NOV21 – SLAM-PPT

- RULES:
  - https://kulbir-singh-ahluwalia.com/cs498gc/fa25/presentations.html

- *DEAR STUDENTS, FEEL FREE TO ADD MORE SLIDES 1 WEEK BEFORE TA FEEDBACK IN OFFICE HOURS!*

CS498GC - Fall 2025

Home Logistics Syllabus Policies Assignments Presentations Setup Gallery

Presentation Schedule Summary

November 21 (Friday) - Before Thanksgiving

3 Presentations:

- Aryan Chauhan - MASt3R-SLAM
- Wenzhou Ding - Splat, Track & Map
- Karteek Gandiboyina - cuVSLAM

Submission deadline: Nov 14

December 3 (Wednesday)

4 Presentations:

- Het Patel - Open-Vocabulary Online Semantic Mapping
- Ruben Hernandez - VSS-SLAM
- Yangkun Liu - SCE-LIO
- Nadeem Mohammed - FMCW-LIO

Submission deadline: Nov 26

December 5 (Friday)

4 Presentations:

- Keisuke Ogawa - WildGS-SLAM
- Tianyu Ren - DRAWER
- William Schafer - GS-SLAM
- Yingxue Wang - RTG-SLAM

Submission deadline: Nov 28

December 10 (Wednesday) - Last Day of Instruction

4 Presentations:

- Tongmiao Xu - Wheat3DGS
- Hariprasad Yuvaraj - Adaptive Mobile Manipulation
- Xiayu Zhao - DynaVINS++
- Krishna Teja Kolla - Multimodal Spatial Language Maps

Submission deadline: Dec 3

Quick Links & Resources

- **Presentations Table:** papers, dates, deadlines (above)
- **Syllabus & weekly schedule** (presentation weeks noted)
- **Policies** (integrity, AI-assistant use, late policy)
- **Logistics** (meeting times, platforms, grade distribution)
- **TA Office Hours:** Wed 1:30-2:30 PM @ SC 4407
- **Questions:** Use Campuswire or email ksa5@illinois.edu

If you have conflicts or accessibility needs, please reach out early via Campuswire or email.

All papers and presentation dates assigned! Start reading your assigned paper early with focus on methodology, loss functions, and novel contributions. Remember to attend TA office hours 1 week before your presentation for slide feedback.

NOV21 – SLAM-PPT

- We have 2 students presenting today:

#	Student Name	Paper Title	Paper Link	Submission Deadline	Presentation Date
1	Aryan Chauhan	MASt3R-SLAM: Real-Time Dense SLAM with 3D Reconstruction Priors	CVPR 2025	Nov 14	Nov 21 (Fri)
2	Karteek Gandiboyina	cuVSLAM: CUDA Accelerated Visual Odometry and Mapping	NVIDIA arXiv Preprint	Nov 14	Nov 21 (Fri)

What We Want You to Learn (and Show)

- **Methodology & architecture** — explain the system diagram and motivate each design choice
- **Novelty** — what's truly new and why it matters
- **Limitations** — be specific and evidence-based (failure cases, assumptions, compute/data constraints)
- **Improvements** — concrete ideas to address those limitations

> Prof. Girish's suggested questions for a compelling talk:
>
> 1. Why is it important?
> 2. Why is it hard?
> 3. How is it novel?
> 4. Why does it work?

Suggested Slide Outline (Target: 15 minutes)

- **Problem & motivation** (tie explicitly to mobile robotics/SLAM)
- **Prior art** (2–3 key related works; cite fairly but highlight shortcomings)
- **Key contribution & approach** (detailed system/architecture diagram with time estimates)
- **Technical depth** (walk us through the math/algorithms; highlight novel formulations)
- **Results** (datasets, metrics, comparison tables, visualizations; be critical)
- **Demo/video** (if available—real-world robot footage is gold!)
- **Limitations** (specific failure cases from the paper or your analysis)
- **Future work & your ideas** (thoughtful proposals for addressing those limitations)
- **Q&A prep** (anticipate technical questions on losses, hyperparameters, training)

