Event and Frame-based Stereo Odometry with Adaptive Frequency and Multisurface Matching

Antea Hadviger, Ivan Petrović, Ivan Marković

University of Zagreb Faculty of Electrical Engineering and Computing, Laboratory for Autonomous Systems and Mobile Robotics, Croatia
antea.hadviger@gmail.com, ivan.petrovic@fer.unizg.hr, ivan.markovic@fer.unizg.hr

Abstract. Event cameras are asynchronous sensors that capture motion information with high temporal resolution, low latency, and high dynamic range, making them suitable for scenarios with rapid motion or challenging illumination, where standard cameras commonly suffer from motion blur. In this paper, we introduce a feature-based stereo event-based visual odometry approach that adjusts the odometry frequency according to camera velocity. Our method involves feature matching through the utilization of multiple event time surfaces. Furthermore, we present a stereo visual odometry framework that integrates information from both events and standard frames through pose graph optimization. This integration aims to leverage the complementary strengths of both camera types. The effectiveness of the proposed method is assessed across various indoor and outdoor self-acquired sequences. The results demonstrate that combining event and frame-based odometries yields more accurate results than employing either method individually.

Keywords: event-based cameras, dynamic vision sensors, stereo vision, visual odometry

1 Introduction

Visual odometry is crucial in robotics for understanding and navigating environments by estimating camera motion from image sequences. Modern systems use traditional cameras, depth sensors, and event-based cameras to improve accuracy. Event cameras, or dynamic vision sensors, detect pixel-level brightness changes asynchronously with microsecond precision. Their high temporal resolution, low latency, and wide dynamic range make them ideal for fast motion and challenging lighting. Unlike conventional cameras, they avoid motion blur and transmit only relevant scene changes, reducing redundant data.

Event-based localization has evolved in complexity since the advent of event cameras. [10] adapted semi-direct visual odometry to track 6-DoF motion by combining events and grayscale frames. A fully event-based 6-DoF visual odometry was later introduced in [9]. EVO [12] extended geometric mapping with parallel tracking and mapping. IMU integration has also been explored, with [15]

using an extended Kalman filter and [11] introducing a tightly-coupled visual-inertial odometry pipeline. [13] fused data from standard and event cameras with IMU measurements. EDS [8] was the first direct method to track 6-DoF motion using events and frames. ESVO [14], the first stereo event-based visual odometry, employs a direct geometric approach with interleaved tracking and mapping. Unlike our method, which supports dynamic odometry frequency, ESVO operates at fixed frequencies and requires reinitialization when event input is insufficient. We compare its publicly available implementation with our proposed method.

Early event cameras included standard frames, but modern high-resolution models no longer do, requiring a separate camera for frames. This creates challenges in associating events with corresponding image segments due to different viewpoints and lenses. Consequently, we propose a loosely coupled fusion of event- and frame-based odometry. Building on our previous work [6], which introduced adaptive-frequency stereo event-based odometry, we refine odometry frequency based on estimated camera velocity. We also improve feature matching using multiple event time surfaces and introduce stereo visual odometry that integrates events and frames via pose graph optimization. The proposed odometry is evaluated on self-acquired indoor and outdoor sequences and prove that fusing event and frame-based approaches gives more accurate results than both individual methods.

The remainder of the paper is organized as follows. Section 2 presents our stereo event-based visual odometry, while frame-based odometry is discussed in Section 3.1. Section 3 elaborates event and frame-based odometry fusion. Experimental data and results are presented in Section 4. Lastly, concluding remarks are given in Section 5.

2 Proposed Event Stereo Visual Odometry

In the following sections, we describe our proposed event-based stereo visual odometry method which follows a traditional feature-based odometry framework. The first part is concerned with event time surface generation and selection, followed by feature matching and management. Then we perform pose optimization, and the estimated camera velocity is used in the subsequent odometry step for optical flow prediction, which is important for time surface selection.

