Design of a Real-time Asynchronous Monocular Odometry for Planetary Exploration

Beñat Iñigo*†, Florian Steidle*, Wolfgang Stürzl*
*Institute of Robotics and Mechatronics
German Aerospace Center (DLR)
† University of Zaragoza

***Abstract*—We describe our preliminary design of a real-time asynchronous event-based monocular odometry for planetary exploration. Operating under strict computational constraints, planetary rovers frequently encounter complex, unpredictable environments that demand high-speed sensing and robustness to high dynamic range (HDR) lighting. Event cameras address these needs by reporting asynchronous, pixel-wise brightness changes with microsecond resolution, significantly reducing data bandwidth while maintaining robustness in extreme lighting conditions. We propose an approach based on an Error-State Kalman Filter (ESKF) that leverages this asynchronous event stream to continuously estimate camera ego-motion. The camera state is updated with every tracked position output generated by RATE, a real-time asynchronous feature tracker.**

I. Introduction

Due to their low power consumption and data bandwidth requirements, as well as high dynamic range and low latency, event cameras have great potential for planetary exploration robotics. Having conducted field tests at the DLR Moon Mars test site [1] with the Scout rover, see Fig. 1, we decided to employ event cameras for pose estimation. In the following we describe our preliminary design of an event-based monocular visual odometry with the focus on low latency pose estimation on standard off-the-shelf CPUs. Note, that although fusing data from different sensors, e.g. camera and inertial measurement unit (IMU), is usually more robust, there are also space rovers that predominantly rely on vision, like the MMX rover IDEFIX that due to size constraints and low gravity on Mars Moon Phobos does not use inertial measurements for state estimation [2].

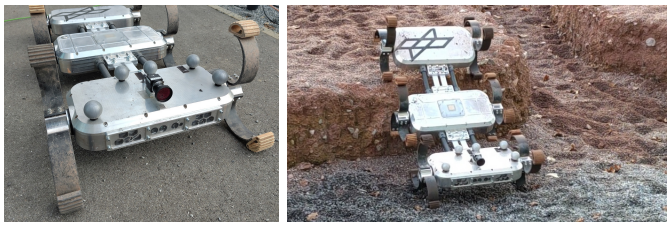


Fig. 1: Scout rover with event camera mounted on top its head segment

II. Related Work

Pure event-driven monocular visual odometry remains sparsely explored, with only few published methods that function independently of both frame data and inertial sensors. One of the first approaches [3] employs three interleaved filters for motion, intensity and depth estimation. EVO [4] which uses parallel tracking and mapping, does not require intensity estimation, allowing it to run faster. Pose tracking of the camera is realized using Lukas-Kanade-based alignment of a template created by projecting the 3D map and an event image created by accumulating events within an short time span. To initialize the system, the local scene is assumed to be planar and fronto-parallel to the sensor. An asynchronous optimization-based method is described in [5]. A Gaussian Process (GP) is used to model the trajectory of the camera. Similar to our approach HASTE is employed for feature tracking. However, the implementation of the full pipeline did not run in real time. In [6], a sliding-window optimizer for GP regression is introduced to reduce the computational complexity. While the event-based feature tracking seems to be similar to our frontend, we employ an Error-State Kalman Filter (ESKF) for online estimation of camera poses, as described in the next section. Recent learning-based approaches, e.g. DEVO [7], can estimate camera trajectories with high accuracy. However, they need powerful GPUs currently not available on planetary rovers.

III. Monocular Odometry with ESKF

For detection, tracking and management of features we employ RATE [8]. To facilitate real-time tracking of multiple event-based features, RATE distributes features into sub-images. Features are detected with Shi-Tomasi corner detection [9] on a binarized Surface of Active Events [10], [11] and then tracked asynchronously with HASTE [12]. Changes of feature states (position ± 1 px, orientation $\pm 4°$) are reported via ROS messages.

We added a second output to RATE, a list of IDs for features to be deleted. Features are removed when they leave the image boundaries, when their tracking quality degrades, or when a higher-scoring feature is detected within the same sub-region.