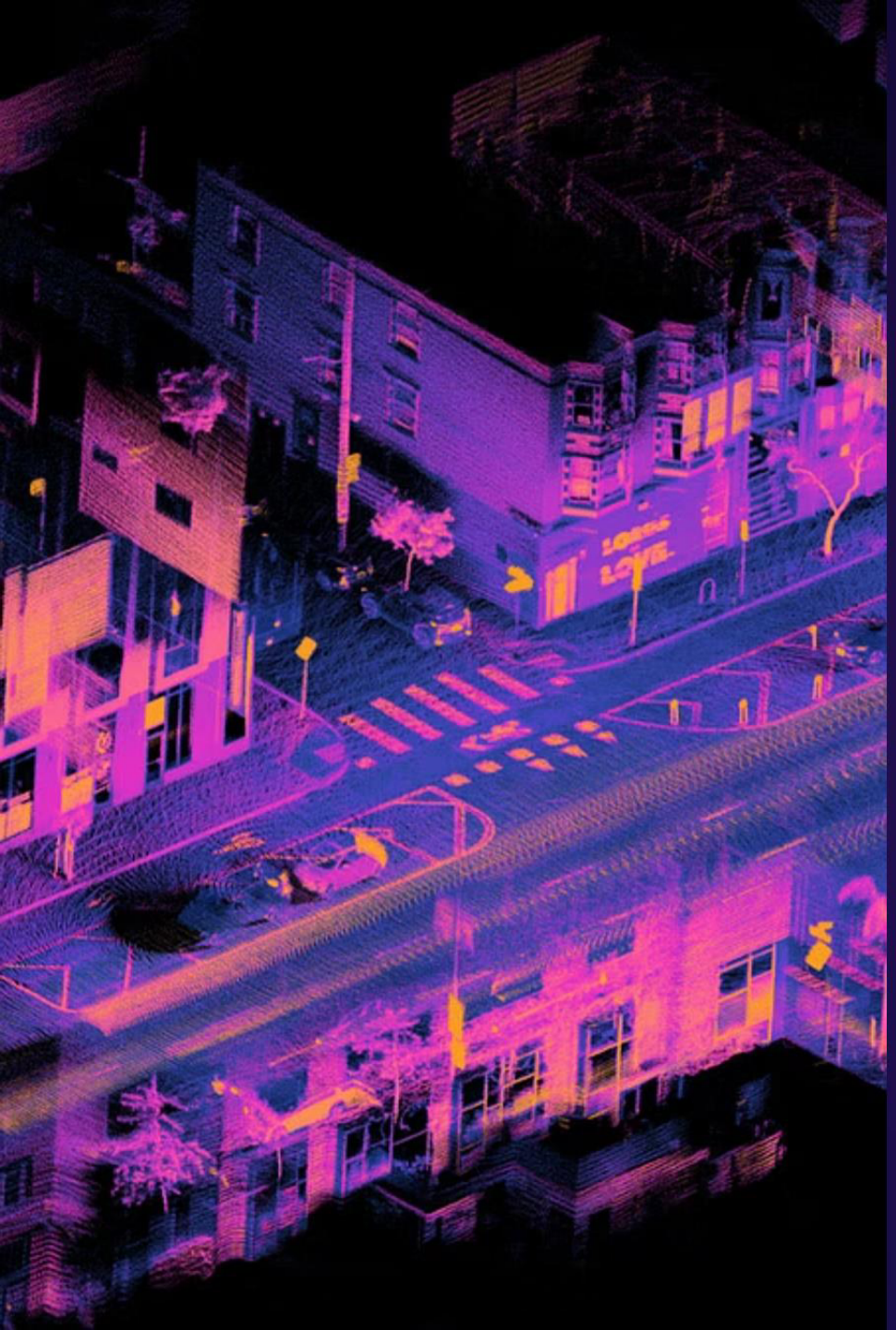
Target: ~15 slides for 15 minutes

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

Presented by: Aryan Chauhan

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

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



MASt3R-SLAM: Dense Monocular SLAM at ~15 FPS using Two-View 3D Priors

This presentation introduces MASt3R-SLAM, a novel real-time monocular SLAM system that achieves globally consistent poses and dense geometry reconstruction.

The first real-time SLAM built on the MASt3R two-view 3D reconstruction prior.

- Operates effectively without a parametric camera model, assuming only a single camera center.
- With calibration, achieves state-of-the-art trajectory and geometry performance on standard benchmarks.

Helpful Definitions

MASt3R

MASt3R stands for **Matching and Structure from 3D Reconstruction**; it is a deep neural network that predicts dense 3D pointmaps and matching features from pairs of images

Parametric Model

The camera is assumed to have certain *parameters* defined such as focal point aperture shutter

Priors

In this context, priors refer to pre-existing knowledge or assumptions used to inform the SLAM process, such as two-view 3D reconstruction.

Point Map Matching

The process of aligning and matching 3D points from different views to create a consistent map.

Camera Tracking

The technique of following the position and orientation of a camera as it moves through space.

Local Fusion

The integration of data from multiple sources or frames to create a coherent local representation.

Graph Construction

Building a graph structure to represent relationships between different data points or frames in SLAM.

Loop Closure

The process of recognizing previously visited locations to correct drift and improve map accuracy.

Global Optimization

A method to refine the entire map or model to ensure consistency and accuracy across all data.

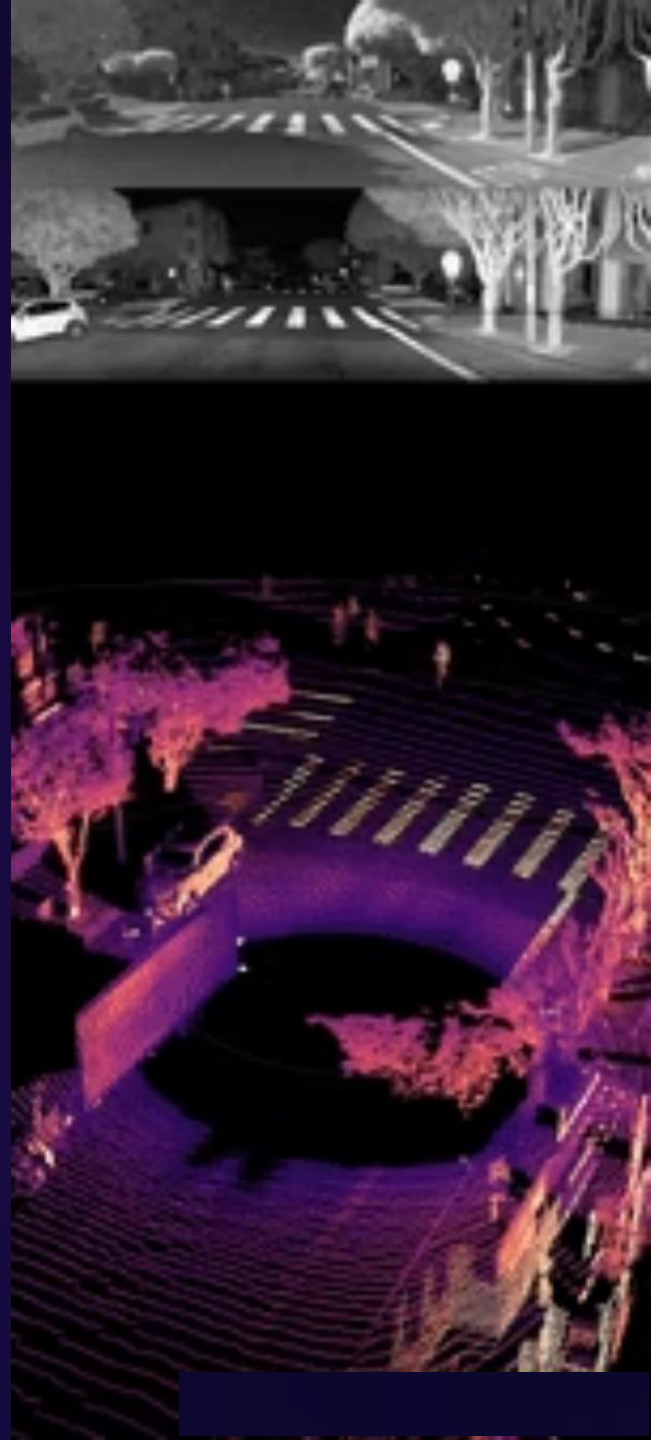
Monocular

Referring to the use of a single camera for capturing and processing visual information.

Why This Matters: Dense SLAM That "Just Works" from a Single Camera

Traditional SLAM systems often require specialized multi-sensor setups, leading to complex and expensive deployments. Many struggle with drift over time or provide only sparse environmental reconstructions, limiting their utility in real-world applications.

MASt3R-SLAM addresses these critical challenges by delivering real-time, dense 3D maps and globally consistent camera poses using just a single monocular camera. This breakthrough simplifies hardware requirements, reduces costs, and opens doors for SLAM to be adopted in a much wider array of use cases, from robotics to augmented reality, where reliability and ease of integration are paramount.

