Monocular Visual-Inertial Odometry with Planar Regularities

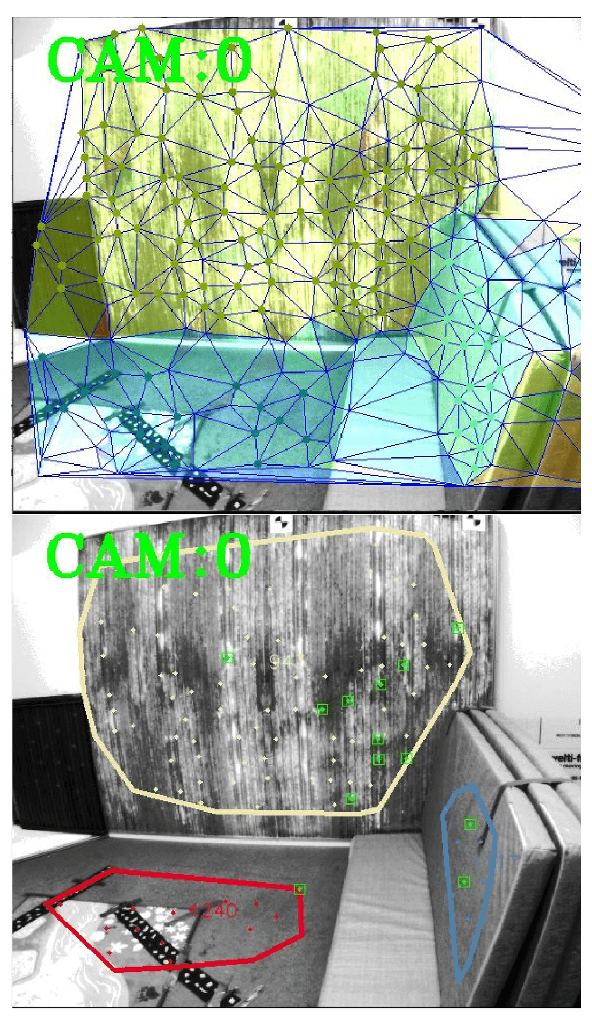
Chuchu Chen*, Patrick Geneva*, Yuxiang Peng, Woosik Lee and Guoquan Huang

Motivation & Contribution

- Man-made environments have rich **structural** information (i.e. wall, ground, table and etc.)
- Regularization constraints can represent **higher-level**, **larger-spatial** geometric information and further improve state estimation
- **Contributions:**
  - **Monocular** filter-based VIO that enforces 3D feature **structural regularities** to environmental planes
  - A novel **robust** plane detection and tracking
  - **Open sourced** code and dataset for community

Plane Feature Detection and Tracking

- **Sparse Point Features**
  - FAST detect and KLT track
  - Enables temporal tracking
- **Point Feature Processing**
  - Recover 3D feature positions
  - Delaunay triangulation
  - Compute triangle normals
- **Vertex Normals**
  - Avg. normals for each vertex
  - Reject high variances ones
- **Vertex Matching Heuristics**
  - Pairwise match comparison
  - Robustly filter outliers