2.1 Event Representation

An event camera outputs an asynchronous stream of events. An event $e_k = (\mathbf{x}_k, t_k, p_k)$ at the pixel $\mathbf{x}_k = (x_k, y_k)^\top$ and time t_k is triggered as soon as the increment in brightness, i.e., log photocurrent $L = log(I)$, reaches a defined contrast threshold C:

$$\Delta L(\mathbf{x}_k, t_k) = L(\mathbf{x}_k, t_k) - L(\mathbf{x}_k, t_k - \Delta t_k) = p_k C, \tag{1}$$

where Δt_k is the time since the last event at the given pixel, and $p_k \in \{+1, -1\}$ is the polarity of the brightness change, i.e., increase or decrease in brightness.

Our method does not operate on an event-per-event basis, but requires data representation that contains context and history of events. A widely used data structure for event representation is a simple 2D map containing the timestamp $t_{last}(x, y)$ of last event that occurred at each pixel. In our method, we disregard the event polarities and only use the latest event timestamps for each location.

2.2 Odometry Frequency

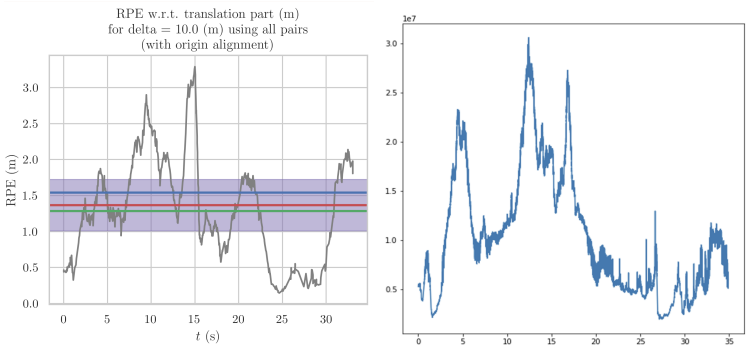


Fig. 1: Left: Relative pose error (translation part) of event-based odometry with the frequency adapted to the number of events. Right: event rate for the same sequence. The spikes in the error coincide with the spikes in the event rate, suggesting that the odometry frequency should not be adapted solely to the event rate.

Existing event-based odometry methods use either a fixed frequency [14] or adapt based on event rate [6]. Fixed-frequency methods require prior dataset analysis and assume a constant rate, while event-based adaptation fails to account for scene structure. Our analysis of the DSEC dataset shows that higher event rates do not always improve accuracy, as error spikes align with event rate spikes (Fig. 1). To address this, we propose adapting odometry frequency based on the camera's translational and rotational velocity. Namely, using the norm of the 3D translational velocity v_t and the norm of the 3D rotational velocity v_r, the odometry frequency f is calculated as follows:

$$f_i = (k\sqrt{v_t^2 + v_r^2} + (1 - k)u_{i-1})/\Delta s, \tag{2}$$

where Δs is a user-defined parameter indicating the displacement that the camera will undergo between consecutive odometry estimates. Namely, setting Δs to 1 would mean that the frequency of odometry would be such that the camera moves for 1 m between estimates during purely translational motion, or that it rotates for 1 rad during pure rotation. We use the parameter k to weight the component u_i derived from current velocity and the previously calculated u_{i-1}. This approach reduces sensitivity to imprecise velocity estimates. To balance computational demands and stability, we set frequency bounds between 30 Hz

and 100 Hz. When running independently, event-based odometry can use any continuous frequency, but for integration with frame-based odometry via pose graph optimization (Section 3), constraining its frequency is more practical.

Given that we have no way of knowing the camera velocity, we propose to perform an exhaustive search of the optimal initial frequency parameter. We pursue pose optimization by experimenting with five different intervals between event frames, ranging from 0.005 s to 0.1 s. The interval that yields the highest number of inliers will then be designated as the initial processing interval, effectively determining the initial odometry frequency.

2.3 Feature Detection and Multisurface Feature Descriptors

We use the Arc* algorithm for event-based corner detection, as described in [1]. The detector works asynchronously, detecting corners for each event by analyzing a circular set of surrounding events. Only detections within a brief time interval relative to the current time surface at t are used. Feature detection is performed on unrectified event coordinates, while stereo-rectified positions are used for matching to ensure alignment along epipolar lines.