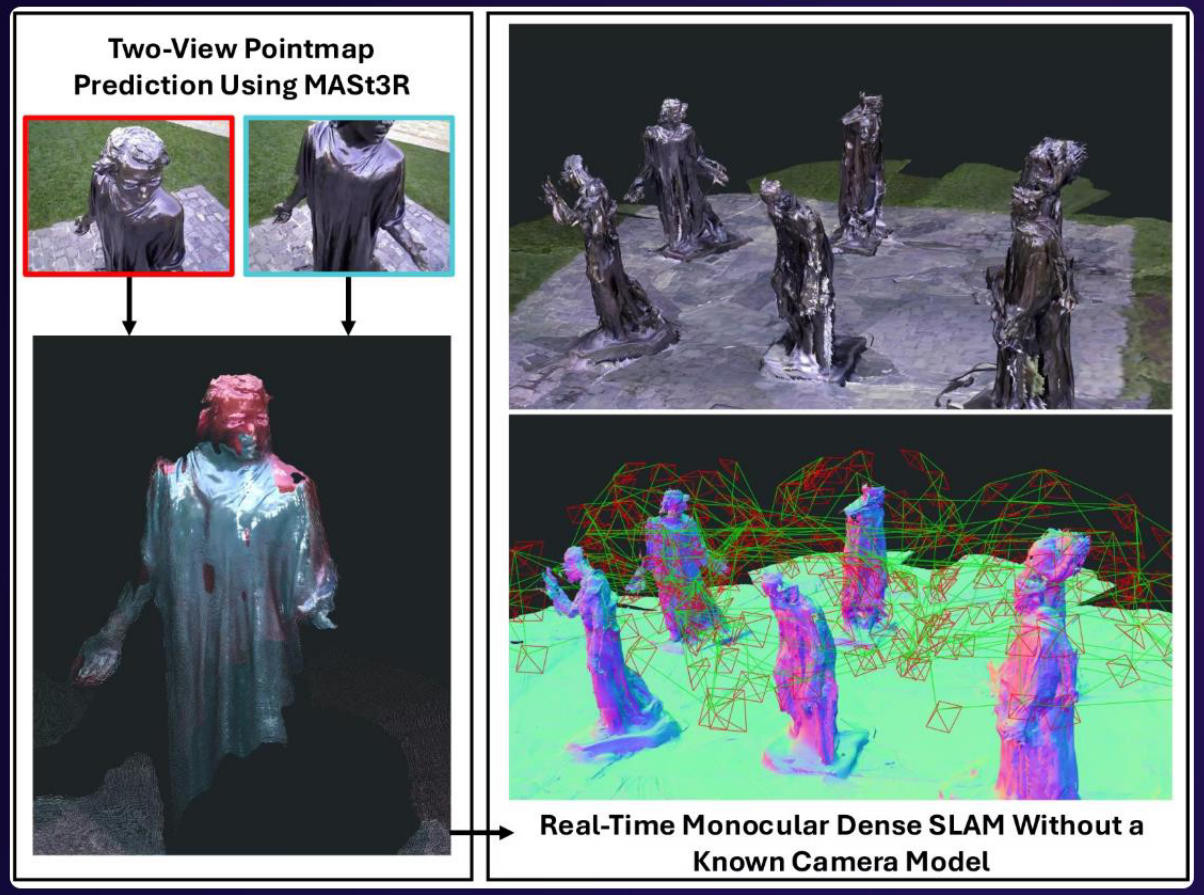


Why This Matters: Dense SLAM That "Just Works" from a Single Camera

Achieving robust dense monocular SLAM in diverse, unconstrained environments remains a significant challenge, often plagued by system brittleness and inconsistencies.

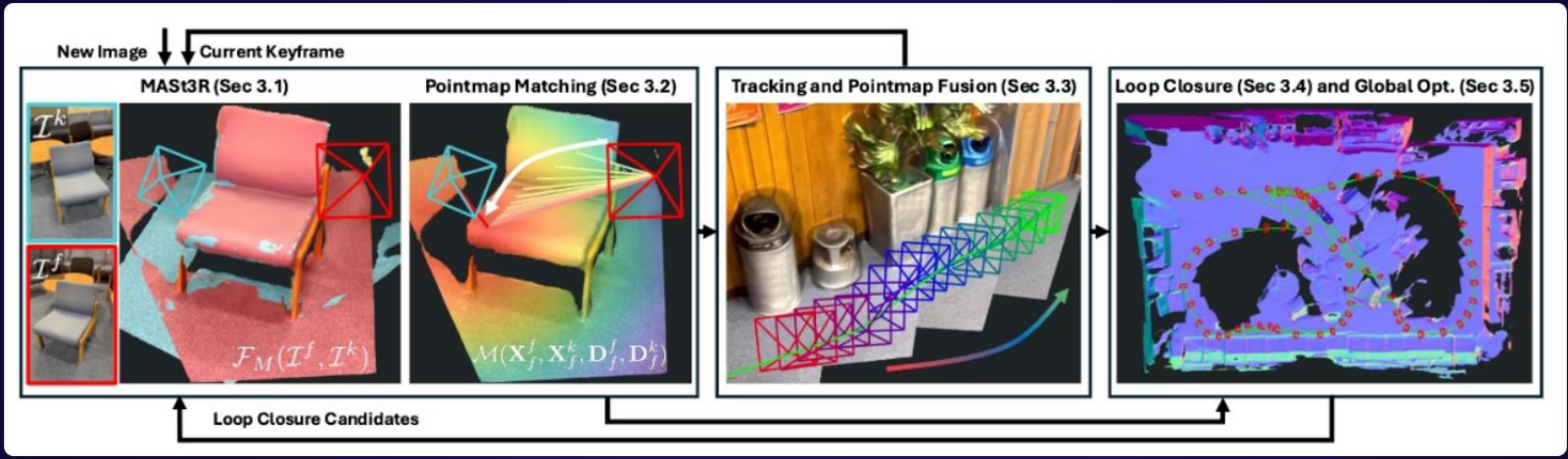
- In-the-wild monocular dense SLAM is often fragile due to calibration requirements, drift, and inconsistency issues.
- Traditional single-view priors introduce ambiguity, while multi-view optical flow tightly couples pose and geometry estimation.
- Two-view **3D** priors, like those provided by DUSt3R/MASt3R, revolutionize Structure-from-Motion (SfM) and SLAM by predicting pointmaps in a shared frame.

The Key Idea: Unifying Prior with MASt3R's Two-View Pointmaps



- MASt3R's outputs are treated as pseudo-measurements, guiding tracking, mapping, and loop closure processes.
- Eliminates the need for fixed camera intrinsics, operating under a generic central camera assumption.
- Combines fast matching techniques with second-order global optimization to ensure overall consistency.

System Diagram: Frontend + Backend Built Around MASt3r



MASt3R-SLAM integrates prediction, matching, tracking, and global optimization into a cohesive, high-performance system.

MASt3R Prediction

Generates dense pointmaps and per-pixel features for each input frame.