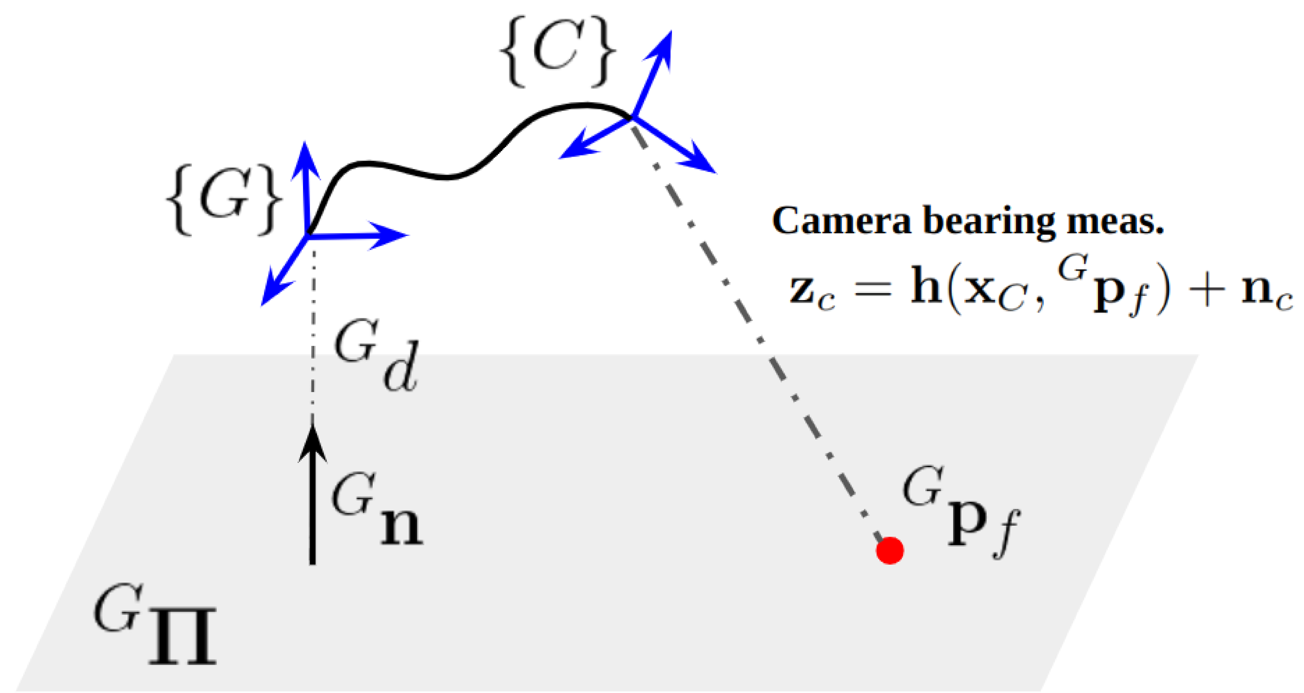
Results - Detection and Tracking

- Plane feature can be tracked longer than point features
- Feature tracking algorithm is **efficient** and **robust**

Table 1: Tacking statistics: number of point feature per plane, plane per frame, plane tracking length and active planes in the state per frame

Dataset	Feat. / PL	PL / Frame	Track Len.	PL Active	Time (ms)
table_01	27.3 ± 13.1	2.7 ± 1.1	61.1 ± 227.6	1.1 ± 0.5	3.5 ± 0.7
table_02	82.0 ± 58.7	2.2 ± 1.3	49.1 ± 249.2	1.2 ± 0.6	4.1 ± 0.9
table_03	33.9 ± 21.3	3.0 ± 1.2	88.5 ± 337.4	1.5 ± 0.6	4.0 ± 0.7
V1_01	19.6 ± 13.3	2.9 ± 1.3	53.4 ± 74.0	0.9 ± 0.7	3.3 ± 0.7
V1_02	13.7 ± 10.9	1.7 ± 1.3	20.0 ± 26.8	0.3 ± 0.5	2.5 ± 0.8
V1_03	10.1 ± 9.4	0.7 ± 1.0	24.9 ± 26.0	0.0 ± 0.2	2.0 ± 0.7

Plane-Aided VIO System Overview



Plane: $^G\mathbf{\Pi} = {}^G\mathbf{n}\,{}^G d$ **Point:** $^G\mathbf{p}_f$

- **State Vector**

$$\mathbf{x}_k = \begin{bmatrix}\mathbf{x}_{I_k}^\top\ \mathbf{x}_C^\top\ \mathbf{x}_f^\top\ \mathbf{x}_\pi^\top\end{bmatrix}^\top \qquad \mathbf{x}_{I_k} = \begin{bmatrix}{}^{I_k}_G\bar{q}^\top\ {}^G\mathbf{p}_{I_k}^\top\ {}^G\mathbf{v}_{I_k}^\top\ \mathbf{b}_{g,k}^\top\ \mathbf{b}_{a,k}^\top\end{bmatrix}^\top$$

$$\mathbf{x}_f = \begin{bmatrix}{}^G\mathbf{p}_{f_1}^\top \dots {}^G\mathbf{p}_{f_g}^\top\end{bmatrix}^\top \qquad \mathbf{x}_\pi = \begin{bmatrix}{}^G\mathbf{\Pi}_1^\top \dots {}^G\mathbf{\Pi}_h^\top\end{bmatrix}^\top$$

- **Point-on-Plane constraints**

$$z_d = \left({}^G\mathbf{p}_f^\top\,{}^G\mathbf{n} - {}^G d\right) + n_d \tag{1}$$

- **Regularity-constrained measurement**

$$\begin{bmatrix}\tilde{\mathbf{z}}_c \\ \tilde{z}_d\end{bmatrix} = \begin{bmatrix}\mathbf{H}_T^c \\ \mathbf{0}\end{bmatrix}\tilde{\mathbf{x}}_C + \begin{bmatrix}\mathbf{H}_f^c \\ \mathbf{H}_f^d\end{bmatrix}{}^G\tilde{\mathbf{p}}_f + \begin{bmatrix}\mathbf{0} \\ \mathbf{H}_\pi^d\end{bmatrix}{}^G\tilde{\mathbf{\Pi}} + \begin{bmatrix}\mathbf{n}_c \\ n_d\end{bmatrix} \tag{2}$$

$$\Rightarrow \tilde{\mathbf{z}} = \mathbf{H}_T\tilde{\mathbf{x}}_C + \mathbf{H}_f\,{}^G\tilde{\mathbf{p}}_f + \mathbf{H}_\pi\,{}^G\tilde{\mathbf{\Pi}} + \mathbf{n} \tag{3}$$

