\mathbf{p}) \approx \mathbf{r}(\mathbf{p}^{(k)}) + \mathbf{J}(\mathbf{p}^{(k)})\,\delta\mathbf{p}$$

$$\nabla_{\delta\mathbf{p}} \|\mathbf{r}^{(k)} + \mathbf{J}\,\delta\mathbf{p}\|^2 = 2\,\mathbf{J}^\top \left(\mathbf{r}^{(k)} + \mathbf{J}\,\delta\mathbf{p}\right) = \mathbf{0}$$

$$\delta\mathbf{p} = -\left(\mathbf{J}^\top \mathbf{J}\right)^{-1} \mathbf{J}^\top \mathbf{r}^{(k)}$$

Tracking the camera pose

Track the camera's position (pose) with respect to a reference keyframe (the last keyframe) in real-time(relative transformation).

The system relies on the point maps generated from two images to track the camera's movement.

Optimization

Equation for 3D Point Error:

$$E_p = \sum_{m,n \in \mathbf{m}_{f,k}} \left\| \frac{\tilde{\mathbf{X}}^k_{k,n} - \mathbf{T}_{kf}\mathbf{X}^f_{f,m}}{w(\mathbf{q}_{m,n}, \sigma_p^2)} \right\|_\rho ,$$

The error between the predicted 3D point location and the actual position of the camera is minimized to estimate the pose.

The error is weighted by confidence values from MASt3R and robust norms like Huber norm are used to make the system more resilient to outliers.

Pixel Wise alignment Vs Explicit matching

The paper uses explicit matching of points between the current frame and the keyframe to improve accuracy, especially when there is a larger baseline (greater camera movement between frames).

Fusing Predictions:
The system averages multiple pointmap predictions into a single pointmap.

Error accumulation:
Tracking errors due to depth inaccuracies accumulate during fusion.

Impact on Keyframe's Pointmap:
These errors degrade the keyframe's pointmap.

The degraded keyframe affects the backend map and overall pose estimation, reducing accuracy.

Ray based error for improved accuracy

Instead of directly using depth values, the system converts 3D points to rays.

Under the assumption that the camera is central, the system treats each point as a ray originating from the camera center.

equation for ray based error:

$$E_r = \sum_{m,n \in \mathbf{m}_{f,k}} \left\| \frac{\psi\left(\tilde{\mathbf{X}}^k_{k,n}\right) - \psi\left(\mathbf{T}_{kf}\mathbf{X}^f_{f,m}\right)}{w(\mathbf{q}_{m,n}, \sigma_r^2)} \right\|_\rho .$$

Advantage: Less sensitive to depth inaccuracies: Angular error between rays is more robust to depth discrepancies than 3D point error.

Pose Update via Gauss-Newton Optimization

The system computes the update to the pose by solving the linear system

$$(\mathbf{J}^T\mathbf{W}\mathbf{J})\,\boldsymbol{\tau} = -\mathbf{J}^T\mathbf{W}\mathbf{r}$$

Once the update is computed, the pose is updated as

$$\mathbf{T}_{kf} \leftarrow \boldsymbol{\tau} \oplus \mathbf{T}_{kf}.$$

After updating the pose, the canonical pointmap is updated to improve accuracy

New points merged via confidence-weighted average

$$\tilde{\mathbf{X}}_k^k \leftarrow \frac{\tilde{\mathbf{C}}_k^k \tilde{\mathbf{X}}_k^k + \mathbf{C}_f^k \left(\mathbf{T}_{kf}\mathbf{X}_f^k\right)}{\tilde{\mathbf{C}}_k^k + \mathbf{C}_f^k}, \quad \tilde{\mathbf{C}}_k^k \leftarrow \tilde{\mathbf{C}}_k^k + \mathbf{C}_f^k.$$

Methodology:Graph Construction and Loop Closure

When should a new keyframe be added?

