HybVIO for HILTI SLAM Challenge

Pekka Rantalankila

Spectacular AI, pekka.rantalankila@spectacularai.com

September 23, 2021

1 System overview

We use the HybVIO [1] method developed by our group. It is a filtering-based VIO part of the MSCKF [2] family, with an optional optimization-based SLAM module. The filtering is based on the EKF framework and supports stereo input, whereas the SLAM is monocular and similar to ORB-SLAM2 (based on OpenVSLAM [3]). The SLAM runs BA in the local neighborhood of the current position and does not perform loop closures.

While HybVIO has support for post-processing using global BA optimization, for this challenge we ran the system in a causal SLAM configuration.

2 Sensors

We utilize only the Alphasense IMU and the two front-facing cameras as stereo, all at maximum available framerates (approx. 10Hz and 200Hz). The cameras are installed upside-down relative to each other, which we handle by rotating the frames from one camera by 180 degrees and modifying the intrinsic and extrinsic camera parameters accordingly. We ignore frames that lack stereo pair with a matching timestamp.

We use static camera parameters provided by the dataset authors, with the Kannala-Brandt camera model [4]. The provided IMU bias values are not used, as we rely on online estimation of the biases and additionally a camera-to-IMU time shift value.

3 Algorithm parameters

We use for all the sequences the parameter configuration from [1] called *Normal SLAM*, without any modifications. Table 1 shows the relevant column from a table presented in [1]. See the article for meaning of the rows.

Table 1: Algorithm parameters.

Parameter		Normal SLAM
feature detector	type	GFTT
	subpixel adjustment	yes
feature tracker	max. features stereo	200
	max. itr.	20
	window size	31
visual updates	n_a	20
	n_{target}	20
	n_{FIFO}	17
SLAM	n_{BA}	20
	n_{matching}	20

Table 2: Hardware.

CPU	AMD Ryzen 5900X
GPU	Geforce RTX 3080 Ti
RAM	32GB
Disk type	SSD
OS	Ubuntu 21.04

4 Timing

We evaluated the method on a desktop machine, using GPU acceleration for some image processing operations. The hardware used is shown in Table 2, and the processing times in Table 3.

References

[1] Otto Seiskari, Pekka Rantalankila, Juho Kannala, Jerry Ylilammi, Esa Rahtu, and Arno Solin. HybVIO: Pushing the limits of real-time visual-inertial odometry. *CoRR*, abs/2106.11857, 2021.

Table 3: Timings for each sequence.

Sequence	Processing time (s)	Sequence length (s)
uzh tracking area run2	50	89
IC Office 1	108	200
Office Mitte 1	144	264
Parking 1	298	582
Basement 1	62	113
Basement 3	160	331
Basement 4	159	350
LAB Survey 2	80	136
Construction Site 1	110	199
Construction Site 2	212	399
Campus 1	222	430
Campus 2	193	375

[2] Anastasios I. Mourikis and Stergios I. Roumeliotis. A multi-state constraint Kalman filter for vision-aided inertial navigation. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pages 3565–3572, Rome, Italy, 2007.

[3] Shinya Sumikura, Mikiya Shibuya, and Ken Sakurada. OpenVSLAM: A versatile visual SLAM framework. *CoRR*, abs/1910.01122, 2019.

[4] Juho Kannala and Sami S. Brandt. A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(8):1335–1340, 2006.

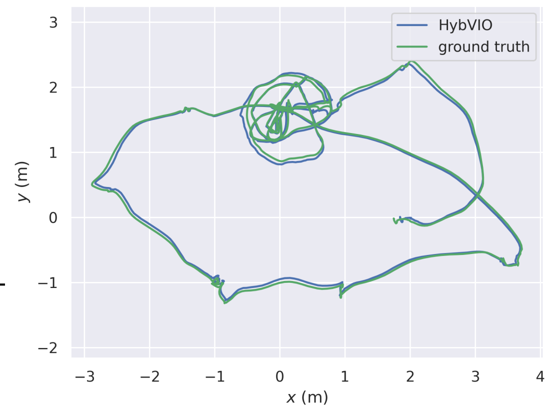

Figure 1: LAB Survey 2

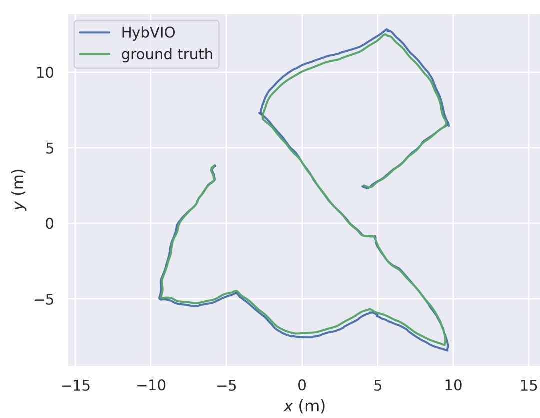

Figure 2: uzh tracking area run2