Exploring Event Camera-based Odometry for Planetary Robots

Florian Mahlknecht[1], Daniel Gehrig[2], Jeremy Nash[1], Friedrich M. Rockenbauer[1], Benjamin Morrell[1], Jeff Delaune[1], and Davide Scaramuzza[2]

***Abstract*—Due to their resilience to motion blur and high robustness in low-light and high dynamic range conditions, event cameras are poised to become enabling sensors for vision-based exploration on future Mars helicopter missions. However, existing event-based visual-inertial odometry (VIO) algorithms either suffer from high tracking errors or are brittle, since they cannot cope with significant depth uncertainties caused by an unforeseen loss of tracking or other effects. In this work, we introduce EKLT-VIO, which addresses both limitations by combining a state-of-the-art event-based frontend with a filter-based backend. This makes it both accurate and robust to uncertainties, outperforming event- and frame-based VIO algorithms on challenging benchmarks by 32%. In addition, we demonstrate accurate performance in hover-like conditions (outperforming existing event-based methods) as well as high robustness in newly collected Mars-like and high-dynamic-range sequences, where existing frame-based methods fail. In doing so, we show that event-based VIO is the way forward for vision-based exploration on Mars.**

***Index Terms*—Vision-Based Navigation; Space Robotics and Automation; Visual-Inertial SLAM**

Multimedia Material:

For code and dataset please visit https://uzh-rpg.github.io/eklt-vio/.

I. Introduction

STATE estimation is critical for enabling autonomous navigation and control of mobile robots, with widespread applications from space exploration to household cleaning robots. There exist well-established algorithms, such as [1], [2], [3], [4] which estimate ego-motion from visual-inertial data. However, vision-based navigation is drastically impacted by the known limitations of conventional cameras, such as motion blur and low dynamic range.

Manuscript received: February, 24, 2022; Revised May, 20, 2022; Accepted June, 14, 2022.

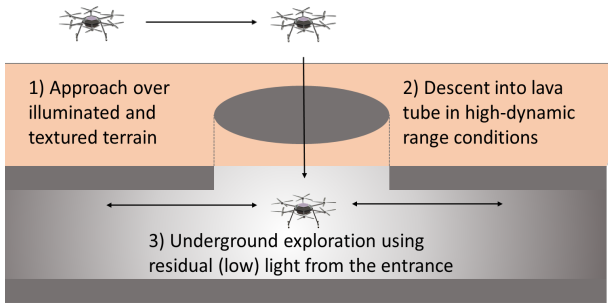
This paper was recommended for publication by Editor Eric Marchand upon evaluation of the Associate Editor and Reviewers' comments.

Part of this research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. © 2021. All rights reserved. The other part was carried out at the Robotics and Perception Group, University of Zurich, under contracts with the National Centre of Competence in Research (NCCR) Robotics through the Swiss National Science Foundation (SNSF) and the European Research Council (ERC) under grant agreement No. 51NF40_185543. We thank Konstantin Kalenberg for the feature prediction implementation improving EKLT's computational efficiency.

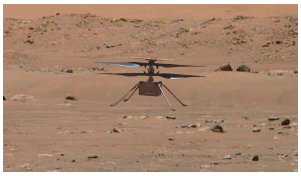
[1]F. Mahlknecht, J. Nash, F. M. Rockenbauer, B. Morrell and J. Delaune are with the Jet Propulsion Laboratory, California Institute of Technology, USA.

[2]D. Gehrig and D. Scaramuzza are with the Robotics and Perception Group, University of Zurich, Switzerland https://rpg.ifi.uzh.ch

Digital Object Identifier (DOI): see top of this page.



(a) Mission scenario



(b) Ingenuity Mars Helicopter

(c) Lava tube

Fig. 1: New mission scenario (a) enabled by EKLT-VIO for a Mars helicopter (b) scouting the entrance of lava tubes (c).

Event cameras promise to address these limitations [5]. Unlike a standard camera that measures absolute pixel brightness using a global exposure time, event camera pixels independently detect positive or negative brightness changes at microsecond resolution. Event cameras can provide data at 1 MHz and 120 dB dynamic range, both orders of magnitude greater than what can be achieved with a standard 60 dB camera. This leads to a significant reduction in motion blur, and enables operation in high dynamic range (HDR), low light, and fast motion conditions [6], [7].

On the application side, computer vision is increasingly used in modern planetary robotic missions [8], [9], [10], [11], [12].The resilient properties of event cameras may enable robots to explore in conditions where frame cameras cannot operate without introducing the size, weight, power, and range limitations of a 3D LiDAR.

In this paper, we focus on a scenario involving the exploration of the entrance of a lava tube by a Mars helicopter, as illustrated in Fig. 1. Lava tubes are natural tunnels created by lava flows in volcanic terrains. Those found on Mars have drawn significant attention because of the possibility that they might host microbial life [13]. The natural protection from radiation offered by lava tubes also makes them candidates to host the first human base on Mars.

Before sending a robotic mission [14] or astronauts to a specific lava tube, it would be desirable to scout and map

several locations. Mars helicopters are candidate platforms to scout multiple lava tubes throughout a single mission. However, Mars helicopters cannot fly LiDARs and have to rely on passive cameras for navigation. Frame cameras are ill-suited to explore lava tubes because of the HDR conditions created by the shadow at the entrance of the tube, as well as the low-light conditions once inside. This capability gap is filled by event cameras, which offer the potential to explore and map the lava tube for potentially tens of meters using residual light from the entrance.

Mars helicopters come with their own requirements on the state estimation system [15], [10]. They must rely on small passive lightweight cameras to observe the full state up to scale and gravity direction. The camera is fused with an inertial measurement unit (IMU), which makes gravity observable, enables a high estimation rate, and acts as an emergency landing sensor in case of camera failure. Finally, a laser range finder is used to observe scale in the absence of accelerometer excitation. The estimation backend must be able to handle feature depth uncertainty associated with helicopter hovering and rotation-only dynamics. Due to this uncertainty successful feature triangulation is often inhibited in these cases, leading to failure of optimization-based backends, which critically rely on triangulated features. By contrast, filter-based approaches leverage priors to initialize depth measurements and thus do not suffer from this issue [16]. This proved critical in Ingenuity Mars helicopter's sixth flight on Mars, where an image timestamping anomaly caused roll and pitch oscillations greater than 20 degrees [17]. Such rotations cause a loss of features, which can lead to estimation failure in non-filter-based state estimation approaches, which are fundamentally unable to handle the depth uncertainty of the new feature tracks without a dedicated re-initialization procedure.