To obtain a set of features that are considered for pose optimization, we implement both stereo and temporal feature matching. Feature descriptors are constructed using the data structure commonly reffered to as an *event time surface*. Namely, for each pixel position x we use the latest event timestamp that occurred at the said position to calculate the time surface value at the time $t \geq t_{last}(\mathbf{x})$ using the exponential decay kernel as follows:

$$\tau(\mathbf{x}, t, \delta) = \exp\left(- \frac{t - t_{last}(\mathbf{x})}{\delta} \right). \tag{3}$$

Herein, the parameter δ represents the exponential decay rate. Time surface values can be between 0 and 1. Pixels with more recent events get higher values. The parameter δ is a decay rate that effectively controls the length of the interval which contains past events that are considered for time surface generation. We define feature descriptors as vectors composed of time surface values inside a square $w \times w$ window around the feature.

In previous works that use time surfaces, δ is usually set to some constant value throughout the entire process [14]. However, to achieve the most accurate scene representation, with sharp and distinct edges, this parameter should be adapted to the optical flow of the observed event. We leverage the velocity of the cameras estimated by visual odometry to dynamically determine the values of δ for each timestamp t. As explained in Section 2.2, the estimated camera velocity is also used to determine the frequency of visual odometry estimates. Thus, δ_i, which is used to generate the time surface at the time t_i, is set to the time between the current and the previous pose estimate $\delta_i = t_i - t_{i-1}$. Features are stereo matched between the left and right time surface at the current timestamp t_i, $\tau_L(t_i, \delta_i)$ and $\tau_R(t_i, \delta_i)$, and temporally matched with the previous two time surfaces at the timestamp $t_{i-1} = t - \delta t$; $\tau_L(t_{i-1}, \delta_{i-1})$ and $\tau_R(t_{i-1}, \delta_{i-1})$. Storing

time surfaces from only one previous step makes our method very efficient in terms of memory. Even though the dynamically determined value of δ_i results in sharper time surfaces than by using a constant value, that we can still get suboptimal results for parts of the scene containing events with relatively high or relatively low optical flow values. To address this issue, as time surface windows around the features are used for feature matching, we consider using different δ for each feature based on its potential optical flow, which is be defined as the vector between the current feature position and the position of the feature at the previous timestamp that is considered as a potential match. Given that this approach is computationally heavy, we opted for a compromise solution by considering three different potential values of δ: $\delta_i^1 = 0.5\delta_i$, $\delta_i^2 = \delta_i$, and $\delta_i^3 = 3\delta_i$. The exact value of δ used to generate the feature descriptor is chosen by leveraging the predicted optical flow value for the feature in question. We predict the optical flow $\mathbf{v}_p$ using the previously estimated odometry transformation, also adapting it to the current interval between frames. Then, we choose δ_i^k such that $k = \lfloor ||\mathbf{v}_p||/3 \rceil$, which is experimentally deduced. This enables us to have discriminative descriptors adapted to the amount of displacement of each feature, but without adding significant computational burden, since we can precompute the three event surfaces only once before feature matching and use one of them for every feature.

2.4 Multisurface Feature Matching

We identify correspondences among events detected as corners by employing the zero-normalized cross-correlation (ZNCC) score between their feature descriptors, which are simply vectors composed of time surface values inside a square $w \times w$ window around the feature. The stereo matching phase is done by comparing descriptors of the candidate matches between the left and right time surface at the current timestamp t_i; $\tau_L(t_i, \delta_i)$ and $\tau_R(t_i, \delta_i)$. Given that the position of the detected feature is $\mathbf{x} = (x_i, y_i)$ and the events are stereo rectified, potential stereo matches are located along the stereo epipolar line, i.e., on the same y coordinate. In the temporal matching phase, features are matched between $\tau_L(t_i, \delta_i)$ and $\tau_L(t_{i-1}, \delta_{i-1})$, as well as $\tau_R(t_i, \delta_i)$ and $\tau_R(t_{i-1}, \delta_{i-1})$. By using the previous odometry estimate (if available), we predict the position of each feature detected in $\tau_L(t_{i-1}, \delta_{i-1})$ and $\tau_R(t_{i-1}, \delta_{i-1})$ at the current timestamp t_i, as described in Section 2.3. This prediction is used as a central point of searching for feature matches in $\tau_L(t_i, \delta_i)$ and $\tau_R(t_i, \delta_i)$ inside a square window. Following the identification of the pixel location with the highest matching score, greater precision is achieved through subpixel refinement. Furthermore, to retain only robust features, we establish a predefined threshold, set to $\text{ZNCC}_{\text{min}} = 20$ in our implementation. Feature matches scoring below this threshold are disregarded. Feature matches are considered for pose optimization only if they have traversed all four time surfaces following a circular matching order: $\tau_L(t_i)$, $\tau_R(t_i)$, $\tau_R(t_{i-1})$, $\tau_R(t_{i-1})$, and finally returning to $\tau_L(t)$. If the matching circle closes with the first feature being matched to the same one at the end, it is deemed a reliable feature.

