End-to-End Learned Event- and Image-based Visual Odometry

Roberto Pellerito,[1] Marco Cannici,[1] Daniel Gehrig,[1] Joris Belhadj,[2] Olivier Dubois-Matra,[2]
Massimo Casasco,[2] Davide Scaramuzza[1]

[1] Robotics and Perception Group, University of Zurich, Switzerland
[2] European Space Agency

***Abstract*— Visual Odometry (VO) is crucial for autonomous robotic navigation, especially in GPS-denied environments like planetary terrains. While standard RGB cameras struggle in low-light or high-speed motion, event-based cameras offer high dynamic range and low latency. However, seamlessly integrating asynchronous event data with synchronous frames remains challenging. We introduce RAMP-VO, the first end-to-end learned event- and image-based VO system. It leverages novel Recurrent, Asynchronous, and Massively Parallel (RAMP) encoders that are $8\times$ faster and 20% more accurate than existing asynchronous encoders. RAMP-VO further employs a novel pose forecasting technique to predict future poses for initialization. Despite being trained only in simulation, RAMP-VO outperforms image- and event-based methods by 52% and 20%, respectively, on traditional, real-world benchmarks as well as newly introduced Apollo and Malapert landing sequences, paving the way for robust and asynchronous VO in space. Code, as well as datasets, will be released upon paper acceptance.**

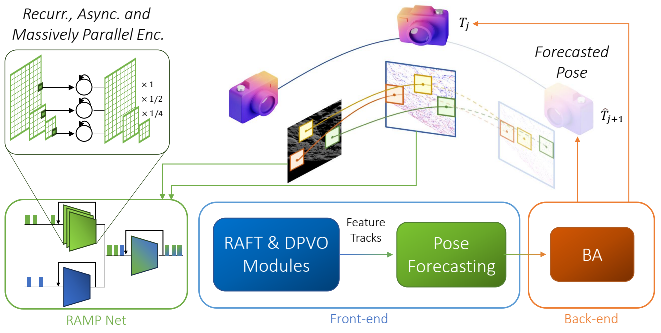


Fig. 1. Overview of the proposed method: Recurrent, Asynchronous, and Massively Parallel Encoders are used to process images and events asynchronously. The resulting encoding is used by DPVO [1] modules to perform data-driven feature tracking and visual odometry. Thanks to the high temporal resolution of the events, we use them to perform pose forecasting, enabling improved pose initialization and early collision prediction.

I. Introduction

Visual Odometry (VO) is a central building block in many robotic and space exploration systems. It provides autonomous navigation capabilities even in GPS-denied environments far from earth-bound satellites. Traditionally, VO systems have relied exclusively on standard RGB cameras to estimate the pose accurately. However, when navigating in challenging scenarios, such as low-light environments, high dynamic range scenes, or low-textured terrains at high speed, standard VO methods typically fail. These shortcomings are mostly caused by well-known limitations inherent in standard cameras, such as their susceptibility to motion blur and limited dynamic range.

Event-based cameras, on the other hand, promise to address all these issues. They are bio-inspired sensors that asynchronously record per-pixel brightness changes at microsecond resolution. They exhibit high dynamic range (HDR), low latency, and low power consumption, making them an ideal complement to regular cameras in VO systems.

Their combination is indeed very promising for critical VO applications where standard sensors such as GPS and LiDAR cannot be used, when navigation involves rapid motion and environments in partial shadow, and in all conditions where Inertial Measurement Units (IMUs) fail due to radiation and temperature variations. These conditions are typical in planetary exploration and landing, making the design of VO systems that fuse images and events extremely promising.

While significant advancements have been made in VO systems fusing event data and images together with IMU [2], [3], the field of monocular VO with only images and events remains mostly unexplored, with only one prior work [4]. These systems are all model-based and typically rely on canonical corner features that can only be reliably extracted in highly textured environments. Dark regions of celestial bodies, like the Moon poles, however, often lack this type of texture making feature extraction and tracking more challenging, and as a result, traditional VO systems less reliable. Data-driven methods like [1], [5] fill this gap by learning to extract visual features end-to-end. Despite the increased robustness in texture-less scenes, these methods still rely on RGB data and, as such, continue to be vulnerable to their failure cases.