A. Error-State Kalman Filter (ESKF)

Messages received from the frontend are used to update the state defined as

$$x = \begin{bmatrix} {}^{G}p_C^\top & {}^{G}q_C^\top & {}^{G}v_C^\top & {}^{G}p_{f_1}^\top & \dots & {}^{G}p_{f_n}^\top \end{bmatrix}^\top \quad (1)$$

Accepted to the Challenges and Opportunities of Neuromorphic Field Robotics and Automation IEEE ICRA Workshop - 2026

where ${}^{G}p_C$, ${}^{G}q_C$ and ${}^{G}v_C$ represent the position, orientation and velocity of the camera in the global frame, and ${}^{G}p_{f_1} \dots {}^{G}p_{f_n}$ represent the set of all landmarks reconstructed by triangulating features from corresponding observations. As new track updates are received from RATE composed as $h_j = (id_j, t_j, (u_j, v_j))$, i.e. feature ID, timestamp, and new feature position, we can update the state and possibly triangulate new features.

We adopt an error state formulation of the Kalman filter, defining the the discrepancy between the true state x_{true} and the nominal state as

$$\delta x = \begin{bmatrix} {}^{G}\delta p_C^\top & {}^{G}\delta\theta_C^\top & {}^{G}\delta v_C^\top & {}^{G}\delta p_{f_1}^\top & \dots & {}^{G}\delta p_{f_n}^\top \end{bmatrix}^\top \quad (2)$$

such that $x_{\text{true}} = x \oplus \delta x$. The orientation error ${}^{G}\delta\theta_C$ is defined as a 3D rotation vector as a minimal representation. We represent the uncertainty of the system via the covariance matrix P:

$$P = \begin{bmatrix} P_{cc} & P_{cf} \\ P_{fc} & P_{ff} \end{bmatrix} \quad (3)$$

where P_{cc} represents the camera pose and velocity covariance, P_{ff} the map features covariance, and P_{cf} the cross-covariance between them.

1) Propagation: We propagate the state forward in time δt assuming a constant velocity and constant orientation kinematic model. The prediction step of the filter is given by

$$\begin{aligned} \hat{x}_{k|k-1} &= f(\hat{x}_{k-1|k-1}, u_{k-1}) \\ P_{k|k-1} &= F_k P_{k-1|k-1} F_k^\top + Q_{k-1} \end{aligned} \quad (4)$$

where $P_{k|k-1} \in \Re^{(9+3M)\times(9+3M)}$ is the error state covariance for M landmarks, $Q_{k-1} \in \Re^{(9+3M)\times(9+3M)}$ the process uncertainty and $F_k \in \Re^{(9+3M)\times(9+3M)}$ the Jacobian of the error-state kinematics evaluated at the zero-mean error state:

$$F_k = \left.\frac{\partial f}{\partial \delta x_{k-1}}\right|_{\delta x_{k-1}=0} \quad (5)$$

2) Correction: Every time we receive a feature update, we check whether it corresponds to an existing feature in the state. If the feature is new, we clone the initial state for further initialization, as detailed in Section III-B.

If it is an existing feature, we compute the innovation $\tilde{y}_k$ defined as the difference between the actual observation z_k and the predicted measurement $\hat{z}_k$:

$$\tilde{y}_k = z_k - h(\hat{x}_{k|k-1}) = z_k - \hat{z}_k \quad (6)$$

We present the measurement model $h(\cdot)$ by considering an update of a single feature track h_j, observed from pose $({}^{G}p_{C_j}, {}^{G}q_{C_j})$. We project 3D feature j into the camera's image plane:

$$\hat{z}_j = \begin{bmatrix} \hat{u}_j \\ \hat{v}_j \end{bmatrix} = \begin{bmatrix} f_x \frac{{}^{C}X_j}{{}^{C}Z_j} + c_x \\ f_y \frac{{}^{C}Y_j}{{}^{C}Z_j} + c_y \end{bmatrix} \quad (7)$$

by first transforming the global feature position ${}^{G}p_{f_j}$ into the camera frame C:

$${}^{C}p_{f_j} = \begin{bmatrix} {}^{C}X_j \\ {}^{C}Y_j \\ {}^{C}Z_j \end{bmatrix} = R({}^{G}q_C)^\top \left({}^{G}p_{f_j} - {}^{G}p_C\right) \quad (8)$$

where ${}^{G}p_{f_j}$ is the 3D feature position in the camera frame. Note that we receive already undistorted pixel coordinates from RATE via a Look-Up Table (LUT).