Pointmap Matching

Iterative projection in ray-space, followed by local feature refinement.

Tracking & Local Fusion

Estimates camera pose relative to the last keyframe and performs robust pixel-wise fusion.

Graph + Loop Closure

Uses feature retrieval to identify loop closure candidates and add graph edges.

Global Optimization

Applies second-order (Gauss-Newton) optimization in ray/pixel space for global consistency.

Preliminaries & Notation (MASt3R)

Image Pair to Predictions

MASt3R takes an image pair and produces a rich set of predictions:

$$(T_i T_j) \xrightarrow{\mathcal{F}_M} (X_i^{j}, X_j^{i}, D_i^{j}, D_j^{i}, C_i^{j}, C_j^{i}, Q_i^{j}, Q_j^{i})$$

- **T**: RGB images ($\mathbb{R}^{H\times W\times 3}$)
- **X**: **Dense Pointmaps** ($\mathbb{R}^{H\times W\times 3}$)
- **D**: **Features** for matching ($\mathbb{R}^{H\times W\times d}$)
- **C, Q**: Per-pixel **Confidences** ($\mathbb{R}^{H\times W\times 1}$)

Indexing Convention: (X_i^{j}) denotes 3D points of image *(i)*, expressed in camera *(j)'s* coordinate frame. Think of the subscript as "which image" and the superscript as "which coordinate frame."

Pose Representation (Sim(3))

Pose is represented in **Sim(3)** to explicitly optimize for scale, as training data may contain varying scales.

$$T \in \mathrm{Sim}(3), \qquad T = \begin{bmatrix} sR & t \\ 0 & 1 \end{bmatrix}, \; R \in \mathrm{SO}(3), \; t \in \mathbb{R}^3, \; s \in \mathbb{R}^+$$

Pose Update

Pose updates are performed by multiplying a small motion on the Lie group:

$$T \leftarrow \tau \oplus T \stackrel{\text{def}}{=} \exp(\tau), T, \quad \tau \in \mathfrak{sim}(3)$$

Camera Model Assumption

MASt3R-SLAM operates under a **generic central camera** assumption, meaning all rays pass through a single camera center. The unit-ray map for a point (X^i) in camera *i* is defined as:

$$\psi\left(X^i\right) = \frac{X^i}{|X^i|} \in \mathbb{S}^2$$

Efficient Matching: Iterative Ray-Space Projection + Feature Patch Search

Core Idea: Local Optimization in Ray-Space

The novel approach optimizes in ray-space, matching rays from a generic central camera. This focuses on angles, not depths, ensuring robustness to depth noise and eliminating camera intrinsics.

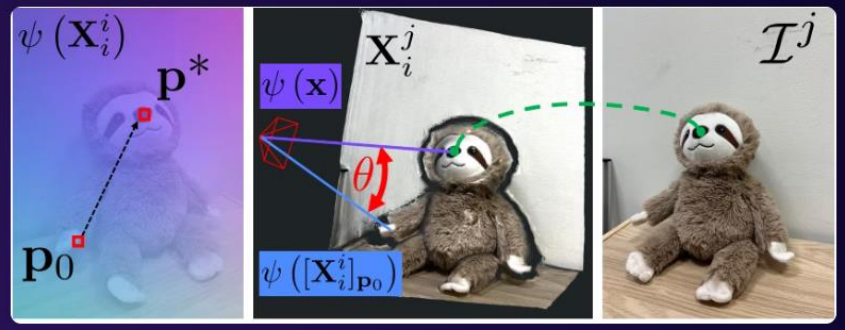
Projection minimizes ray error for each point $\mathbf{x} \in X_j^i$ using:

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

The angular distance between two rays ψ_1 and ψ_2 is:

$$|\psi_1 - \psi_2|^2 = 2(1 - \cos\theta), \quad \cos\theta = \psi_1^\top \psi_2 \tag{3}$$

This non-linear least-squares problem is solved efficiently with the **Levenberg-Marquardt** algorithm, converging rapidly due to the ray image's smoothness in about 10 iterations.



Refinement, Robustness & Performance

- **Initialization:** Strong initialization from previous frames (tracking) or identity mapping (new keyframes).
- **Outlier Handling:** Matches are invalidated if they exhibit large 3D separation, effectively managing occlusions and noise.
- **Feature Refinement:** Initial geometric matches are improved via a local patch search around projected points, maximizing similarity between descriptor patches.
- **Parallelization:** Custom **CUDA kernels** achieve high performance, ensuring **~2ms** for tracking and quick graph edge formation.

Robust Tracking & Local Fusion for Frontend Stability

Tracking: Estimating Relative Pose

This module estimates the relative pose between the current frame (I^f) and the last keyframe (I^k), with initial pointmaps and features generated by MASt3R.

We use two residual types for pose estimation, with the Ray-space Residual being preferred due to its robustness to point-depth errors and effectiveness with large baselines:

Naïve 3D Point Residual (depth-sensitive):

$$E_p = \sum_{m,n \in \mathcal{M}_{f,k}} \left\| \hat{X}^k_{k,n} - T_{k \leftarrow f} X^f_{f,m} \right\|^2_\rho / w(q_{m,n}, \sigma^2_\rho)$$

Ray-space Residual (preferred):

$$E_r = \sum_{m,n \in \mathcal{M}_{f,k}} \left\| \psi\left(\hat{X}^{,k}_{k,n}\right) - \psi\left(T_{k \leftarrow f} X^{,f}_{f,m}\right) \right\|^2 / w(q_{m,n}, \sigma^2)$$

The pose is updated by solving a non-linear least squares problem using Iteratively Reweighted Least Squares (IRLS) and Gauss–Newton optimization in $\mathrm{Sim}(3)$, with a small distance term to resolve pure-rotation ambiguities.

$$(J^\top W J)\tau = -J^\top W r, \qquad T_{k \leftarrow f} \leftarrow \exp(\tau) T_{k \leftarrow f}$$

Pointmap Fusion: Updating the Keyframe Map

After robust pose estimation, new, confident geometry from the current frame is integrated into the canonical keyframe map ($\hat{X}^{,k}$).

This fusion uses a confidence-weighted running average:

$$\hat{X}^{,k}_i \leftarrow \frac{\hat{C}^{,k}_i \hat{X}^{,k}_i + C^{,f}_i \left(T_{k \leftarrow f} X^{,f}_i\right)}{\hat{C}^{,k}_i + C^{,f}_i}, \qquad \hat{C}^{,k}_i \leftarrow \hat{C}^{,k}_i + C^{,f}_i$$

This process rejects occlusions and large 3D inconsistencies. Direct fusion with the keyframe (rather than just the last frame) improves accuracy over larger baselines and prevents "smearing" from inconsistent depth estimates.