State-of-the-art event-based VIO methods are unsuitable in these conditions since they either *(i)* use optimization-based backends, which do not model depth uncertainty, thus featuring brittle performance in mission-typical rotation-only motion, or when a significant portion of features are lost [6], or *(ii)* show a higher tracking error, due to the use of suboptimal event-based frontends [18]. Image-based VIO methods such as [15], [19] have addressed this by using depth priors [15] or motion classification [19].

In this work, we introduce EKLT-VIO, which builds on the EKF backend in [15] which handles pure rotational motion, and combines it with the state-of-the-art event-based feature tracker EKLT [20], thereby addressing the limitations above. EKLT-VIO is accurate, outperforming previous state-of-the-art frame-based and event-based methods on the challenging Event-Camera Dataset [21], with a 32% improvement in terms of pose accuracy. Moreover, by leveraging depth uncertainty it reduces its reliance on triangulating features, which both increase robustness during purely-rotational motion, and facilitates rapid initialization, both of which are limitations of existing optimization-based methods. This is because they require lengthy bootstrapping sequences, which would be impractical on Mars. Additionally, it maintains state-estimate, even when frame-based methods fail due to excessive motion blur. We show that our event-based EKLT frontend has a higher tracking performance than existing methods on newly collected data in Mars-like conditions. This demonstrates the viability of our EKLT-VIO on Mars. Our contributions are:

- We introduce EKLT-VIO, an event-based VIO method that combines an accurate state-of-the-art event-based feature tracker EKLT with an EKF backend. It outperforms state-of-the-art event- and frame-based methods, reducing the overall tracking error by 32%.
- We show accurate and robust tracking even in rotation-only sequences, which are closest to the hover-like scenarios experienced by Mars helicopters, outperforming optimization-based and frame-based methods.
- We outperform existing methods on newly collected Mars-like sequences collected in the JPL Mars Yard and Wells Cave for planetary exploration.

II. Related Work

Frame-based VIO: An overview of existing approaches is discussed in [22]. Frame-based VIO algorithms can be roughly segmented into two classes: optimization-based and filter-based algorithms [22]. While both algorithms focus on tracking camera poses by minimizing both visual and inertial residuals, optimization-based methods solve this by performing iterative Gauss-Newton steps, while filtering-based methods achieve this through Kalman Filtering steps.

Since optimizing both 3D landmarks (*i.e.*, SLAM features) and camera poses is costly, several filtering-based techniques exist that focus on refining camera poses from bearing measurements (*i.e.*, multi-state constraint Kalman filter (MSCKF) features [23]) directly. However, MSCKF features need translational motion and provide updates only after the full feature track is known. The filtering-based approach, xVIO [15], combines the advantages of both features, with robustness to depth uncertainty in rotation-only motion and computational efficiency with many MSCKF features.

Event-based VIO: First event-based, 6-DOF visual odometry (VO) algorithms only started to appear recently [24], [25]. Later work incorporated an IMU to improve tracking performance and stability [26], [18], achieving impressive tracking on a fast spinning leash [26]. Despite their robustness, these methods are affected by drift due to the differential nature of the used sensors. This is why Ultimate SLAM (USLAM) [6] used a combination of events, frames, and IMU, all provided by the Dynamic and Active Vision Sensor (DAVIS) [27]. It tracks FAST corners [28] on frames and motion-compensated event frames separately using the Lucas-Kanade tracker (KLT) [29] and fuses these feature tracks with IMU measurements in a sliding window.

While addressing drift, USLAM still relies on a sliding window optimization scheme, which is expensive and does not allow pose-only optimization through the use of MSCKF features. Moreover, its FAST/KLT frontend, first introduced in [26], is optimized explicitly for frame-like inputs and was shown to transfer suboptimally to event-based frames [20]. In this work, we incorporate the state-of-the-art event-based tracker EKLT [20], which takes a more principled approach

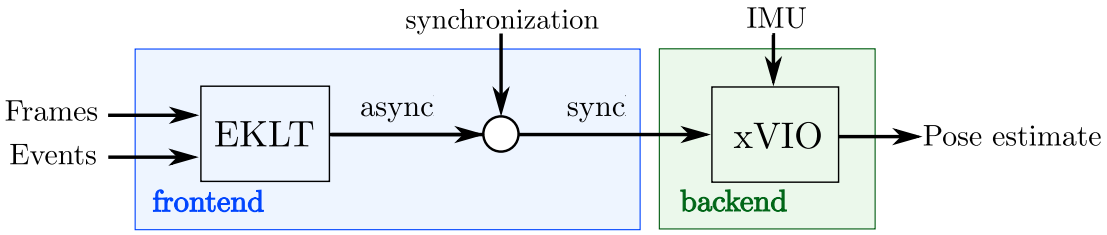


Fig. 2: We combine the feature tracker EKLT, which use frames and events, with the filter-based backend xVIO to enable low-translation state-estimation. In contrast to standard, frame-based VIO, an additional synchronization step converts asynchronous tracks to synchronous matches, which are used by the backend. This enables variable-rate backend updates.

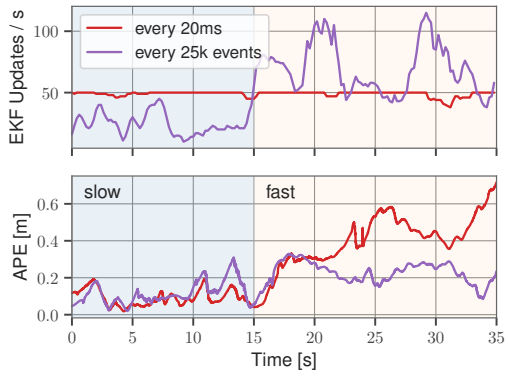


Fig. 3: Synchronous feature updates (red) tend to generate too many updates during slow sequences and too few during fast sequences, leading to high tracking error. Our irregular update strategy (purple) adapts to the event-rate, and thus maintains low tracking error in both scenarios.

to fusing events and frames, and thus achieves better feature tracking performance compared to [6], [26].