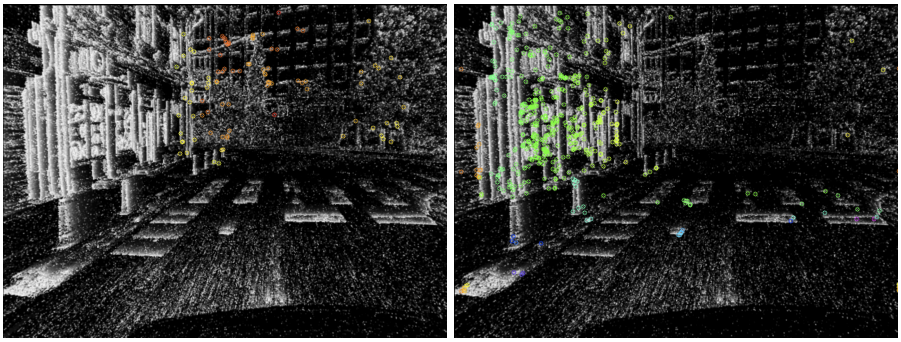


Fig. 2: Time surfaces generated with $\delta = \Delta t$ (left) and $\delta = 0.5\Delta t$ (right) with circles denoting feature matches (color indicates disparity) during forward camera motion. Features with smaller optical flow, found further away from the camera, are matched using the feature descriptors generated using the higher exponential decay rate (δ) of the time surface. On the other hand, features with a larger displacement are matched using the lower value of δ, to avoid motion blur.

Visualization of feature matches found using two different values of δ to generate the time surfaces is shown in Fig. 2.

2.5 Event-based Motion Estimation

The estimation of rotation and translation, representing the camera pose change, is achieved through joint optimization by minimizing reprojection errors. Feature points from the previous view are projected into 3D space using the known camera calibration parameters. 3D points $\mathbf{X} = (x, y, z)^\top$ are then reprojected onto the 2D plane of the current image view, corresponding to the event time surface. The reprojection is expressed as:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \pi(\mathbf{X}; \mathbf{R}, \mathbf{t}) = \begin{bmatrix} f & 0 & c_u \\ 0 & f & c_v \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R} | \mathbf{t} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}. \tag{4}$$

Here, $(u, v)^\top$ are the homogeneous feature coordinates, f is the focal length, and $(c_u, c_v)^\top$ are the image principal point coordinates. The rotation matrix $\mathbf{R}$ and translation vector $\mathbf{t}$ describe the transformation between views.

For stereo projections, we define $\pi^{(l)}$ and $\pi^{(r)}$ for the left and right event time surfaces, and the reprojection error is:

$$\sum_i \left| \mathbf{x}_i^{(l)} - \pi^{(l)}(\mathbf{X}_i; \mathbf{R}, \mathbf{t}) \right|^2 + \left| \mathbf{x}_i^{(r)} - \pi^{(r)}(\mathbf{X}_i; \mathbf{R}, \mathbf{t}) \right|^2. \tag{5}$$

To minimize this error, we use Gauss-Newton optimization and the RANSAC method [4], initially estimating $\mathbf{R}$ and $\mathbf{t}$ 50 times with random feature selections. The best hypothesis is refined with all inliers, defined as points whose reprojection distance is within a threshold.