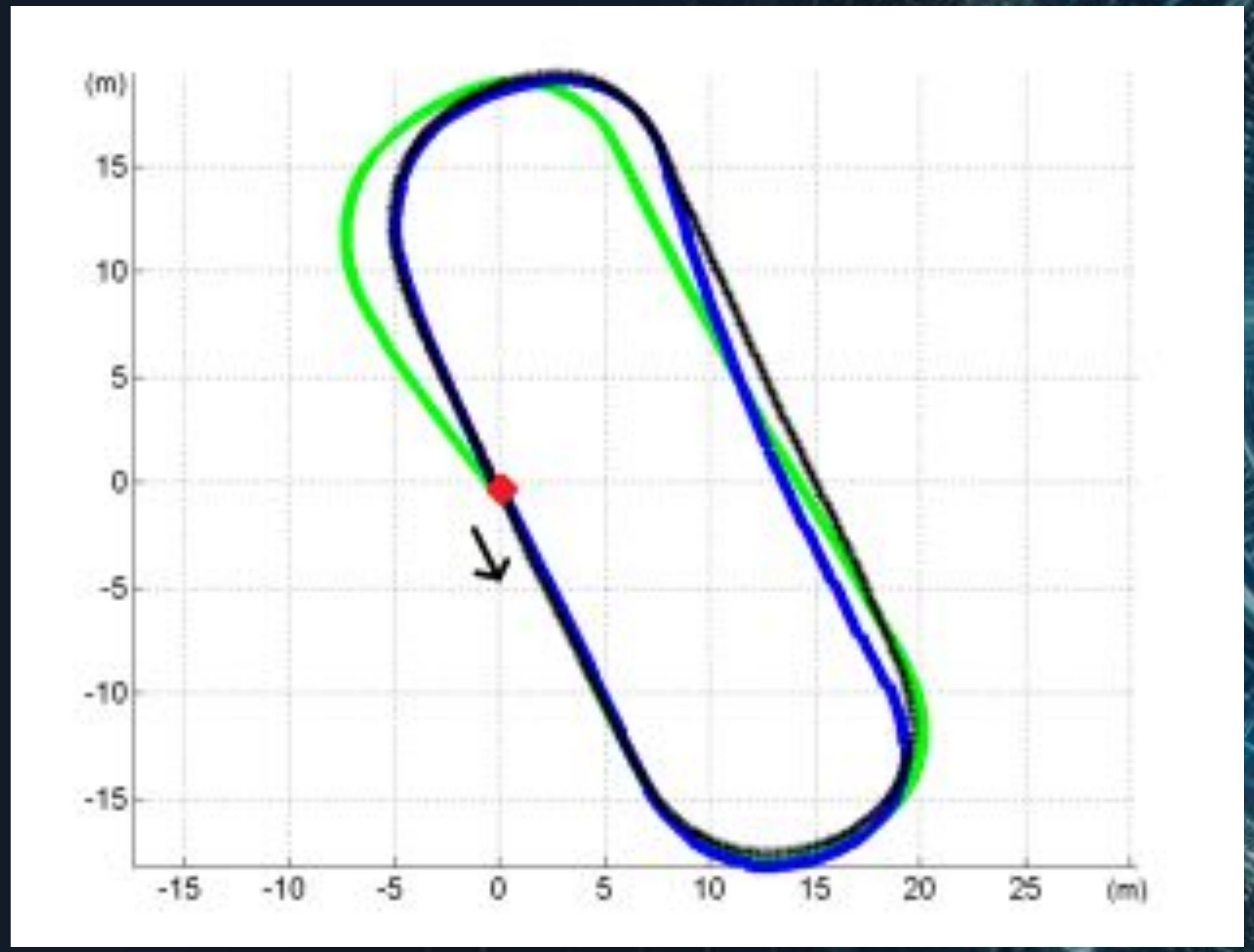
$\mathcal{K}_{i-1}$ → Corresponding(such as feature matches) ← $\mathcal{K}_i$