III. Methodology

In this section we present EKLT-VIO, which is illustrated in Fig. 2. It is an event-based VIO algorithm based on the state-of-the-art event tracker EKLT, coupled with a filter-based xVIO backend.

A. Backend

We start by providing a summary of the xVIO backend. For more details see [15]. The backend fuses data from an inertial measurement unit (IMU) and feature tracks from the frontend. It does this by using an extended Kalman filter (EKF) with an IMU state $\mathbf{x}_I$ and a visual state $\mathbf{x}_V$:

$$\mathbf{x} = \begin{bmatrix}\mathbf{x}_I^\intercal & \mathbf{x}_V^\intercal\end{bmatrix}^\intercal \tag{1}$$

The IMU state follows an inertial propagation scheme as described in [30]. The visual state $\mathbf{x}_V$ is split into sliding window states $\mathbf{x}_S$ and feature states $\mathbf{x}_F$:

$$\mathbf{x}_V = \begin{bmatrix}\mathbf{x}_F^\intercal & \mathbf{x}_S^\intercal\end{bmatrix}^\intercal, \quad \mathbf{x}_F = \begin{bmatrix}\mathbf{f}_1 & \dots & \mathbf{f}_N\end{bmatrix}^\intercal \tag{2}$$

$$\mathbf{x}_S = \begin{bmatrix}\mathbf{p}_w^{c_1\intercal} & \dots & \mathbf{p}_w^{c_M\intercal} & \mathbf{q}_w^{c_1\intercal} & \dots & \mathbf{q}_w^{c_M\intercal}\end{bmatrix}^\intercal \tag{3}$$

The sliding window states contain the positions, $\mathbf{p}_w^{c_i}$, and attitudes parameterized as quaternions, $\mathbf{q}_w^{c_i}$, of the last M camera poses $\{c_i\}$ with respect to a world frame $\{w\}$. The feature states contain the 3D positions, $\mathbf{f}_j$, of N SLAM features. In this work $N = 15$ and $M = 10$.

We use a discrete-time VIO approach, as opposed to one based on splines [31], [32], [33], [34]. Although they can incorporate event-data [31] more elegantly they are notoriously computationally expensive [31] and less established. This is why we opt for discrete-time VIO and leave splines for future work.

SLAM features are parametrized with respect to an anchor pose $\mathbf{p}_w^{c_{a_j}}$ in the sliding window, and defined as $\mathbf{f}_j = [\alpha_j \quad \beta_j \quad \rho_j]$ with α_j and β_j being normalized image coordinates and ρ_j being the inverse depth. Each time the feature tracks are updated, each SLAM feature j is converted from inverse-depth to Cartesian coordinates in the associated anchor camera frame $\{c_{a_j}\}$.

$$\mathbf{p}_{c_i}^j = \mathbf{C}(\mathbf{q}_w^{c_i})\left(\mathbf{p}_w^{c_{a_j}} + \frac{1}{\rho_j}\mathbf{C}(\mathbf{q}_w^{c_{a_j}})^\intercal \begin{bmatrix}\alpha_j \\ \beta_j \\ 1\end{bmatrix} - \mathbf{p}_w^{c_i}\right), \tag{4}$$

The measurement model is the normalized feature:

$$\mathbf{z}_j = \pi(\mathbf{p}_{c_i}^j) + \mathbf{n}_j, \quad \pi(\mathbf{x}) = \begin{bmatrix}x_1/x_3 & x_2/x_3\end{bmatrix}^T, \tag{5}$$

where $\pi(\mathbf{x})$ performs feature projection, $\mathbf{n}_j$ is Gaussian noise, and $\mathbf{z}_j$ are the new feature observations by the frontend, expressed in normalized image coordinates. Eqs. (4) and (5) can be used to develop the EKF update by linearizing the SLAM feature reprojection. Details are given in [15].

In addition to SLAM features, the backend maintains MSCKF features that additionally constrain the camera poses without an explicit inverse depth. MSCKF features are thus not part of the state, resulting in a smaller computational cost per feature. They need to be observed for the last $2 \leq m \leq M$ frames, providing a corresponding observation for each pose in the sliding window. MSCKF features require triangulation using those pose priors, so they can only be processed once a track with significant translation is observed. Successfully triangulated MSCKF features are used to initialize SLAM features. When there is insufficient translation for triangulation, xVIO instead initializes the inverse depth with $\rho_0 = \frac{1}{2d_{\text{min}}}$ and uncertainty $\sigma_0 = \frac{1}{4d_{\text{min}}}$, corresponding to a semi-infinite depth

prior, and discards the MSCKF feature track [35]. This depth prior is especially useful during pure rotation or initialization, where few features can be triangulated, since it can directly contribute to reducing the state covariance.

B. Frontend

Here we provide a summary of our EKLT frontend, and refer the reader to [20] for more details. EKLT tracks Harris corners, extracted on frames, by aligning the predicted and measured brightness increment in a patch around the corners. It minimizes the normalized distance between these patches to recover the warping parameters $\mathbf{p}$ and normalize optical flow $\mathbf{v}$ as

$$\{\mathbf{p}, \mathbf{v}\} = \arg\min_{\mathbf{p},\mathbf{v}} \left\| \frac{\Delta L(\mathbf{u})}{\|\Delta L(\mathbf{u})\|} - \frac{\Delta \hat{L}(\mathbf{u}, \mathbf{p}, \mathbf{v})}{\left\|\Delta \hat{L}(\mathbf{u}, \mathbf{p}, \mathbf{v})\right\|} \right\|. \quad (6)$$

While ΔL is defined as an aggregation of events in a local patch, $\Delta \hat{L}$ is defined as the negative dot product between the local log image gradient and optical flow vector, following the linearized event generation model [36]. Here $W(\mathbf{u}, \mathbf{p})$ aligns the image gradient with the measured brightness increments according to the alignment parameters p. EKLT minimizes Eq. (6) using Gauss-Newton and the Ceres library[37], and recovers alignment parameters p and optical flow v. As opposed to the reference implementation of EKLT, which optimizes in a sliding window fashion after a fixed number of events, we trigger the optimization only when the adaptive number of events is reached, using each event batch only once. This entails a significant speed-up without loss in accuracy.