- **Planar point feature update**
  - SLAM plane + SLAM point: Standard EKF update
  - SLAM plane + MSCKF point

$$\tilde{\mathbf{z}} = \mathbf{H}_T\tilde{\mathbf{x}}_C + \cancel{\mathbf{H}_f\,{}^G\tilde{\mathbf{p}}_f} + \mathbf{H}_\pi\,{}^G\tilde{\mathbf{\Pi}} + \mathbf{n} \tag{4}$$

  - MSCKF plane + SLAM point:

$$\tilde{\mathbf{z}} = \mathbf{H}_T\tilde{\mathbf{x}}_C + \mathbf{H}_f\,{}^G\tilde{\mathbf{p}}_f + \cancel{\mathbf{H}_\pi\,{}^G\tilde{\mathbf{\Pi}}} + \mathbf{n} \tag{5}$$

  - MSCKF plane + MSCKF point:

$$\tilde{\mathbf{z}} = \mathbf{H}_T\tilde{\mathbf{x}}_T + \cancel{\begin{bmatrix}\mathbf{H}_f\ \mathbf{H}_\pi\end{bmatrix}\begin{bmatrix}{}^G\tilde{\mathbf{p}}_f \\ {}^G\tilde{\mathbf{\Pi}}\end{bmatrix}} + \mathbf{n} \tag{6}$$

Explore Our Code and Dataset!

RPNG
ROBOT PERCEPTION & NAVIGATION GROUP

ccchu@udel.edu
chuchuchen.net
pgeneva@udel.edu
pgeneva.com

Source Code

AR Table Dataset

Monte-Carlo Simulation

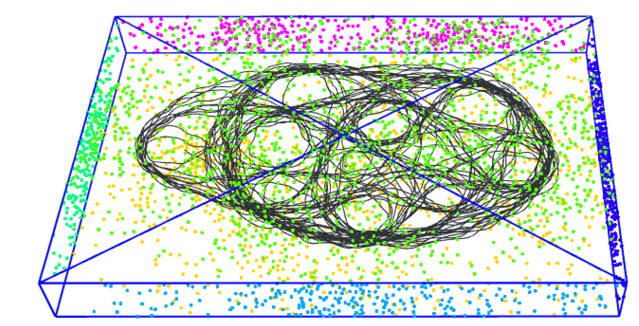


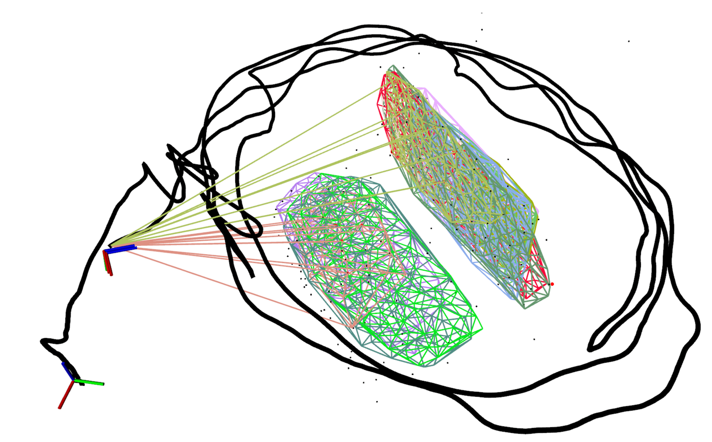
Table 2: Average 20 run RPE and NEES for different algorithm configurations. Units are in degree / cm

Algorithm	60m	80m	100m	120m	NEES(3)
M-PT	0.37 / 4.3	0.44 / 5.0	0.50 / 5.6	0.55 / 6.2	3.39 / 1.75
M-PT & M-PL	0.37 / 4.3	0.43 / 4.9	0.48 / 5.5	0.53 / 6.1	3.34 / 1.72
M-PT & MS-PL	**0.36** / **3.6**	**0.42** / **4.1**	**0.48** / **4.6**	**0.53** / **5.1**	3.99 / 1.44
MS-PT	0.30 / 3.6	0.35 / 4.1	0.40 / 4.6	0.43 / 5.1	3.45 / 1.63
MS-PT & M-PL	0.29 / 3.5	**0.33** / 4.0	**0.37** / 4.5	**0.41** / 4.9	3.09 / 1.44
MS-PT & MS-PL	**0.29** / **2.9**	0.35 / **3.3**	0.39 / **3.7**	0.42 / **4.1**	3.38 / 1.20