As measurements are asynchronous, we process one measurement at a time, defining a sparse measurement Jacobian matrix $H_k \in \Re^{2\times(9+3M)}$ mapping to the state vector:

$$H_k = \begin{bmatrix} H_{pos} & H_{rot} & 0_{2\times3} & 0_{2\times3} & \dots & H_{f_j} & \dots & 0_{2\times3} \end{bmatrix} \quad (9)$$

where H_{pos} and $H_{rot} \in \Re^{2\times3}$ are the Jacobians of the pixel error with respect to the camera position and orientation, and $H_{f_j} \in \Re^{2\times3}$ is the Jacobian with respect to the 3D landmark, evaluated at the zero error state:

$$\begin{aligned} H_{pos} &= \left.\frac{\partial z}{\partial {}^{G}\delta p_C}\right|_{\delta x_{k-1}=0} \\ H_{rot} &= \left.\frac{\partial z}{\partial {}^{G}\delta \theta_C}\right|_{\delta x_{k-1}=0} \\ H_{f_i} &= \left.\frac{\partial z}{\partial {}^{G}\delta p_{f_j}}\right|_{\delta x_{k-1}=0} \end{aligned} \quad (10)$$

We compute the Kalman gain $K_k \in \Re^{(9+3M)\times2}$ and update the state and covariance:

$$\begin{aligned} S_k &= H_k P_{k|k-1} H_k^\top + R_k \\ K_t &= P_{k|k-1} H_k^\top S_k^{-1} \\ \delta\hat{x}_{k|k} &= K_k \tilde{y}_k \\ \hat{x}_{k|k} &= \hat{x}_{k|k-1} \oplus \delta\hat{x}_{k|k} \\ P_{k|k} &= (I - K_k H_k) P_{k|k-1} \end{aligned} \quad (11)$$

where $\oplus$ defines the composition between the estimated nominal state and the error state.

3) Reset: Finally, we apply the ESKF error reset operation, defined by function $g(\cdot)$ as:

$$\delta x \leftarrow g(\delta x) = \delta x \ominus \delta\hat{x}_{k|k-1} \quad (12)$$

where $\ominus$ stands for the composition inverse of $\oplus$. The reset operation on the state and covariance is thus:

$$\begin{aligned} \delta\hat{x}_{k|k} &\leftarrow 0 \\ P_{k|k} &\leftarrow G P_{k|k} G^\top \end{aligned} \quad (13)$$

with G denoting the Jacobian of the reset operation:

$$G = \left.\frac{\partial g}{\partial \delta x}\right|_{\delta x_k = \delta\hat{x}_{k|k}} \quad (14)$$

B. 3D Feature/Landmark Initialization

If the feature track $h_j = (id_j, t_j, (u_j, v_j))$ does not correspond to any existing landmark in the state, we must save the camera pose from which this new feature was observed for the first time for future triangulation. Because this past pose depends on future corrections, we save it by augmenting the state using a stochastic cloning update [13]. We delay triangulation until sufficient parallax is achieved, at which point we augment the state with the newly triangulated feature.

Considering a single measurement z_k, the respective augmented state and covariance matrix become:

$$\begin{aligned}\delta\bar{x} &= C\left[\delta x_k\right]_{(9+3M)} = \begin{bmatrix}\delta x_k \\ \delta x_{clone}\end{bmatrix}_{(9+3M+6)} \\ \bar{P} &= CP_{k|k}C^\top \\ C &= \begin{bmatrix}I \\ I^*\end{bmatrix}_{(9+3M+6)\times(9+3M)}\end{aligned} \tag{15}$$

where I^* is a sparse selection matrix that extracts the 6-DoF camera pose (position and orientation) from the current state to be cloned.

To ensure real-time performance, we marginalize out lost features from the state vector by removing its corresponding rows and columns from the state vector and the covariance matrix.