All tracking and fusion steps are highly parallelized on GPU, with tracking typically completing in a few milliseconds. The process yields a refined relative pose ($T_{k \leftarrow f}$) and a cleaner, more trustworthy keyframe pointmap ($\hat{X}^{,k}$) for subsequent backend optimization and loop closure.

> The ray-space residual ensures stability even when MASt3R's depth predictions have minor inaccuracies.

Graph, Loop Closure, and Relocalization: Eliminating Drift

The system effectively counters drift and recovers from tracking loss using MASt3R features for robust loop closure and relocalization.

01

Keyframe Insertion

New keyframes are added when match coverage falls below a predefined threshold, ensuring sufficient overlap.

02

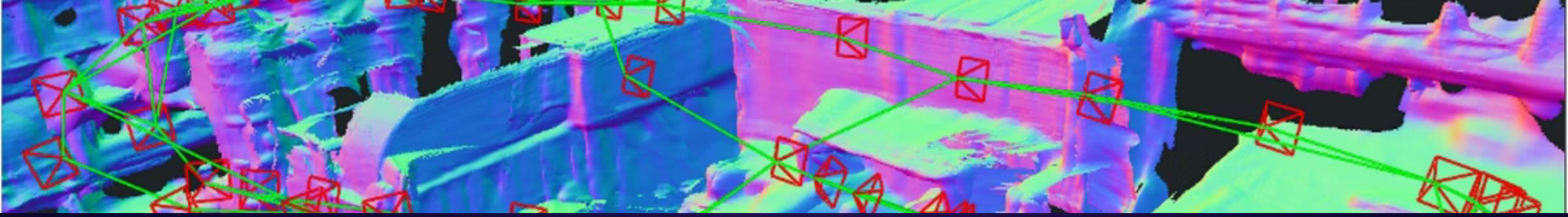
Feature Retrieval & Decoding

Image retrieval is performed on encoded features, which are then decoded with MASt3R. Edges are added if matches pass the ω_L threshold.

03

Relocalization

Achieved through stricter retrieval and match checks, allowing the system to recover its pose after tracking loss.



Backend Optimization: Details on Global Consistency

The backend ensures global consistency by optimizing all keyframe poses ($\mathbf{T}_{WC_i} \in \mathrm{Sim}(3)$) and canonical keyframe pointmaps ($\tilde{\mathbf{X}}^{,i}$) such that all pairwise matches agree in ray-space.

The total energy minimized across all graph edges ($\mathcal{E}$) is given by:

$$E_g = \sum_{i,j \in \mathcal{E}} \sum_{m,n \in \mathcal{M}_{i,j}} \left\| \psi\left(\tilde{\mathbf{X}}_m^{,i}\right) - \psi\left(\mathbf{T}_{ij}, \tilde{\mathbf{X}}_n^{,j}\right) \right\|_\rho^2 / w(q_{m,n}, \sigma_r^2)$$

where $\mathbf{T}_{ij} = \mathbf{T}_{WC_i}^{-1}, \mathbf{T}_{WC_j}$ is the relative pose, $\psi(\cdot)$ maps to unit rays, $\rho(\cdot)$ is the robust Huber function, and $w(\cdot)$ uses match confidence $q_{m,n}$.

Solver Architecture

A **second-order Gauss–Newton** method is employed with **sparse Cholesky decomposition** for efficient optimization.

System Size & Efficiency

For **N** keyframes, we optimize a **7N × 7N** system. Each graph edge contributes **14 × 14 Hessian blocks**. Efficiency is boosted by **analytic Jacobians** and custom **CUDA reductions**.

Stability & Practicality

A tiny **distance-consistency** term is added to prevent degeneracy in pure rotation cases. The solver converges in **≤10 iterations** per new keyframe and is **not the runtime bottleneck**.

To remove global ambiguity, the first 7-DoF pose is anchored, effectively fixing the global scale, rotation, and translation.

Add results, baseline, ablation study

- What conclusion do you plan to draw from the results?
- Max 4-5 lines per slide

Add more demos and videos, visual graphs/tables + 2-3 key observations

- Point out limitations
- Suggest improvements
- Engage the audience

3D Prior + Ray-Space Modeling + Second-Order Backend

MASt3R-SLAM's effectiveness stems from a synergistic combination of advanced 3D priors, flexible ray-space processing, and robust global optimization.

Two-View 3D Priors

Reduces ambiguity inherent in traditional single-view depth or normal priors, providing a strong initial structure.

Ray-Space Residuals

Enables operation with unknown camera intrinsics, optimizing in ray-space; switches to pixel-space when calibrated for higher precision.

Loop Closure & 2nd-Order Optimization

Achieves global consistency, effectively eliminating long-term drift through a robust and efficient backend.

Current Limitations & Bottlenecks

While MASt3R-SLAM demonstrates robust performance, understanding its inherent limitations is crucial for continued development and application.

- **No full global geometry refinement:** The system's geometry filters are applied only at the frontend, meaning no full global refinement of the entire 3D map is performed post-optimization.
- **Pinhole camera dependency:** MASt3R is currently trained primarily on pinhole camera models, leading to performance degradation when confronted with severe lens distortion.
- **Full-resolution decoder bottleneck:** The full-resolution decoder, while providing high detail, introduces a latency bottleneck during critical tracking and loop closure verification steps.

This example illustrates tracking instability in environments with significant lens distortion, highlighting a current vulnerability.

Future Work: Future iterations will focus on training MASt3R with diverse camera models, including those with radial and tangential distortions, to enhance real-world applicability.

(As referenced in the **arXiv paper**)

Important, Hard, Novel, Works

MASt3R-SLAM's impact and unique approach can be summarized by these four key takeaways:

Why it's Important

Enables real-time dense SLAM from a single monocular camera, making advanced 3D understanding accessible with minimal hardware assumptions.

Why it's Hard

Overcomes inherent ambiguities in monocular systems (pose, geometry, intrinsics), manages drift effectively, and maintains performance under strict runtime constraints.

How it's Novel

Introduces a two-view 3D prior as the unifying signal for the entire SLAM pipeline, going beyond traditional Structure-from-Motion applications.

Why it Works

Leverages fast ray-space matching, robust loop closure, and a powerful second-order optimization backend to deliver globally consistent and accurate results.

For more details, refer to the arXiv paper.