While combining data-driven approaches and designing systems that leverage images and events appears promising, effectively combining event data—with its distinctive asynchronicity and sparsity—into synchronous and dense frames is a non-trivial challenge. As a result, traditional learning-based methods usually employ feed-forward architectures and sacrifice asynchronicity and high temporal resolution of events by artificially building synchronized representations at regular intervals. However, in tasks such as VO, this simplification might significantly hinder the algorithm's per-

This work was supported by the European Union's Horizon 2020 Research and Innovation Programme under grant agreement No. 871479 (AERIAL-CORE) and the European Research Council (ERC) under grant agreement No. 864042 (AGILEFLIGHT).

formance and limit its ability to exploit asynchronous events for tracking features.

This work introduces RAMP-VO, the first learning-based VO method fusing events and frames. RAMP-VO leverages RNN-inspired Recurrent, Asynchronous, and Massively Parallel (RAMP) encoders and internal pyramidal memory to fuse images and events by explicitly considering their irregular and asynchronous nature. A motion-aware strategy based on event data is then exploited to extract robust patch-based feature tracks, which are later processed by a differentiable bundle adjustment module [1]. A pose forecasting module extrapolates each patch to future times, enabling future pose prediction at arbitrary timestamps which further exploits the high temporal resolution of the events.

We train RAMP-VO on an event-based version of TartanAir [6]. To address the lack of visual odometry datasets that feature image and event data in challenging space landing settings, we also introduce two novel datasets: the Malapert landing and the Apollo landing datasets which feature challenging motion and lighting conditions due to stark shadows cast by the sun. The first dataset represents a realistic simulation of a spacecraft landing, covering several kilometers of descent near the Malapert crater in the south Moon pole. The second dataset features landings on a 3D scale model of the lunar surface. It is captured with a real RGB camera, an event camera, and precise ground truth camera poses, making it a valuable resource for research and evaluation.

Despite being trained purely in simulation, RAMP outperforms both image-based and event-based methods by 60% and 38% respectively on traditional real-world benchmarks, as well as on the newly introduced Apollo and Malapert landing datasets. To summarize our contributions are:

- A novel massively parallel feature extractor, termed RAMP encoder that fuses images and events, both spatially and temporally. It is 7.9 times faster and achieves a 22% higher performance than state-of-the-art asynchronous encoders.
- A new pose forecasting technique, that extrapolates feature tracks and uses them to predict the camera pose in the future, for improved pose initialization, leveraging the high temporal resolution of the events.
- Two novel datasets, Apollo and Malapert landing, targeting challenging planetary landing scenarios.

II. Related Work

Event-based visual odometry. While pixel-level tasks are mostly dominated by learned approaches, all the existing event-based VO methods are model-based [7]. These are usually distinguished as indirect or direct based on whether they directly process raw pixels, or use intermediate representations. Early direct methods [8], [9] are based on scene reconstruction from events, while newer ones [4] propose to also incorporate frames by minimizing a photo-metric loss based on the Event Generation Model [10] (EGM). In contrast, indirect approaches build upon event-based feature extractors [11], [12], [13], [14] and trackers [15]. Some of them [16], [17], [2] extract features from frames and track them in the blind time with events. Others [3], motion-compensate events and leverage additional sensors like the IMU.

Despite reducing errors from geometry estimation or the EGM, these methods still suffer in low-texture environments as they typically rely on edge-like features. Similar robustness issues are also typical in classical image-based VO, as they still heavily rely on hand-crafted features [18], [19], [20]. In and attempt to extend the operation domain of VO systems to these environments, new data-driven approaches emerged [5], [1], which extract deep features and track them using RAFT-inspired [21] refinement modules. Our work builds upon this recent trend and proposes the first learning-based event-based VO system. Compared to other model-based VO, we demonstrate improved performance and increased robustness in challenging scenarios.

Fusing events and frames. While a great variety of approaches leverage, optimize, or fuse images and events for different downstream tasks [22], [23], [24], [25], the topic of effectively fusing the two data modalities while considering their different nature has thus far been underexplored. Some methods [22], [26], [23] synchronize and concatenate both modalities and process them together with a shared encoder. Others [27], [28], instead, use specialized feature extractors, but still resort to data synchronization for processing.

To date, only one prior study, RAM Net [29], has proposed a specialized and asynchronous way of fusing events and frames. However, RAM Net relies on a sequential hierarchical feature extraction process and utilizes slow Conv-GRU modules. Our asynchronous encoders build upon RAM Net but exploit pixel-wise operations and parallel extraction of multi-scale features, demonstrating both higher performance as well as improved efficiency.