reference frame

ω_k

New frame

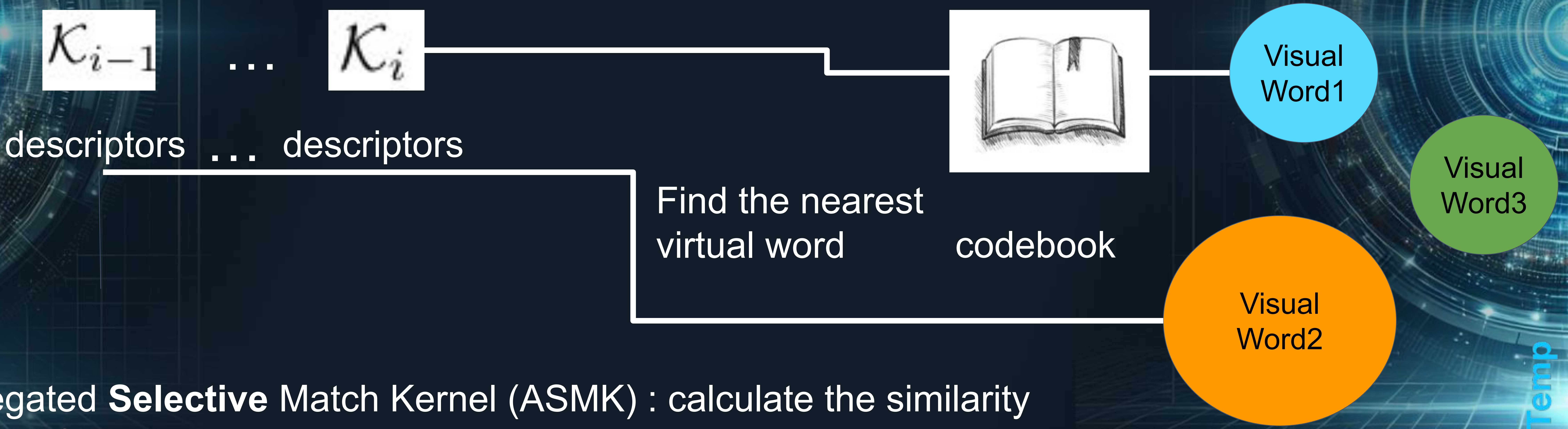
It’s time to add a new keyframe!



Drifting

Methodology:Graph Construction and Loop Closure

A loop closure is used to decrease drifting. But how to know when the loop is closure?



Aggregated **Selective** Match Kernel (ASMK) : calculate the similarity score by focusing on local descriptors that share a common visual word and have sufficiently similar orientation

Methodology:Graph Construction and Loop Closure

A loop closure is used to decrease drifting. But how to know when the loop is closure?