3 Event and Frame-based Odometry Fusion

3.1 Frame-based Odometry

We propose using standard images alongside events for visual odometry. By independently estimating event-based and frame-based odometries and fusing the results, any frame-based stereo odometry method can be applied. For indoor experiments, we used the *libviso2* library, which minimizes feature reprojection errors and refines velocity estimates using a Kalman filter. For outdoor sequences, we applied the SOFT2 method [2], designed for road vehicles and optimized for non-pure rotation configurations, based on epipolar geometry and kinematics. This method, ranked highest on the KITTI scoreboard [3], was chosen for its suitability in outdoor experiments.

3.2 Pose Graph Construction

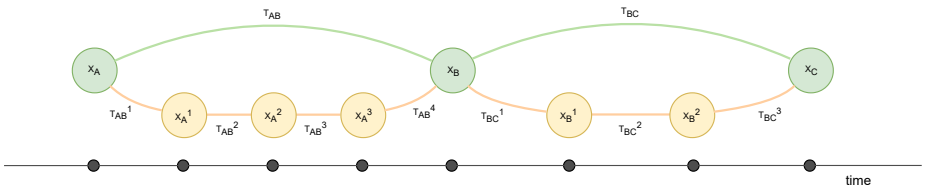


Fig. 3: Pose graph of our proposed visual odometry. Green nodes X_A, X_B and X_C correspond to timestamps where both event-based and frame-based poses are available, whereas yellow nodes are event-based poses. The edges connecting the nodes represent visual odometry transform estimates. The number of the event nodes between frame nodes is dynamically determined.

A pose graph represents poses and their relationships, with nodes corresponding to positions and orientations of sensors over time, and edges representing relative transformations from sensor data like GPS, lidar, or visual odometry. The goal is to optimize poses to best fit sensor constraints, typically using nonlinear least squares optimization.

In our method, each node represents a pose at a specific timestamp (Figure 3). Green nodes correspond to poses with both event and frame-based data, while yellow nodes contain only event-based data, forming "bridges" between frame-based poses. To maintain alignment between event- and frame-based odometry, event-based frequency f_{ev} must be a multiple of frame-based frequency f_{im}, ensuring the final event-based node matches a frame-based timestamp. While this structure maintains consistency, it can reduce accuracy during abrupt velocity changes. To mitigate this, we dynamically adjust f_{ev} within each interval, increasing or decreasing it based on remaining time until the next frame. Once time constraints limit further adjustments, the frequency is fixed. To construct the pose graph and perform pose graph optimization, we use the open-source

C++ library GTSAM[1] that implements smoothing and mapping in robotics and computer vision.

In addition to the transformation constraints within the pose graph, we also need to determine the covariances associated with these transformations. Covariances reflect the reliability of the odometry estimates, allowing us to adjust confidence levels between event-based and frame-based odometry. To quantify confidence, we calculate three covariance values for position and three for orientation, filling the diagonal covariance matrix as follows:

$$\sigma_i = \exp\left(- c_i \frac{N_{inliers}}{N_{max}} \right), \tag{6}$$

where N_{max} is the threshold number of feature matches considered adequate, and c indicates the confidence in the translation or rotation estimation. This approach helps balance the generally higher accuracy of frame-based odometry for rotation, despite event-based odometry having more inliers. We set N_{max} to 100 and 300 for image-based and event-based constraints, respectively, depending on camera and scene characteristics.

4 Experimental Results

We present experimental results across multiple publicly available indoor and outdoor sequences, which are a part of our stereo visual localization dataset featuring event cameras [7][2]. The goal of the experiments is to showcase that fusing event and frame-based odometry estimates can result in a more accurate odometry result than both individual odometries. For this to be possible, we need to identify scenarios where standard cameras suffer from motion blur, whereas event-based data is still reliable. This usually happens during rapid motion or long exposure time due to low lighting. The performance is assessed using the relative pose error metric. Our evaluation leverages the publicly accessible toolbox EVO [5]. Our method has been implemented in C++ within a ROS environment.