Once sufficient parallax is established between the cloned state and the current nominal state, we triangulate the new feature position and add it to the state. As the total number of landmarks is not known a priori, the state vector expands incrementally. We can define the state expansion function $q(\cdot)$ for feature i as:

$$\begin{aligned}\bar{x}_k &= q(x_{k-1}, z_k, z_{k-j}, {}^G p_{C_{k-1}}, {}^G p_{C_{k-j}}) \\ &= \begin{bmatrix} x_{k-1} \\ g({}^G p_{C_{k-1}}, {}^G q_{C_{k-1}}, {}^G p_{C_{k-j}}, {}^G q_{C_{k-j}}, z_k, z_{k-j})\end{bmatrix} \\ &= \begin{bmatrix} x_{k-1} \\ {}^G p_{f_{i_{k-1}}}\end{bmatrix}\end{aligned} \tag{16}$$

where ${}^G p_{C_{k-j}}$ represents the camera pose cloned for this specific feature, and $g(\cdot)$ is the triangulation function that computes the feature's position ${}^G p_{f_i}$.

After a new landmark j is triangulated, the expanded state vector becomes:

$$\bar{x}_{k+1} \leftarrow \begin{bmatrix} x_k & {}^G p_{f_j}\end{bmatrix}^\top \tag{17}$$

By linearizing the triangulation function, we estimate the error of the 3D landmark position:

$${}^G\delta p_{f_j} \approx G_x \delta x_k + G_z \delta z_k \tag{18}$$

where G_x is the Jacobian of the triangulation with respect to the augmented state vector and G_z is the Jacobian with respect to the measurement noise. Due to the high non-linearity of the inverse-depth formulation, G_x and G_z are evaluated numerically via finite differences:

$$G_x \approx \frac{\Delta g_k}{\Delta x_k},\ G_z \approx \frac{\Delta g_k}{\Delta z_k} \tag{19}$$

We then augment the covariance matrix accordingly:

$$\bar{P} = \begin{bmatrix} P_{xx} & P_{xm} \\ P_{mx} & P_{mm}\end{bmatrix} = \begin{bmatrix} P & PG_x^\top \\ G_x P & G_x P G_x^\top + G_z R G_z^\top\end{bmatrix} \tag{20}$$

C. Filter Initialization using Homography

Our current ESKF-based approach assumes that the 3D position of at least a small number of features are known or can be triangulated. Therefore we need a separate procedure to bootstrap the system at the start when neither 3D landmarks nor camera poses are known, which is based on a classical computer vision method. Because state updates of features are not synchronized like those from conventional frame-based cameras, we group them into sets, see Fig. 2. We wait until the first N feature updates arrive, which are then used as a reference set. The current set is initialized as a clone of the reference and updated every time a track update from RATE arrives. In order to make the tracks more stable, we apply a Kalman filter to each individual feature to act as a smoother. Once sufficient parallax is achieved, we initialize the system, i.e. estimate the motion between the current and the reference set using homography and then triangulate feature 3D positions. Since we currently rely on homography for this setup, we choose a planar dataset.

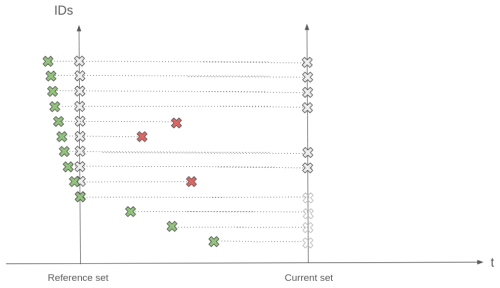


Fig. 2: Event sets for initial triangulation, green showing birth and red showing termination of tracks. Vertical axis for IDs and horizontal axis for time.

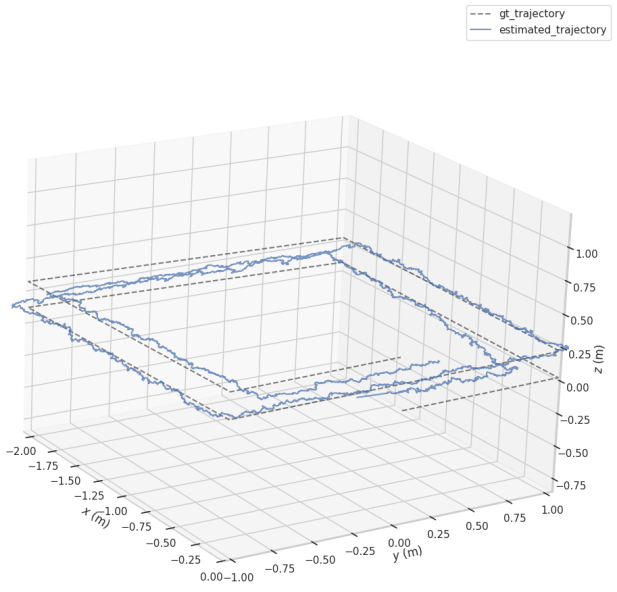
IV. Preliminary Results

To validate the proposed Error-State Kalman Filter (ESKF) pipeline for asynchronous event-based tracking, we conducted preliminary evaluations in both a controlled simulation environment and on a real-world event camera dataset.