Future work

- Kindly add future work to address limitations
- Suggest improvements

cuVSLAM: CUDA Accelerated Visual Odometry and Mapping

Presented by: Karteek Gandiboyina

2025-7-9

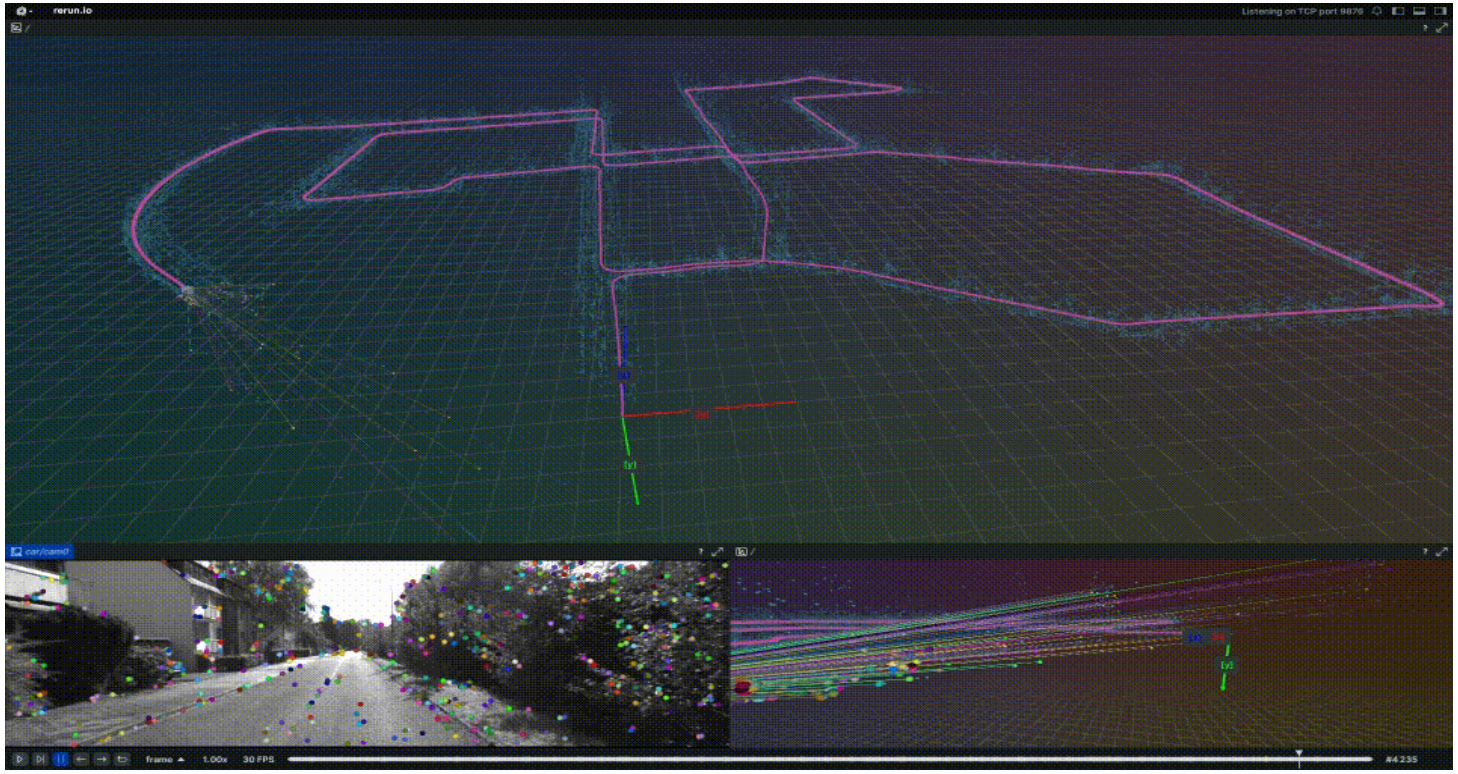
cuVSLAM: CUDA accelerated visual odometry and mapping

Alexander Korovko[1], **Dmitry Slepichev**[1], **Alexander Efitorov**[1], **Aigul Dzhumamuratova**[1], **Viktor Kuznetsov**[1], **Joydeep Biswas**[1], **Hesam Rabeti**[1] **and Soha Pouya**[1]

[1]NVIDIA , {akorovko, dslepichev, aefitorov, adzhumamurat, vkuznetsov, jbiswas, hrabeti, spouya}@nvidia.com