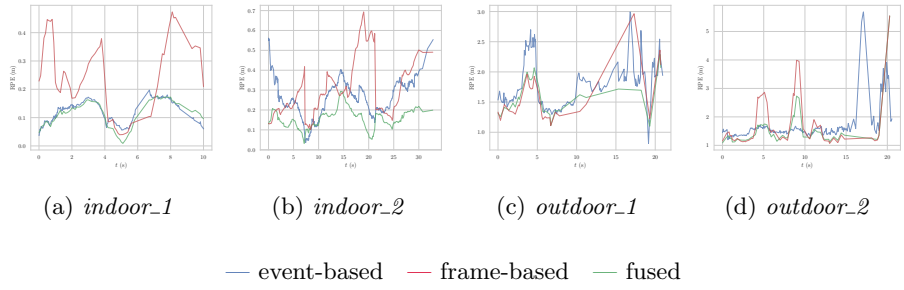
4.1 Indoor Experiments

We evaluate our method on six indoor sequences captured by a handheld rig under daylight streaming through windows. To highlight the advantages of event cameras, we introduce shaking and rapid movements, which induce motion blur in standard frames. Since the frame-based VISO2 odometry relies on features, motion blur significantly reduces its accuracy. Results are presented in Figure 5. The relative pose error, shown in Table 1, is defined as an average relative translation and rotation root-mean-square-error (RMSE) over a sliding window of different lengths (10%, 20%, 30%, 40% and 50% of the total ground truth trajectory length). The results of the proposed fused odometry are more accurate

[1] https://github.com/borglab/gtsam

[2] The dataset is available at http://www.bitbucket.com/unizg-fer-lamor/event-dataset.

compared to both event and frame-based odometry, proving that it exhibits the advantages offered by each information source, which can be seen in the comparison of the relative pose errors in Figure 4a and 4b. Notably, the fused method not only adapts to the more accurate odometry component but at times even achieves higher accuracy than each individual odometry, such as in segments of the *indoor_2* experiment. This occurs primarily in sequences where both event and frame-based odometries exhibit comparable accuracy, indicating that we do not disproportionately favor one over the other during the fusion process. Consequently, the fused odometry estimate functions as a kind of average between the two individual odometries, mitigating errors that may arise if they have drifted in opposite directions. The experiment *indoor_3* was carried out under slightly dimmer lighting conditions, rendering the standard frames even more susceptible to motion blur and less sharp. In comparison to the ESVO event-based method, our approach achieves higher accuracy in both rotation and translation across all sequences.

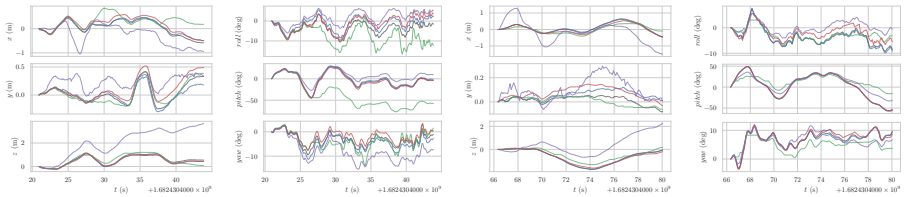


(a) *indoor_1* (b) *indoor_2* (c) *outdoor_1* (d) *outdoor_2*

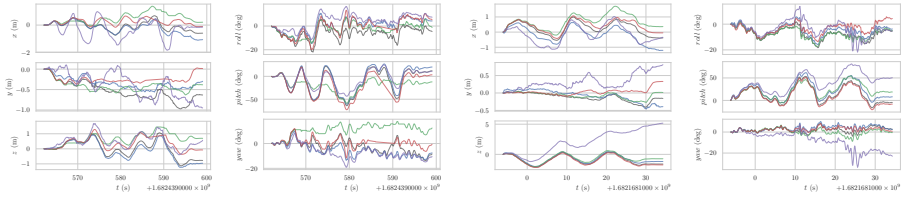
Fig. 4: Relative pose error for event-based, frame-based and fused odometry on indoor (a, b) and outdoor (c, d) sequences.