C. Frontend Adaptations

Asynchronous feature updates: We convert the asynchronous feature tracks provided by EKLT to synchronous feature tracks via a synchronization step (Fig. 2. This step produces a temporally synchronized list of feature positions, which are passed to the backend. The backend uses the associated correspondences $\mathbf{z}_i \iff \mathbf{z}_j$ together with consecutive camera poses c_i and c_j to update the state as discussed in Sec. III-A. It is performed by selecting the most recent feature in the currently tracked feature set and extrapolating the positions of all other features to its timestamp. We synchronize every time, a fixed number of events n_e is triggered, enabling variable-rate backend updates. We empirically found $n_e = 3200$ to work best, see Tab. I. We argue that reducing n_e will introduce additional noisy updates to the EKF which reduce the accuracy, while having too high n_e makes our approach less robust during high-speed motion.

n_e	500	1000	3200	4800	7200	9200	15000	20000
MMPE	0.57	0.55	**0.49**	0.59	0.60	0.68	0.83	1.72

TABLE I: Median Mean Position Error (MMPE) [%] on the Event Camera Dataset for different EKF event update thresholds

This variable rate allows our algorithm to adapt to the scene dynamics (Fig. 3), leading to fewer EKF updates in slow sequences (Fig.3, left) and a lower tracking error during high-speed sequences, compared to fixed rate updating. These features motivate the use of an event-based frontend since a purely frame-based one is limited by the framerate of the camera. Although, this may lead to drift in purely stationary environments where no events are triggered, this can easily be amended by enforcing a minimal backend update rate. Or by enforcing a no-motion prior when the event rate goes below a threshold, as in [6].

Outlier rejection: For EKLT we exclusively reject outliers by setting a maximum threshold on the optimized residual of the alignment score in Eq. (6). This allows outliers to be rejected quickly, without the need for costly geometric verification, such as 8-point RANSAC.

IV. Experiments

We start by validating our approach on standard benchmarks in Sec. IV-B, where we compare the performance of EKLT-VIO against state-of-the-art event-based [18], frame-based [15] and event- and frame-based methods [6]. To study the effect on the event-based feature tracker, we also study an additional baseline, based on the HASTE feature tracker [38]. We then proceed to demonstrate the suitability of our approach on two important use-cases motivated by the Mars exploration scenario: *(i)* pure rotational motion, imitating hover-like conditions on Mars (Sec. IV-C), and *(ii)* challenging HDR conditions on newly collected datasets in the JPL Mars Yard and at the entrance of the Wells Cave, emulating the entry into lava tubes (Sec. IV-D).

A. Baselines and Compared Methods

USLAM [6] is an event- and frame-based VIO method, which fuses feature tracks derived from frames and event-frames in an optimization-based backend.
EVIO [18] uses only events and IMU. Events are used to generate asynchronous feature tracks, which are then fused in a filter-based backend. Since open-source code is not available, we only report results on real sequences.
KLT-VIO [15] is a frame-based VIO method that fuses feature tracks based on FAST/KLT in a filter-based backend, and is specifically designed for use during helicopter flight.
HASTE-VIO [38] Finally, we combine the state-of-the-art purely event-based tracker HASTE [38] with xVIO as an additional baseline. Similar to EKLT, it produces asynchronous feature tracks which are first synchronized using the method described in Sec. III-C, before being fed into the backend.

B. Real Data

We benchmark our methods on the Event-Camera Dataset [21] , recorded with a DAVIS 240C [27] with synchronized images, events, IMU measurements, and very fast hand-held motions in an HDR scenario. An OptiTrack is used for ground-truth camera trajectories. We evaluate the pose tracking accuracy using the same protocol as [6], and report

Dataset	USLAM* [6]		USLAM [6]		EVIO [18]		KLT-VIO [15]		HASTE-VIO		EKLT-VIO (**ours**)	
	MPE	MYE	MPE	MYE	MPE	MYE	MPE	MYE	MPE	MYE	MPE	MYE
Boxes 6DOF	0.30	0.04	**0.68**	**0.03**	4.13	0.92	0.97	0.05	2.03	**0.03**	0.84	0.09
Boxes Translation	0.27	0.02	1.12	2.62	3.18	0.67	**0.33**	**0.08**	2.55	0.46	0.48	0.25
Dynamic 6DOF	0.19	0.10	0.76	0.09	3.38	1.20	0.78	**0.03**	**0.52**	0.06	0.79	0.06
Dynamic Translation	0.18	0.15	0.63	0.22	1.06	0.25	0.55	0.06	1.32	0.06	**0.40**	**0.04**
HDR Boxes	0.37	0.03	1.01	0.31	3.22	0.15	**0.42**	**0.02**	1.75	0.09	0.46	0.06
HDR Poster	0.31	0.05	1.48	0.09	1.41	0.13	0.77	0.03	0.57	**0.02**	**0.65**	0.04
Poster 6DOF	0.28	0.07	0.59	0.03	5.79	1.84	0.69	**0.02**	1.50	0.03	**0.35**	**0.02**
Poster Translation	0.12	0.04	0.24	**0.02**	1.59	0.38	**0.16**	**0.02**	1.34	**0.02**	0.35	0.03
Shapes 6DOF	0.10	0.04	1.07	0.03	2.52	0.61	1.80	0.03	2.35	**0.02**	**0.60**	0.03
Shapes Translation	0.26	0.06	1.36	**0.01**	4.56	2.60	1.38	0.02	1.09	0.02	**0.51**	0.03
Average	0.24	0.06	0.89	0.34	3.08	0.88	0.79	**0.04**	1.50	0.08	**0.54**	0.07

*per-sequence hyperparameter tuning and correct IMU bias intialization

TABLE II: Pose estimate accuracy comparison on the Event-Camera Dataset [21] in terms of mean position error (MPE) in % and mean yaw error (MYE) in deg/m. Grayed-out results with (*) by USLAM [6] were achieved through per-sequence parameter tuning and correct IMU bias initialization, while results in black used a single parameter set, tuned on all sequences simultaneously, and were initialized with an IMU bias of zero.