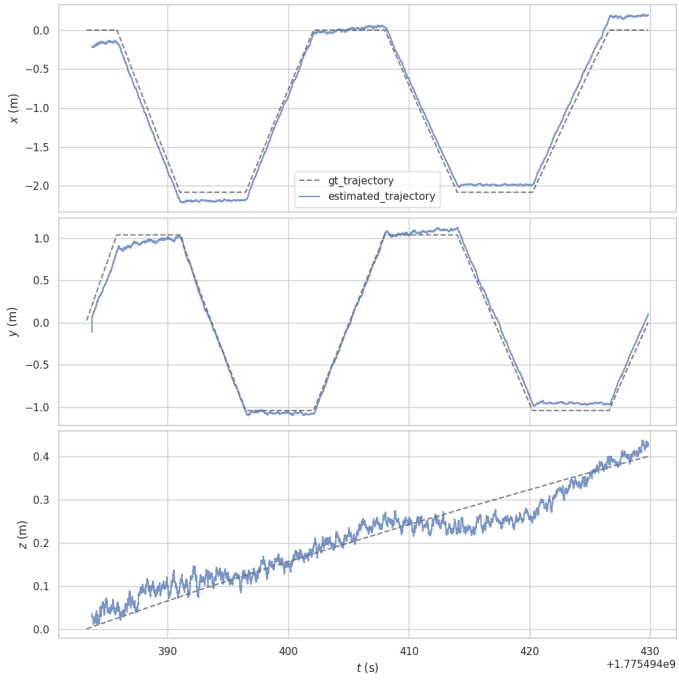
A. Simulation Evaluation

To rigorously evaluate the filter's behavior, we developed a custom simulation environment. The simulator generates spatially distributed 3D landmarks that dynamically appear and disappear, mimicking the realistic scenario of physical features being initialized and subsequently lost. The tracking front-end processes asynchronous events generated from a simulated sensor. To emulate the asynchronous nature of event cameras and the continuous feature updates from the RATE tracker, the active 3D landmarks are projected onto the 2D image plane sequentially at random order. Furthermore, to evaluate the system's robustness against the measurement noise inherent to real event sensors, we introduced a synthetic reprojection error by injecting zero-mean Gaussian noise $\mathcal{N}(0, \sigma^2)$ to the 2D pixel coordinates.

Because the system operates as a purely monocular pipeline without IMU integration, state propagation relies strictly on a constant-velocity kinematic model. To account for unmodeled dynamics when undergoing sharp accelerations, we empirically tune the process noise covariance as $\sigma_a, \sigma_w = 2.0$, and the measurement noise is set to $\sigma_u = \sigma_v = 1.0$ pixels.



(a) Robot trajectory



(b) Trajectory error

Fig. 3: Plot shows the estimated trajectory against the ground truth for noise level $\sigma = 1.0$ pixel. For convenience, [14] was used for evaluation. After alignment mean absolute pose error is 0.065 m.

Due to the lack of metric scale observability, the estimated trajectory was aligned to the ground truth using a Sim(3) Umeyama alignment. Preliminary simulation results validate the core asynchronous update formulation, see Fig. 3 for an example. The high-frequency variance observed in the estimated trajectory is characteristic of heuristically tuned noise covariance matrices, where the measurement noise covariance R_k currently outweighs the process noise covariance Q_k, leading to over-correction from noisy pixel observations.

B. Real-World Evaluation

For real-world validation, we evaluated our system using the widely benchmarked UZH DAVIS Event Camera Dataset [15]. Specifically, initial testing was conducted on the wall poster sequence.