$\mathcal{K}$	$\mathcal{K}$	...	$\mathcal{K}$

Query

old $\longrightarrow$ latest

ASMK scores:
If the score of a pair is higher than the threshold

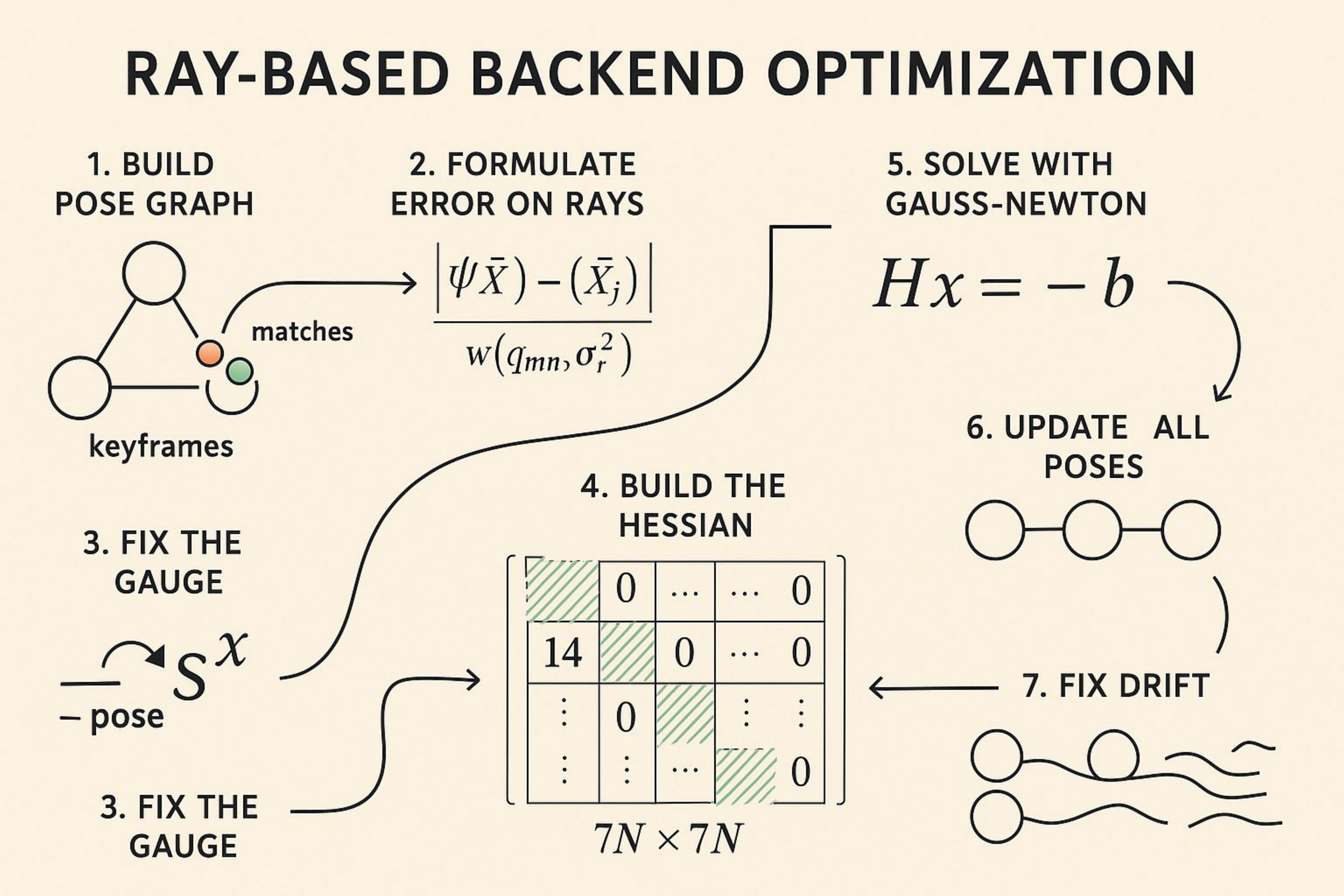
ω_r

MASt3R

ω_k

Loop Closed

Backend Optimisation



Minimize global consistency across all poses and geometry

Used Gauss-Newton optimization(a second-order method)