Dataset	USLAM [6]		KLT-VIO [15]		HASTE-VIO		EKLT-VIO (**ours**)	
	MPE	MYE	MPE	MYE	MPE	MYE	MPE	MYE
Dynamic Rotation	*unfeasible*		9.97	**0.13**	**6.22**	2.32	7.71	1.52
Boxes Rotation			*diverging*		20.57	**1.32**	**8.78**	1.36
Poster Rotation			*diverging*		3.96	**0.09**	**1.44**	**0.09**
Shapes Rotation			*diverging*		*diverging*		**6.95**	**4.59**

TABLE III: Mean position and yaw error (MPE and MYE) in % and deg/m on rotation-only sequences.

mean position error (MPE) in % of the total trajectory length and mean yaw error (MYE) in deg/m in Tab.II.

In [6], USLAM uses different parameters for each sequence, and correct IMU bias initialization, resulting in the gray columns in Tab. II. We mark this method as USLAM*. However, on Mars, VIO systems should perform robustly in unknown environments, making, parameter tuning and bias initialization infeasible. For this reason, we retune the parameters of USLAM to perform best on all sequences simultaneously resulting in the black values in Tab. II. All other methods were tuned in the same way. Comparing USLAM* with USLAM shows that IMU bias initialization, and per-sequence hyperparameter tuning are clearly important to achieve low tracking error, reducing the error from 0.89% to 0.24%. Our EKLT-VIO, on the other hand, achieves an average error of 0.54% without bias initialization, 39% lower than USLAM. This improvement indicates that EKLT-VIO is simultaneously more robust to zero IMU bias initialization, and per-sequence hyperparameter tuning.

In terms of position error, EKLT-VIO outperforms all other methods on 5 out of 10 sequences. With an average MPE of 0.54% EKLT-VIO shows a 32% lower MPE than runner-up KLT-VIO with 0.79%. Finally, with a 3.08% MPE, EVIO [18] is outperformed by EKLT-VIO by 82%.

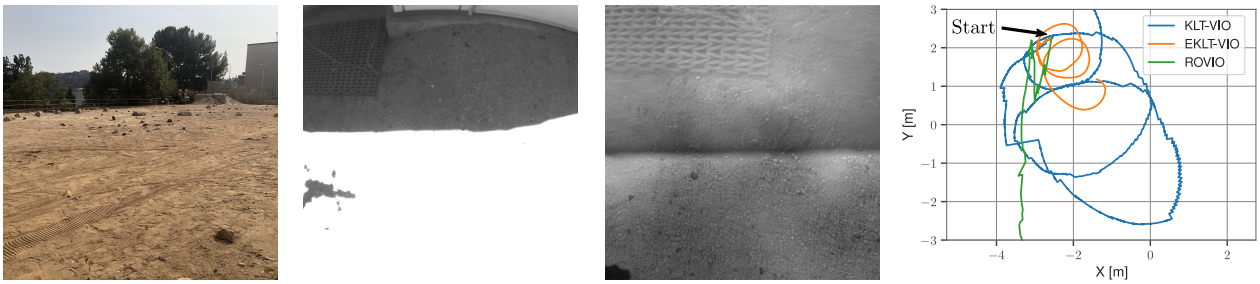
C. Rotation-only sequences

As a next step, we show the suitability of EKLT-VIO in a Mars Mission-like scenario. To do this, we evaluate all methods on the rotation-only sequences of the Event-Camera Dataset, which are challenging for optimization-based backends such as USLAM [6]. Similar to hover-like conditions expected during Mars missions, these sequences translate only little compared to the average scene depth, which poses a challenge for keyframe generation and triangulation.

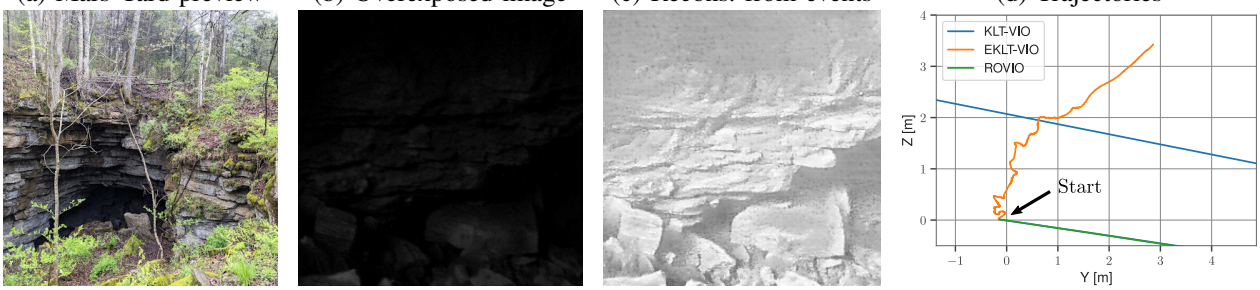
We adopt the same evaluation protocol as before and report results for all methods in Tab. III. We observed during this experiment that USLAM did not initialize during these sequences since it could never detect sufficient translation to insert a new keyframe, and it is thus marked with *unfeasible*. Frame-based KLT-VIO tracks well for the first 30s, but diverges in the second part, where rapid shaking motion causes motion blur on the frames, and high feature displacements, both of which significantly impact the accuracy of the KLT frontend. This leads to a diverging state estimate. By contrast, event-based methods EKLT-VIO and HASTE-VIO can track robustly, because their event-based front-ends are unaffected by motion-blur. EKLT-VIO, however, is the only method to converge on all sequences and yields a consistently lower tracking error compared to all compared methods. In summary, EKLT-VIO leverages the advantages of event-based frontends for robust high-speed tracking and the advantages of a filter-based backend to fuse small translational motions. This shows that EKLT-VIO is most suitable in these conditions.

D. Mars-mission Scenario: Wells Cave and JPL Mars yard

Finally, we show the capabilities of EKLT-VIO in Mars-like exploration scenarios, by comparing it to image-based methods KLT-VIO [16], ORB-SLAM3 [41], OpenVINS [42], VINS-Mono [3] and ROVIO [40] on sequences recorded at the JPL Mars Yard (Fig. 4 (a)), and Wells Cave Nature



(a) Mars Yard preview (b) Overexposed image (c) Recons. from events (d) Trajectories



(e) Wells Cave preview (f) Underexposed image (g) Recons. from events (h) Trajectories

Fig. 4: In the Mars Yard (a) we test HDR conditions which cause severe oversaturation artefacts in standard images (b). Instead in the Wells Cave (e) we study low light scenarios encountered in lava tubes, which cause undersaturation (f). HDR images reconstructed from events [39] (c,g) do not suffer from these artefacts, and are used by our method. As a result, we outperform existing frame-based approaches KLT-VIO [15] and ROVIO [40] on both trajectories.