Following with the simulation methodology, the estimated trajectory was aligned to the ground truth using a Sim(3) Umeyama alignment. A representative example is shown in Fig. 4. After successful initialization, the camera pose could be tracked with an RMS of 0.06 m. After about 17.5 s pose estimation fails due to loss of the majority of feature tracks.

V. Next Steps

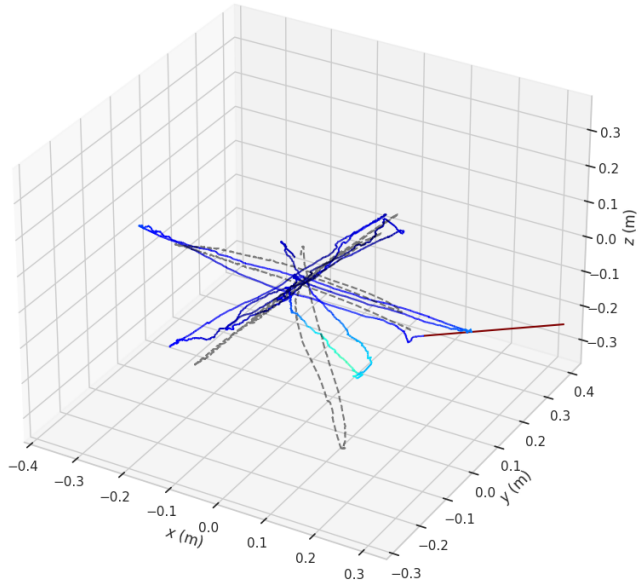
There are at least two aspects of the described approach that we want to improve in the near future.

a) Improved pose estimation during feature initialization: In time intervals where the number of features in the filter state is low due to tracking losses, accuracy of the estimated camera poses can drop significantly. In these cases of tracking loss or feature deletion, RATE attempts to detect and track new features. However, even if successful, features cannot be triangulated without sufficient parallax and thus camera pose estimation cannot be enhanced immediately. To improve the situation and make earlier use of recently detected and tracked features, we propose to add a measurement error based on epipolar constraint, which does not depend on features being triangulated.

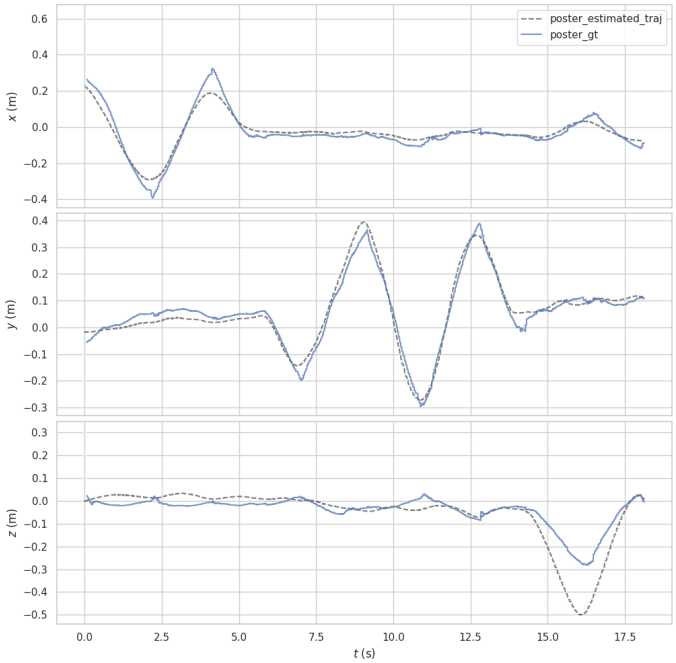
b) Improved filter initialization: For non-planar scenes, the current initialization based on homography has to be replaced with a more general approach, e.g. using 5pt-algorithm [16] in a RANSAC-based approach [17]. However, a filter-based initialization, possibly using epipolar residuals, is preferred as it better aligns with the asynchronous stream of feature state updates.

We are currently trying to address (a) and (b) with a modified approach which will be tested and evaluated with respect to run-time, latency and accuracy against the described method and existing baseline approaches. Preliminary investigations show that CPU load is dominated by the frontend, i.e. feature tracking and management.