- **M**: MSCKF features, **S**: SLAM features, **PT**: point, **PL**: plane.
- Extend OpenVINS [1] simulator to generate room walls and visual points lying on each plane
- Planar regularities **improve** state estimation and constrain both in-state and out-of-state features

Real-World Experiments

- Tested on released Indoor AR Table and EuRoC MAV datasets
- Planar regularities improve state estimation performance
- Plane detection and tracking is robust and efficient, tracking planes for significant periods of time
- The system is computationally cheap and efficient



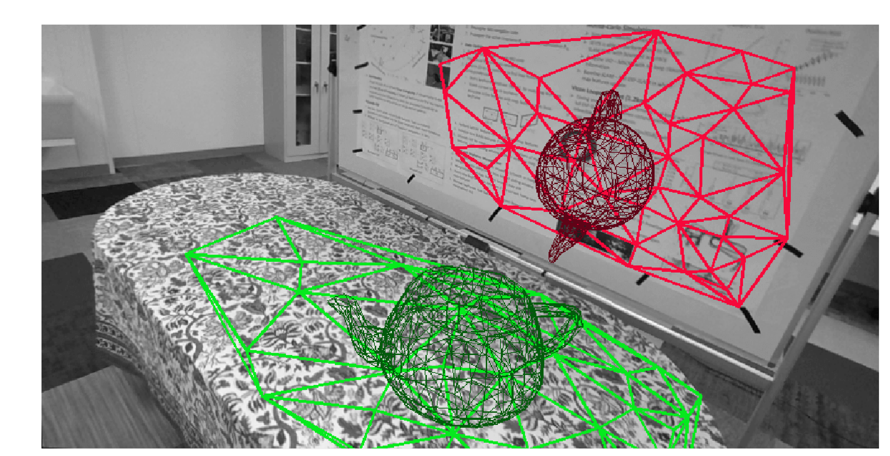


Table 3: ATE on self-collected Indoor AR Table and EuRoC MAV datasets (degree / cm). $n_d = 0.01$ was used.

Algorithm	table_01	table_02	table_03	V1_01	V1_02	V1_03	Time (ms)
M-PT	**0.45** / 6.8	0.85 / 2.4	1.37 / 5.6	0.83 / 8.6	1.57 / 9.1	2.50 / 15.5	8.7 ± 1.7
M-PT & M-PL	0.52 / 6.5	0.91 / 2.5	1.44 / 5.9	0.82 / 8.6	1.58 / 9.2	**2.45** / **15.3**	13.3 ± 3.2
M-PT & MS-PL	0.67 / **4.6**	**0.72** / **2.0**	**0.96** / **3.0**	**0.75** / **7.6**	**1.55** / **9.0**	2.50 / 15.5	13.9 ± 2.9
MS-PT	**1.15** / 5.7	1.79 / 4.1	2.41 / 6.9	1.32 / 8.4	1.58 / 7.0	**2.20** / **12.2**	9.4 ± 2.0
MS-PT & M-PL	1.32 / 5.5	0.89 / 2.5	**1.03** / **4.5**	**0.61** / **5.3**	1.58 / 7.5	2.32 / 12.5	15.0 ± 3.9
MS-PT & MS-PL	1.25 / **5.1**	**0.65** / **2.3**	1.05 / 4.6	0.75 / 6.9	**1.55** / **6.9**	2.41 / 12.5	14.7 ± 3.2
VINS-Fusion [2]	1.62 / 5.8	1.32 / 3.0	1.47 / 7.6	1.24 / 5.8	2.61 / 11.5	3.61 / 20.5	35.6 ± 17.0*
OKVIS [3]	2.48 / 9.0	2.01 / 7.7	3.94 / 15.3	0.72 / 8.3	2.01 / 14.5	10.47 / 107.4	85.5 ± 32.6*

* Timing for VINS-Fusion [2] and OKVIS [3] *only* reports their optimization time (no feature tracking).

[1] P. Geneva et al. "OpenVINS: A Research Platform for Visual-Inertial Estimation", ICRA, 2020. `https://github.com/rpng/open_vins`
[2] Qin et al. "VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator", TRO, 2018.
[3] Leutenegger et al. "Keyframe-Based Visual-Inertial SLAM using Nonlinear Optimization", RSS, 2013.