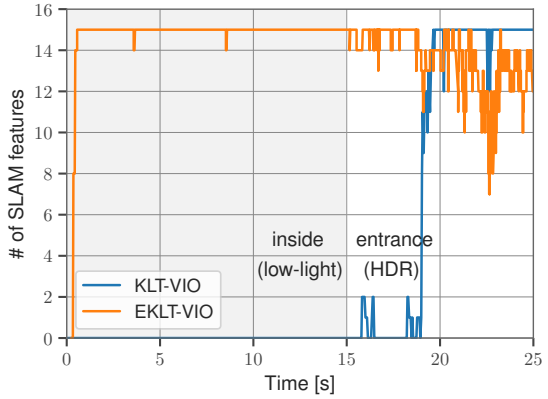


Fig. 5: Tracked features on the Wells Cave sequence. While KLT-VIO and ROVIO quickly diverge, due to lacking features (c), EKLT-VIO can track successfully.

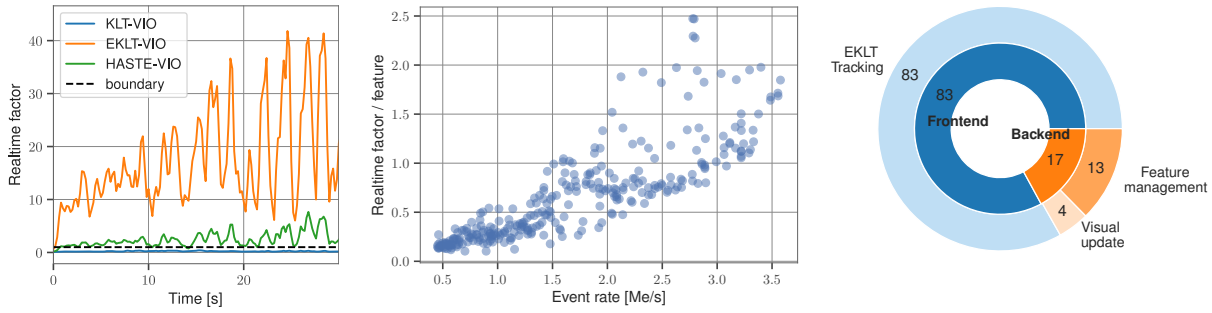
Preserve (Fig. 4 (e)). The Mars Yard sequence features rapid illumination changes that challenge the autoexposure and result overexposures in the images (Fig. 4 (b)). The Wells Cave instead is a cave system used by JPL to emulate lava tubes on Mars. It features a low illumination, leading to underexposure in the images (Fig. 4 (f)). In the Wells Cave we use the DAVIS 346[27], and in the Mars Yard, we use a mvBlueFOX-MLC200wG standard camera, a DVXplorer event camera, and an MPU9250 IMU.

Here we show that EKLT-VIO can run on events alone, by using images reconstructed from events provided by the method E2VID [39]. They feature a much higher dynamic range than the standard images (Fig. 4 (c,g)). We reconstruct frames every 15'000 events, resulting in an HDR video used by our method. For a resolution of 640 $\times$ 480 these images can be provided with 30 FPS on a Quadro RTX 4000 GPU. However, EKLT-VIO only needs a subset of these images, since it only uses them for feature initialization.

Mars Yard: The trajectory used in this analysis is a hand-held circular motion with a diameter of 1.5 meters over a sharp shadow with increasing speed. The trajectories tracked by all methods are shown in Fig. 4 (d). While EKLT-VIO consistently tracks the circular motion for at least two revolutions, filter-based methods KLT-VIO and ROVIO diverge due to a lack of features caused by motion blur and HDR conditions. The optimization-based methods ORB-SLAM3 and VINS-MONO fail to initialize, since the sequence starts directly from hover, and misses an initialization trajectory, with which to generate an initial map. OpenVINS fails to initialize due to missing parallax. These methods are therefore not plotted. This shows that thanks to the use of an event-based frontend and filter-based backend EKLT-VIO can overcome this condition.
Wells Cave: Finally, the trajectories in the Wells Cave, for all methods are shown in Fig. 4 (h). Only filter-based methods KLT-VIO and ROVIO manage to initialize, but diverge quickly. EKLT-VIO tracks consistently, until reaching the tunnel entrance. Again, ORB-SLAM3 and VINS-Mono fail to initialize and therefore are not plotted.OpenVINS fails to initialize due to missing features. As shown in Fig. 5, EKLT-VIO consistently maintains SLAM features, while KLT-VIO only does so once it exits the cave.

E. Limitations

We study EKLT-VIO, KLT-VIO, and HASTE-VIO in terms of their real-time factor (RTF, Fig. 6 (a)) and report the RTF



(a) Realtime factor (b) Realtime factor vs. Event rate (c) Computational pie chart

Fig. 6: Real-time factor (RTF) (a) for EKLT-VIO (orange), HASTE-VIO (green) and KLT-VIO (blue) on *Poster 6DOF*. The RTF per tracked feature (b) increases with the event rate. Our method can process 89'000 events per second when tracking 45 features. As seen in (c), EKLT-VIO spends most of its computation time tracking features.

per feature (b) and computation allocations (c) for EKLT-VIO. We conduct all our experiments on a laptop with an Intel i7-7700HQ quadcore processor, exploiting however only a single core in the current implementation. The RTF measures how much time is spent to process a second of real-time, and RTF<1 indicates real-time performance. As seen in Fig. 6 (a) there exists a clear speed-accuracy trade-off between EKLT-VIO, HASTE-VIO, and KLT-VIO, since EKLT-VIO achieves a maximum real-time factor of around 45. Note that this is 45 times slower than real-time. For EKLT-VIO, the real-time factor correlates with the event rate (Fig. 6 (b)), which depends on the scene texture and camera speed. On *Poster 6DOF* it can process 89'000 kEv/sec.