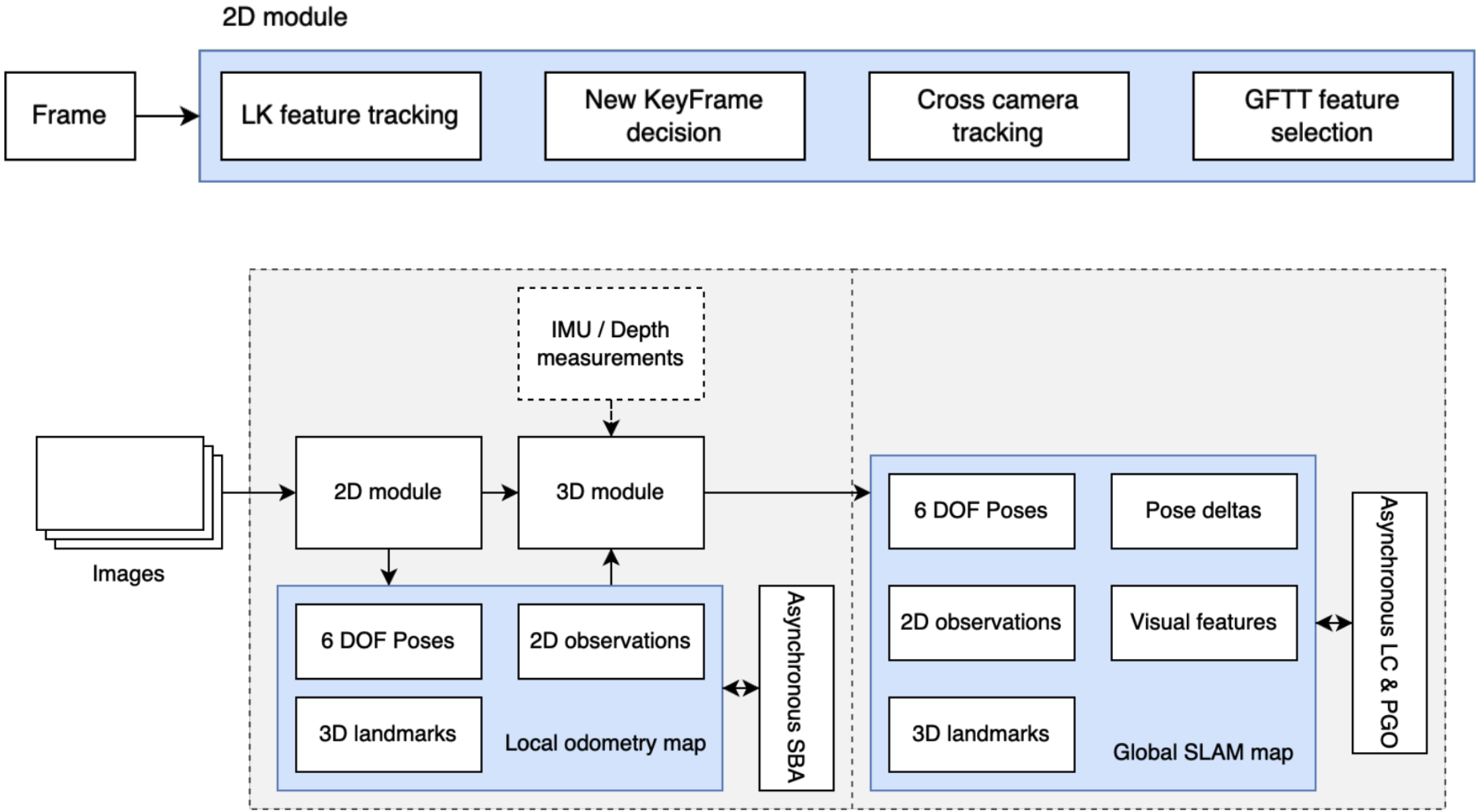
PROBLEM & MOTIVATION

- Real-Time V-SLAM on Edge Devices
- Lack of Unified Multi-Sensor Support



Definitions / Background

- Visual Odometry (VO): Estimates the robot's pose sequentially, using only local visual information (e.g., between two consecutive frames). Suffers from accumulating drift over long trajectories.
- SLAM: Combines VO with Loop Closing (LC) and Global Optimization to eliminate accumulated drift and build a globally consistent map.
- cuVSLAM Architecture: Uses a fast, smooth Frontend (VO) for immediate pose updates and a robust, asynchronous Backend (SLAM) for map consistency.


2D module
Frame
LK feature tracking
New KeyFrame decision
Cross camera tracking
GFTT feature selection
IMU / Depth measurements
Images
2D module
3D module
6 DOF Poses
2D observations
3D landmarks
Local odometry map
Asynchronous SBA
6 DOF Poses
Pose deltas
2D observations
Visual features
3D landmarks
Global SLAM map
Asynchronous LC & PGO

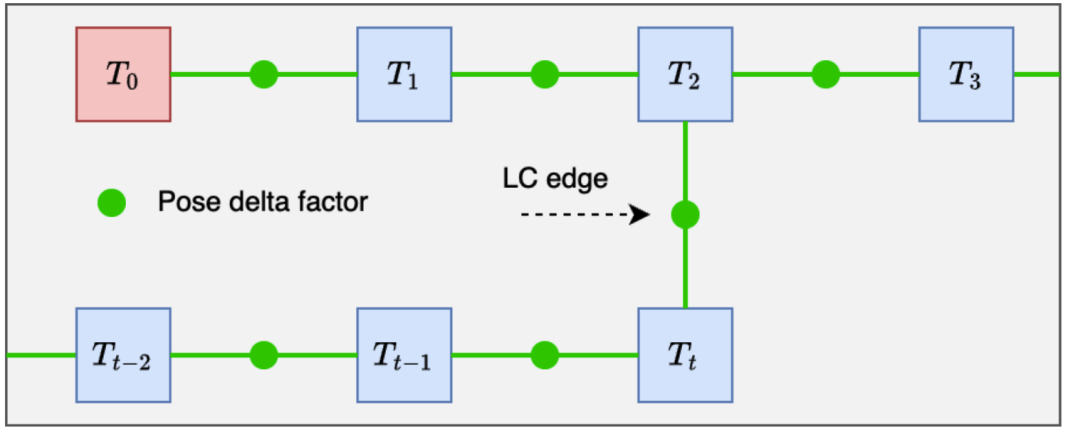
Method details

2D Module	3D Module
Real-time extraction and tracking of features on the 2D image plane	Estimate the camera's 6-Degrees-of-Freedom (6 DOF) pose (position and orientation) and build the local 3D map.
• Feature Selection: Identifies and extracts high-contrast image keypoints for tracking • 2D Tracking: Tracks the movement of these features across consecutive image frames, often using a method like the Lucas-Kanade algorithm.	• Landmark Triangulation: Uses the 2D observations from one or more camera views (from the 2D module) to calculate the 3D coordinates of the features • 2. Pose Estimation (Localization): Determines the camera's 6 DOF pose (translation and rotation) using the newly tracked 2D features and their corresponding 3D landmarks in the map • Saves the new 3D landmarks and camera poses to the local odometry map. • Local Refinement: Initiates a local Sparse Bundle Adjustment (SBA) to optimally refine the recently calculated poses and landmark positions for improved local accuracy.
Output: 2D Observations: A set of tracked feature points and their coordinates on the image plane	Output • 6 DOF Poses: The accurate position and orientation of the camera/robot in the world. • 3D Landmarks: A consistent set of 3D points representing the environment.

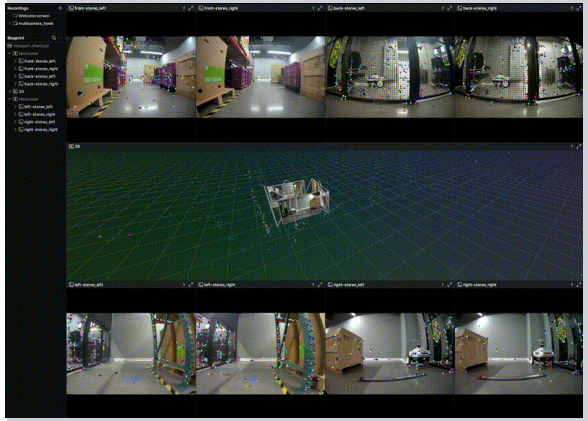
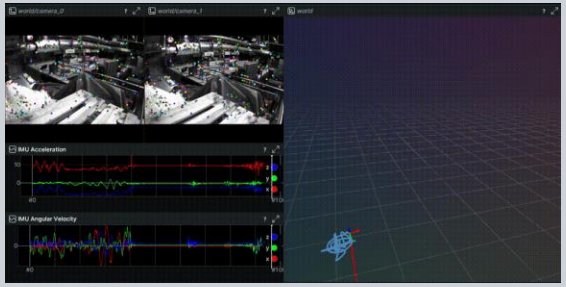
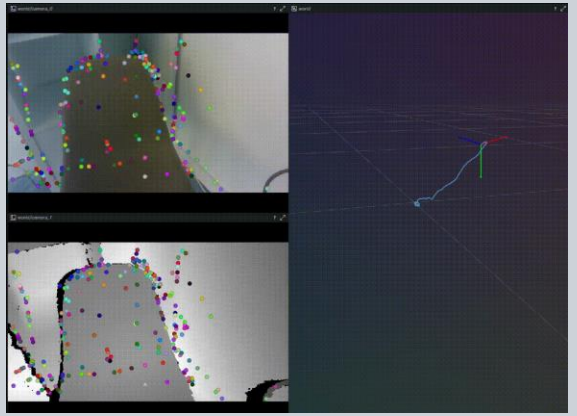
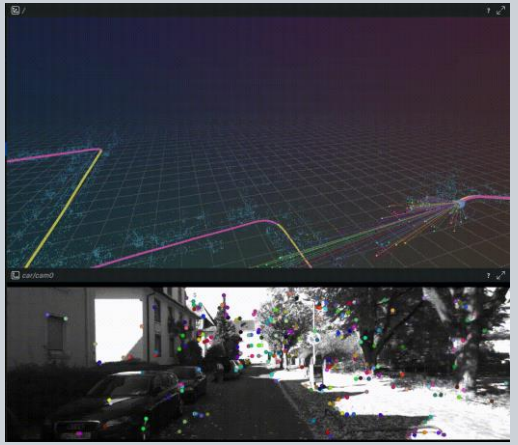
Method details