4.2 Outdoor Experiments

Sequences for our outdoor experiments are recorded while driving a car through the urban city areas, with the sensor rig mounted on the car roof during dusk. Occasional motion blur occurs in the standard frames due to extended exposure times and the velocity at which the car is moving. The standard cameras operate at a frequency of 10 Hz, implying that the displacement between two consecutive frames can be quite substantial. This, coupled with the lighting conditions in the sequences, presents a considerable challenge for frame-based odometry. The accuracy of frame-based odometry is significantly lower on the chosen sequences than on the data with a higher frame rate captured in bright daylight. Translation primarily occurs along the x and z directions, while rotation is predominantly observed in terms of the yaw angle. Results are visualized in Figure 6, whereas the comparison in terms of accuracy is detailed in Table 1. In contrast to the indoor experiments, in this case we calculate the RMSE using the sliding windows



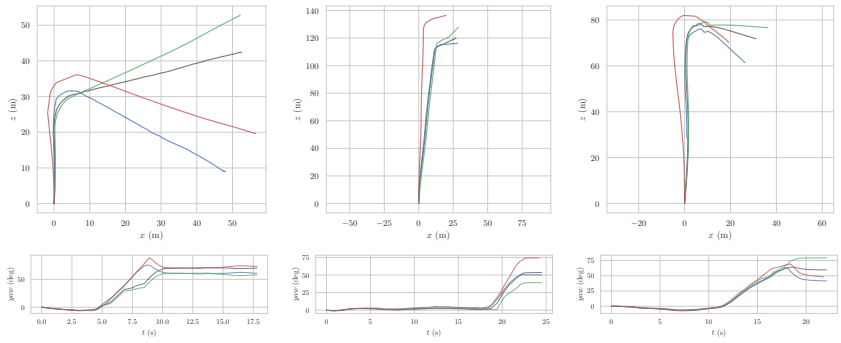
(a) *indoor_1 (t)* (b) *indoor_1 (R)* (c) *indoor_2 (t)* (d) *indoor_2 (R)*



(e) *indoor_3 (t)* (f) *indoor_3 (R)* (g) *indoor_4 (t)* (h) *indoor_4 (R)*

ground truth event-based frame-based fused ESVO

Fig. 5: Comparison of translation and rotation estimates against ground truth on indoor sequences. The figures show raw trajectories and orientations with matching origins.



(a) *outdoor_1* (b) *outdoor_2* (c) *outdoor_3*

ground truth event-based frame-based fused

Fig. 6: Comparison of translation (x and z) and rotation (yaw) estimates against ground truth on three outdoor sequences. The figures show raw trajectories and orientations with matching origins.

of fixed lenghts of 5, 10, 20, 30 and 40 meters. Our event-based odometry and the frame-based SOFT2 odometry exhibit comparable precision in translation

for the chosen sequences. SOFT2 sometimes outperforms in terms of rotational accuracy. Consequently, for frame-based transform estimates incorporated into the pose graph, we opted to assign lower covariance values to the rotational components ($c_i = 2$) compared to their event-based counterparts ($c_i = 1$). As seen in Figures 4c and 4d, the fused odometry adeptly adjusts to the more precise component. This adaptability mitigates spikes in the RPE, indicative of diminished accuracy. Due to intermittent drops in event data resulting from a high event load, the ESVO method encountered challenges in providing viable results, necessitating frequent resets.

Table 1: Translation [%] and rotation [°/m] evaluation results of the proposed visual odometry method compared to ESVO using relative pose error (RMSE).

			Event		Frames		Fused		ESVO	
Data	Time [s]	Length [m]	**t**	**R**	**t**	**R**	**t**	**R**	**t**	**R**
indoor_1	22.9	7.77	7.89	1.55	11.16	11.53	**6.86**	**1.54**	50.93	4.82
indoor_2	13.7	5.07	**9.61**	1.62	22.87	18.76	9.78	**1.47**	85.43	9.17
indoor_3	36.9	11.90	15.03	2.24	17.12	10.67	**13.38**	**1.91**	40.01	5.38
indoor_4	40.4	14.61	8.55	1.27	9.77	3.51	**4.94**	**0.82**	45.59	4.57
indoor_5	27.0	12.45	7.68	**0.95**	25.50	9.83	**6.16**	1.43	39.83	3.14
indoor_6	16.9	5.69	25.17	**3.28**	27.84	33.96	**19.11**	7.13	90.16	16.35
outdoor_1	18.0	94.78	19.55	**0.38**	17.75	0.81	**17.70**	0.68		
outdoor_2	23.6	142.0	21.67	**0.37**	24.59	0.74	**19.54**	0.42		
outdoor_3	21.8	107.58	17.65	**0.57**	17.74	0.67	**16.16**	0.67		
outdoor_4	24.0	215.68	**21.81**	0.23	30.26	**0.07**	26.41	0.09		
outdoor_5	24.0	172.93	17.59	0.28	19.28	**0.08**	**16.12**	0.09		
outdoor_6	23.2	175.06	18.96	0.31	18.26	**0.06**	**15.48**	0.07		