F. Speedup Strategies

Fig.6 (c) shows that, the EKLT frontend remains the bottleneck, which directs future work toward speeding up EKLT. Tab. IV illustrates three speedup strategies to achieve realtime capabilities evaluated *Poster 6DOF*. (i) We reduce the number of tracked frontend features from 45 to 15, (ii) we increased n_e, the number of events before triggering an update, by a factor of two and (iii) we reduce the event rate with random filtering (RF), randomly keeping every r^{th} event, or refractory period filtering (RPF), where events within a time τ of the previous event are discarded. To improve the convergence in (ii) we additionally implemented IMU-based feature prediction [40], to improve the initial guess. While naive RF degrades performance, RPF with $\tau = 10$ ms reduces the median RTF to 7.7. Reducing the frontend features results in an RTF of 8.7, and, when combined with filtering, leads to an RTF of 4.2. These steps lead to a minimal increase of the MPE from 0.36 to 0.41. Setting $n_e = 6400$ results in an RTF of 9.7, while reducing the MPE from 0.36 to 0.24. However, when combined with additional filtering, we found that the method diverges with an MPE of 3.79, but a lower RTF of 2.05. The remaining gap can be closed by software-side techniques, such as distributing the workload to multiple cores (see https://github.com/Doch88/rpg_eklt_multithreading). There, up to four cores were parallelized, leading to a 3.6-fold speedup.

Speedup method	MPE	MYE	RTF Max	RTF Median
Baseline	0.36	0.02	43.6	17.9
RF $r = 2$	*diverging*		11.2	5.20
RF $r = 5$	*diverging*		5.70	2.20
RPF ($\tau = 1$ ms)	0.27	0.02	37.3	15.40
RPF ($\tau = 10$ ms)	0.48	0.02	15.7	7.70
$n_e = 6400$	0.24	0.02	21.2	9.70
15 Features	0.31	0.02	18.6	8.70
15 Features, RPF ($\tau = 10$ ms)	0.41	0.02	8.20	4.20
15 Features, RPF ($\tau = 10$ ms), $n_e = 6400$	3.79	0.02	4.39	2.05

TABLE IV: Real-time factor speedup on *Poster 6DOF*. We compare random filtering (RF), refractory period filtering (RPF), reducing the number of features, and increasing n_e. Our baseline tracks 45 features and updates each feature, every $n_e = 3200$ events. RTF > 1 is slower than real-time.

V. Conclusion

Future planetary missions, require us to venture into previously inaccessible domains, such as lava-tubes on Mars, which pose challenging lighting conditions for traditional image-based VIO. We explored the use of event cameras, which promise to shed light in these domains due to their high dynamic range. We present EKLT-VIO which integrates the state-of-the-art feature tracker EKLT with the filter-based backend xVIO thus leveraging the advantages of both. The event-based frontend provides robust high-speed feature measurements even in low-light and HDR scenarios while the filter-based backend addresses the limitations of traditional optimization-based VIO algorithms in near-hovering conditions. We show an evaluation on Mars-like sequences and challenging hand-held sequences of the Event-Camera dataset. On these sequences, we demonstrate the robust pose tracking the performance of our methods, showing a mean position error reduction of up to 32% compared to event- and frame-based state-of-the-art methods. Additionally, we showcase the advantages of our backend and frontend in the first successful evaluation on the rotation-only sequences of the Event-Camera Dataset with fast motion and challenging lighting conditions. Finally, we demonstrate our method's robustness in visually challenging conditions recorded in the JPL Mars Yard and in the Wells Cave, replicating our mission scenario. To spur further research in this direction, we open-source the implementation of this work and release our Mars-like sequences.

REFERENCES

[1] M. Li and A. I. Mourikis, "Optimization-based estimator design for vision-aided inertial navigation," in *Robotics: Science and Systems*. Berlin Germany, 2013, pp. 241–248.

[2] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual-inertial SLAM using nonlinear optimization," *Int. J. Robot. Research*, 2015.

[3] T. Qin, P. Li, and S. Shen, "VINS-Mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 1004–1020, 2018.

[4] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual–inertial odometry," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, 2016.

[5] G. Gallego, T. Delbruck, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. Davison, J. Conradt, K. Daniilidis, and D. Scaramuzza, "Event-based vision: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2020.

[6] A. Rosinol Vidal, H. Rebecq, T. Horstschaefer, and D. Scaramuzza, "Ultimate SLAM? combining events, images, and IMU for robust visual SLAM in HDR and high speed scenarios," *IEEE Robot. Autom. Lett.*, vol. 3, no. 2, pp. 994–1001, Apr. 2018.

[7] S. Sun, G. Cioffi, C. De Visser, and D. Scaramuzza, "Autonomous quadrotor flight despite rotor failure with onboard vision sensors: Frames vs. events," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 580–587, 2021.

[8] A. Johnson, S. Aaron, J. Chang, Y. Cheng, J. Montgomery, S. Mohan, S. Schroeder, B. Tweddle, N. Trawny, and J. Zheng, "The lander vision system for Mars 2020 entry descent and landing," in *AAS Guidance, Navigation, and Control Conference*, 2017.

[9] B. Bos, M. Ravine, M. Caplinger, J. Schaffner, J. Ladewig, R. Olds, C. Norman, D. Huish, M. Hughes, S. Anderson, D. Lorenz, A. May, C. Adam, D. Nelson, M. Moreau, D. Kubitschek, K. Getzandanner, K. Gordon, A. Eberhardt, and D. Lauretta, "Touch and go camera system (TAGCAMS) for the OSIRIS-REx asteroid sample return mission," *Space Science Reviews*, vol. 214, 01 2018.

[10] D. S. Bayard, D. T. Conway, R. Brockers, J. H. Delaune, L. H. Matthies, H. F. Grip, G. B. Merewether, T. L. Brown, and A. M. San Martin, "Vision-based navigation for the NASA Mars helicopter," in *AIAA Scitech 2019 Forum*, 2019, p. 1411.

[11] M. Maimone, Y. Cheng, and L. Matthies, "Two years of visual odometry on the Mars exploration rovers," *J. Field Robot.*, vol. 24, no. 3, pp. 169–186, 2007.