Sterio	Multi-Sterio	Visual-Inertial	Mono-Depth	Mono
• Cross-Camera Triangulation: Performs left-to-right feature tracking to obtain stereo observations and calculate 3D landmarks on keyframes. • Pose Estimation: Uses the PnP algorithm on subsequent frames.	• Enhanced Robustness: Treats the configuration as multiple stereo pairs to maintain robustness in feature-poor areas. • Setup: Builds a Frustum Intersection Graph (FIG) to manage cross-camera tracking only between cameras with overlapping Fields of View (FoV). Requires synchronized images.	• Factor Graph Optimization: Defines the robot's state as a 15-DOF vector (Pose, Velocity, Biases). • Key Factors: Constrains consecutive states using IMU preintegration factors, visual factors, and a prior factor. Includes a crucial Gravity Estimation step.	Dense Frame-to-Frame Estimation: Estimates pose using a combination of multiple factors in an optimization problem : 1. Visual Factor (reprojection error). 2. Dense Intensity & Depth Factors (per-pixel constraints). 3. Point-to-Point Factor (using 2D matches).	• Relies on motion baseline to triangulate points from different camera poses. • Initialization: Uses the Fundamental Matrix and RANSAC for the first frames to estimate the initial pose up-to-scale.

Loop closing and global map refinement



- Loop Closure Detection: Identifies when the robot revisits a past location to detect and measure accumulated drift (odometry error) across the long trajectory.
- Pose Graph Optimization (PGO): Adds a constraint for the detected loop, then globally optimizes the entire history of camera poses to distribute the error and correct the map's overall shape.
- Global Bundle Adjustment (BA): Final step that simultaneously refines both the globally corrected poses and 3D landmark positions for maximum final map accuracy and consistency.

Stereo/Multi-Sterio	Visual-Inertial	Mono-Depth	Mono
$$E_{Stereo} = \sum_{k} \sum_{i,j} \|\mathbf{p}_k^j - \pi(\mathbf{T}_{w,j}^{-1}\mathbf{X}_k)\|$$	$$E_{VI} = E_{Prior} + \sum_{k} E_{IMU} + \sum_{j,i} E_{Reproj}$$	$$E_{RGBD} = E_{Reproj} + \lambda_D E_{Depth} + \lambda_I E_{Int}$$	$$E_{Mono} = \sum_{i} \sum_{j} \|\mathbf{p}_i^j - \pi(\mathbf{T}_{w,j}^{-1}\mathbf{X}_i)\|$$
			

Results

Configuration	Track call time, ms		Jetson HW utilization	
	Desktop	Jetson	CPU %	GPU %
Mono	0.9	2.7	NA	NA
Mono-Depth	5.9	15.1	2.6	55.0
Stereo-Inertial	1.3	3.8	1.3	2.2
Stereo	0.4	1.8	5.5	1.7
Multicamera (2-stereo)	0.8	2.0	8.3	9.0
Multicamera (3-stereo)	1.4	2.3	8.3	11.0
Multicamera (4-stereo)	2.1	NA	NA	NA

Results

Mode	Dataset	Method	avgRTE, %	avgRE, deg	RMSE APE
Mono-Depth	AR table	ORBSLAM3	-	-	-
		DPVO	4.42	**1.65**	0.77
		cuVSLAM Odom	0.34	3.59	0.09
		cuVSLAM SLAM	**0.19**	1.68	**0.025**
	ICL-Nuim	ORBSLAM3	-	-	-
		DPVO	10.28	**0.77**	0.61
		cuVSLAM Odom	**0.41**	0.99	**0.026**
		cuVSLAM SLAM	0.44	0.97	**0.026**
	TUM RGB-D	ORBSLAM3	**0.50**	2.98	0.067
		DPVO	13.56	**1.98**	0.80
		cuVSLAM Odom	1.35	5.52	0.11
		cuVSLAM SLAM	0.99	4.13	**0.065**
Stereo	EuroC*	ORBSLAM3	0.21	1.41	0.068
		DPVO	0.21	**0.96**	0.10
		cuVSLAM Odom	0.29	1.96	0.13
		cuVSLAM SLAM	**0.17**	1.12	**0.054**
	Kitti	ORBSLAM3	0.31	1.20	2.98
		DPVO	21.69	1.14	195.05
		cuVSLAM Odom	0.33	1.14	3.00
		cuVSLAM SLAM	**0.27**	**0.93**	**1.98**
Stereo-Inertial	EuroC	ORBSLAM3	5.70	72.23	**0.066**
		DPVO	-	-	-
		cuVSLAM Odom	0.39	2.69	0.19
		cuVSLAM SLAM	**0.29**	**2.27**	0.13
	TUM-VI Room	ORBSLAM3	2.17	**1.37**	**0.077**
		DPVO	-	-	-
		cuVSLAM Odom	0.20	3.85	0.18
		cuVSLAM SLAM	**0.12**	3.00	0.12

Limitations

- Reliance on Features: Accuracy drops significantly in environments with repetitive textures, lack of features (e.g., white walls), or motion blur.
- Initialization (Scale Ambiguity): Monocular mode requires motion to initialize and estimates pose only up-to-scale, requiring external sensing or a dedicated initialization sequence.
- Sensor Dependency: Robustness relies heavily on accurate sensor calibration (intrinsics/extrinsics) and hardware synchronization for multi-camera and VIO modes.
- Lost Tracking/Kidnapping: The system requires external localization (e.g., LiDAR or an external module) to recover the correct global pose after tracking is fully lost.

Future work

- Kindly add future work to address limitations
- Suggest improvements

Thank you!

Any Questions?