5 Conclusion

This paper presents improvements to our stereo event-based visual odometry, adapting the odometry frequency based on camera velocity instead of using a fixed frequency or event rate. We enhanced feature matching by using multiple event time surfaces for robust feature descriptors. Additionally, we introduced a stereo odometry framework that fuses event and frame data via pose graph optimization. Evaluations on indoor and outdoor sequences show that combining event and frame-based odometries yields more accurate trajectories than using either method individually.

Acknowledgment

This research has been supported by the European Regional Development Fund under the grant PK.1.1.10.0007 (DATACROSS).

References

1. Alzugaray, I., Chli, M.: Asynchronous corner detection and tracking for event cameras in real time. IEEE Robotics and Automation Letters **3**(4), 3177–3184 (2018)
2. Cvišić, I., Marković, I., Petrović, I.: Soft2: stereo visual odometry for road vehicles based on a point-to-epipolar-line metric. IEEE Transactions on Robotics **39**(1), 273–288 (2022)
3. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2012)
4. Geiger, A., Ziegler, J., Stiller, C.: Stereoscan: Dense 3d reconstruction in real-time. In: 2011 IEEE intelligent vehicles symposium (IV). pp. 963–968. Ieee (2011)
5. Grupp, M.: evo: Python package for the evaluation of odometry and slam. `https://github.com/MichaelGrupp/evo` (2017)
6. Hadviger, A., Cvišić, I., Marković, I., Vražić, S., Petrović, I.: Feature-based event stereo visual odometry. In: 2021 European Conference on Mobile Robots (ECMR). pp. 1–6. IEEE (2021)
7. Hadviger, A., Štironja, V.J., Cvišić, I., Marković, I., Vražić, S., Petrović, I.: Stereo visual localization dataset featuring event cameras. In: 2023 European Conference on Mobile Robots (ECMR). pp. 1–6. IEEE (2023)
8. Hidalgo-Carrió, J., Gallego, G., Scaramuzza, D.: Event-aided direct sparse odometry. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5781–5790 (2022)
9. Kim, H., Leutenegger, S., Davison, A.J.: Real-time 3d reconstruction and 6-dof tracking with an event camera. In: European Conference on Computer Vision. pp. 349–364. Springer (2016)
10. Kueng, B., Mueggler, E., Gallego, G., Scaramuzza, D.: Low-latency visual odometry using event-based feature tracks. In: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 16–23. IEEE (2016)
11. Rebecq, H., Horstschaefer, T., Scaramuzza, D.: Real-time visual-inertial odometry for event cameras using keyframe-based nonlinear optimization. In: BMVC (2017)
12. Rebecq, H., Horstschäfer, T., Gallego, G., Scaramuzza, D.: Evo: A geometric approach to event-based 6-dof parallel tracking and mapping in real time. IEEE Robotics and Automation Letters **2**(2), 593–600 (2016)
13. Vidal, A.R., Rebecq, H., Horstschaefer, T., Scaramuzza, D.: Ultimate SLAM? Combining events, images, and IMU for robust visual SLAM in HDR and high speed scenarios. IEEE Robotics and Automation Letters **3**(2), 994–1001 (2018)
14. Zhou, Y., Gallego, G., Shen, S.: Event-based stereo visual odometry. IEEE Transactions on Robotics (2021)
15. Zhu, A.Z., Atanasov, N., Daniilidis, K.: Event-based visual inertial odometry. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5816–5824. IEEE (2017)