[12] J. Delaune, R. Brockers, D. S. Bayard, H. Dor, R. Hewitt, J. Sawoniewicz, G. Kubiak, T. Tzanetos, L. Matthies, and J. Balaram, "Extended navigation capabilities for a future mars science helicopter concept," in *IEEE Aerospace Conference*, 2020, pp. 1–10.

[13] B. Carrier, D. Beaty, M. Meyer, J. Blank, L. Chou, S. DasSarma, D. Des Marais, J. Eigenbrode, N. Grefenstette, N. Lanza, A. Schuerger, P. Schwendner, H. Smith, C. Stoker, J. Tarnas, K. Webster, C. Bakermans, B. Baxter, M. Bell, and J. G. Xu, "Mars extant life: What's next? conference report," *Astrobiology*, vol. 20, 05 2020.

[14] C. Phillips-Lander, J. Wynne, N. Chanover, C. Demirel-Floyd, K. Uckert, K. Williams, T. Titus, J. Blank, P. Boston, K. Mitchell, D. Wyrick, S. Shkolyar, K. Retherford, and F. J. Martín-Torres, "Mars astrobiological cave and internal habitability explorer (MACIE): A new frontiers mission concept," in *38th Mars Exploration Program Analysis Group*, 04 2020.

[15] J. Delaune, D. S. Bayard, and R. Brockers, "Range-visual-inertial odometry: Scale observability without excitation," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2421–2428, 2021.

[16] ——, "xVIO: A range-visual-inertial odometry framework," *arXiv preprint arXiv:2010.06677*, 2020.

[17] H. Grip, "Surviving an In-Flight Anomaly: What Happened on Ingenuity's Sixth Flight," NASA, Tech. Rep., 2021.

[18] A. Z. Zhu, N. Atanasov, and K. Daniilidis, "Event-based visual inertial odometry," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2017, pp. 5816–5824.

[19] D. G. Kottas, K. J. Wu, and S. I. Roumeliotis, "Detecting and dealing with hovering maneuvers in vision-aided inertial navigation systems," *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, 2013.

[20] D. Gehrig, H. Rebecq, G. Gallego, and D. Scaramuzza, "EKLT: Asynchronous photometric feature tracking using events and frames," *Int. J. Comput. Vis.*, 2019.

[21] E. Mueggler, H. Rebecq, G. Gallego, T. Delbruck, and D. Scaramuzza, "The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and SLAM," *Int. J. Robot. Research*, vol. 36, no. 2, pp. 142–149, 2017.

[22] J. Delmerico and D. Scaramuzza, "A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2018.

[23] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2007, pp. 3565–3572.

[24] H. Kim, S. Leutenegger, and A. J. Davison, "Real-time 3D reconstruction and 6-DoF tracking with an event camera," in *Eur. Conf. Comput. Vis. (ECCV)*, 2016, pp. 349–364.

[25] H. Rebecq, T. Horstschäfer, G. Gallego, and D. Scaramuzza, "EVO: A geometric approach to event-based 6-DOF parallel tracking and mapping in real-time," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 593–600, 2017.

[26] H. Rebecq, T. Horstschaefer, and D. Scaramuzza, "Real-time visual-inertial odometry for event cameras using keyframe-based nonlinear optimization," in *British Mach. Vis. Conf. (BMVC)*, 2017.

[27] C. Brandli, R. Berner, M. Yang, S.-C. Liu, and T. Delbruck, "A 240x180 130dB 3μs latency global shutter spatiotemporal vision sensor," *IEEE J. Solid-State Circuits*, vol. 49, no. 10, pp. 2333–2341, 2014.

[28] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Eur. Conf. Comput. Vis. (ECCV)*, 2006, pp. 430–443.

[29] B. D. Lucas, T. Kanade, *et al.*, "An iterative image registration technique with an application to stereo vision." Vancouver, British Columbia, 1981.

[30] S. Weiss, M. Achtelik, S. Lynen, M. Chli, and R. Siegwart, "Real-time onboard visual-inertial state estimation and self-calibration of MAVs in unknown environments," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2012.

[31] E. Mueggler, G. Gallego, H. Rebecq, and D. Scaramuzza, "Continuous-time visual-inertial odometry for event cameras," *IEEE Trans. Robot.*, vol. 34, no. 6, pp. 1425–1440, Dec. 2018.

[32] G. Cioffi, T. Ciesleski, and D. Scaramuzza, "Continuous-time vs. discrete-time vision-based slam: A comparative study," in *IEEE Robotics and Automation Letters (RA-L)*, 2022.

[33] M. Li and A. Mourikis, "Vision-aided inertial navigation with rolling-shutter cameras." *Int. J. Robot. Research*, 2014.

[34] K. Eckenhoff, P. Geneva, and G. Huang, "MIMC-VINS: A versatile and resilient multi-imu multi-camera visual-inertial navigation system," *IEEE Trans. Robot.*, 2021.

[35] J. Montiel, J. Civera, and A. Davison, "Unified inverse depth parametrization for monocular SLAM," in *Robotics: Science and Systems (RSS)*, 2006.

[36] G. Gallego, J. E. A. Lund, E. Mueggler, H. Rebecq, T. Delbruck, and D. Scaramuzza, "Event-based, 6-DOF camera tracking from photometric depth maps," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 10, pp. 2402–2412, Oct. 2018.

[37] S. Agarwal, K. Mierle, and T. C. S. Team, "Ceres Solver," 3 2022. [Online]. Available: https://github.com/ceres-solver/ceres-solver

[38] I. Alzugaray and M. Chli, "Asynchronous multi-hypothesis tracking of features with event cameras," in *2019 International Conference on 3D Vision (3DV)*, 2019, pp. 269–278.

[39] H. Rebecq, R. Ranftl, V. Koltun, and D. Scaramuzza, "High speed and high dynamic range video with an event camera," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2019.

[40] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct EKF-based approach," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, 2015.

[41] C. Campos, R. Elvira, J. Rodriguez, J. Montiel, and J. Tardos, "Orb-slam3: An accurate open-source library for visual, visual-inertial and multi-map slam," vol. 37, no. 6, 2021, pp. 1874–1890.

[42] P. Geneva, K. Eckenhoff, W. Lee, Y. Yang, and G. Huang, "OpenVINS: A research platform for visual-inertial estimation," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, Paris, France, 2020.