Scale ambiguity: While it is interesting to investigate what can be achieved with a single event camera, it is also clear that our monocular VO can estimate camera poses only up to an unknown scale unless additional information is given. In a planetary scenario this could be, for example, the lander, a reference tag of known size on the lander or an already



(a) Robot trajectory



(b) Trajectory error

Fig. 4: Plot shows the estimated trajectory against ground truth. After alignment mean absolute pose error is 0.06 m. Total trajectory length is approx. 6.5 m. The red line indicates initialization with homography.

mapped crater. Also, additional sensors that can provide scale information, like a laser range finder or an IMU, could be utilized.

REFERENCES

[1] M. Görner, J. Cebulsky, A. Dömel, M. Durner, R. Giubilato, M. Kuhne, M. G. Müller, K. Lakatos, P. Lehner, R. Lichtenheldt, B. Rebele, M. Roser, R. Sakagami, Y. Scheeler, M. Schuster, M. Schütt, W. Stürzl, M. Vayugundla, and A. Wedler, "The DLR Moon-Mars test site for robotic planetary exploration," in *International Conference on Space Robotics (iSpaRo)*, 2024.

[2] M. Vayugundla, T. Bodenmüller, L. Burkhard, M. Schuster, B.-M. Steinmetz, M. Sewtz, N. Borgsmüller, F. Buse, W. Stürzl, R. Giubilato, F. Schuler, M. G. Müller, M. Kuhne, J. Langwald, A. Lund, A. Wedler, R. Triebel, M. Smisek, and M. Grebenstein, "The DLR autonomous navigation experiment with the IDEFIX rover: Software architecture, autonomous navigation features and preliminary operations concept," in *IEEE Aerospace Conference (AERO)*, 2025.

[3] H. Kim, S. Leutenegger, and A. J. Davison, "Real-time 3D reconstruction and 6-DoF tracking with an event camera," in *European Conference on Computer Vision (ECCV)*, 2016, pp. 349–364.

[4] H. Rebecq, T. Horstschaefer, G. Gallego, and D. Scaramuzza, "EVO: A geometric approach to event-based 6-DOF parallel tracking and mapping in real time," *IEEE Robotics and Automation Letters (RAL)*, vol. 2, no. 2, pp. 593–600, 2017.

[5] D. Liu, A. Parra, Y. Latif, B. Chen, T.-J. Chin, and I. Reid, "Asynchronous optimisation for event-based visual odometry," in *International Conference on Robotics and Automation (ICRA)*, 2022, pp. 9432–9438.

[6] Z. Wang, X. Li, Y. Zhang, and P. Huang, "AsynEVO: Asynchronous event-driven visual odometry for pure event streams," 2025. [Online]. Available: https://arxiv.org/abs/2402.16398

[7] S. Klenk, M. Motzet, L. Koestler, and D. Cremers, "Deep event visual odometry," in *International Conference on 3D Vision (3DV)*, 2024, pp. 739–749.

[8] M. Ikura, C. L. Gentil, M. G. Müller, F. Schuler, A. Yamashita, and W. Stürzl, "RATE: Real-time asynchronous feature tracking with event cameras," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024, pp. 11 662–11 669.

[9] J. Shi and Tomasi, "Good features to track," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1994, pp. 593–600.

[10] R. Benosman, C. Clercq, X. Lagorce, S.-H. Ieng, and C. Bartolozzi, "Event-based visual flow," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 2, pp. 407–417, 2014.

[11] Ö. Yılmaz, C. Simon-Chane, and A. Histace, "Evaluation of event-based corner detectors," *Journal of Imaging*, vol. 7, no. 2, 2021.

[12] I. Alzugaray and M. Chli, "HASTE: Multi-hypothesis asynchronous speeded-up tracking of events," in *British Machine Vision Conference (BMVC)*, 2020.

[13] S. Roumeliotis and J. Burdick, "Stochastic cloning: a generalized framework for processing relative state measurements," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2002, pp. 1788–1795.

[14] M. Grupp, "evo: Python package for the evaluation of odometry and SLAM." https://github.com/MichaelGrupp/evo, 2017.

[15] 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," *The International Journal of Robotics Research*, vol. 36, no. 2, pp. 142–149, 2017.

[16] D. Nistér, "An efficient solution to the five-point relative pose problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 6, pp. 756–77, 2004.

[17] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, p. 381–395, 1981.