Why

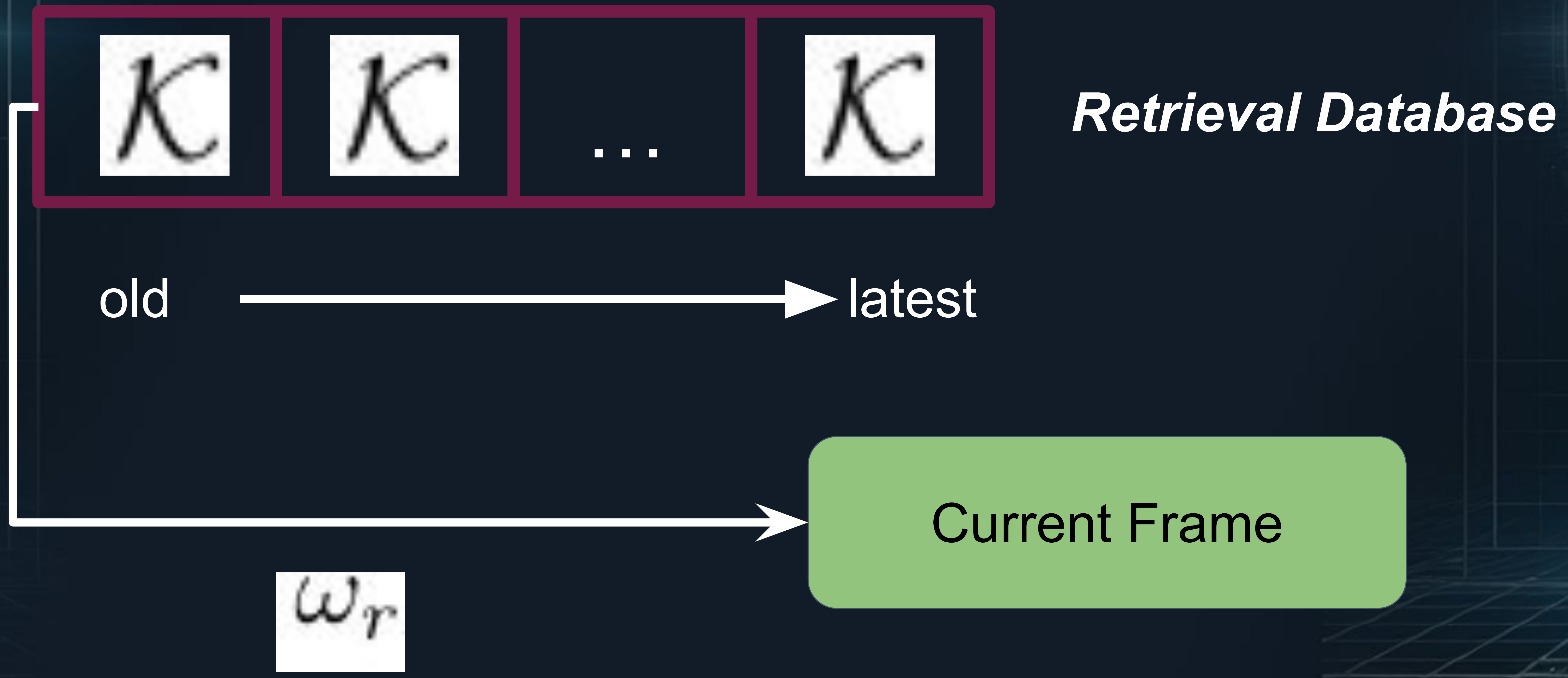
First-order requires rescaling after every iteration

Gauge Freedom - First 7 DOF

3 for Translation
3 for Rotation
1 for scaling

Methodology:Relocalization

When the system loses tracking due to an insufficient number of matches, relocalisation mode is triggered.



Methodology:Known Calibration

1.Constraining the Pointmap with Calibration: Calculate ray with known intrinsic parameters

2.Pixel-Space Residuals Instead of Ray-Space

$$E_{\Pi} = \sum_{i,j \in \mathcal{E}} \sum_{m,n \in \mathbf{m}_{i,j}} \left\| \frac{\mathbf{p}^{i}_{i,m} - \Pi\left(\mathbf{T}_{ij} \tilde{\mathbf{X}}^{j}_{j,n}\right)}{w(\mathbf{q}_{m,n}, \sigma^2_{\Pi})} \right\|_{\rho},$$

Instead of calculating in ray space, the residual can be calculated in pixel space

Improvement

- Instead of using confidence, we can use other advance methodology to pointmap fusion such as Kalman Filter)
- Train MASt3R with more datasets

Thank you for watching

Credit | Image: Unsplash

EaTemp