III. Methodology

A. Overview

Our end-to-end event- and frame-based visual odometry algorithm, RAMP-VO, takes inspiration from recurrent asynchronous multimodal (RAM) networks [29] and deep patch visual odometry (DPVO) [1]. The RAMP encoder processes events and frames as a temporally ordered, asynchronous stream of data, which we will call frames. Similar to DPVO, for each new frame with index j the following processing steps are performed, found in more detail in [1]: First, matching m_j and context features c_j are computed from frame j through a sequence of feature extractors, then N patches with dimension $p \times p$ are extracted from these feature maps. While DPVO opts for a random sampling strategy, we instead extract corners where there is a high event density and apply non-maximum suppression to apply equal spacing. Without this measure, many corners would fall in regions with few events. Denote a generic patch $\mathbf{P}_j^l$ with index l extracted in frame j as

$$\mathbf{P}_j^l = \begin{bmatrix} \mathbf{x} & \mathbf{y} & \mathbf{1} & \mathbf{d} \end{bmatrix}^T \quad \mathbf{x}, \mathbf{y}, \mathbf{d} \in \mathbb{R}^{1 \times p^2}. \tag{1}$$

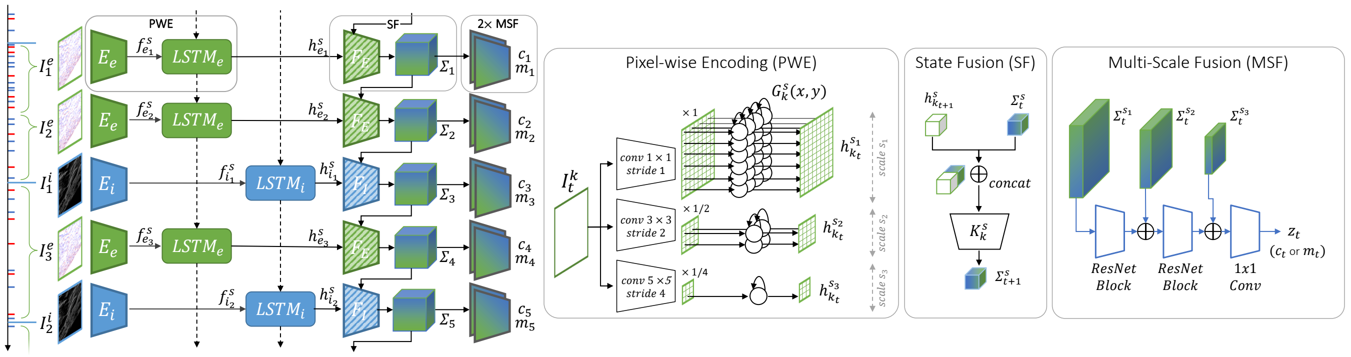


Fig. 2. An overview of the proposed RAMP Net encoder. Events and images are first asynchronously processed by two parallel pixel-wise, multi-scale, encoding branches (PWE) made of a set of convolutional layers followed by pixel-wise LSTMs G^s_k. Features coming from different data modalities k are then fused (SF), at each scale, into a shared state Σ^s_t by employing sensor-specific encoders. The multi-scale features are then finally combined through two separate fusion modules (MSF) to produce the matching and context features, m_t and c_t.

Note that, as in DPVO, we model patches as collections of contiguous pixels, and not only as single points. Here $\mathbf{d}$ is the patch depth (constant for all pixels within the patch), and $\mathbf{x}, \mathbf{y}$ are the coordinates of the pixels in the patch.

These patches are then projected into the *previous frames* of index $i \in \{j-r, j-r+1, ..., j\}$ and patches extracted in previous frames (going back to frame $j-r$) are projected into the current frame j. A visualization is provided in Figure 3. Let the projection of patch P^l_j, into frame i be

$$\mathbf{P}^{l\prime}_{ji} \sim \mathbf{K}\mathbf{T}_i\mathbf{T}_j^{-1}\mathbf{K}^{-1}\mathbf{P}^l_j \tag{2}$$

where K is the 4×4 camera matrix and T_i, T_j are poses at frames i, j. We summarize this as $\mathbf{P}^{l\prime}_{ji} = \omega(T_i, T_j, \mathbf{P}^l_j)$.

Next, RAMP-VO computes camera motion by estimating corrections $\Delta_{li} \in \mathbb{R}^2$ for each projected patch $\mathbf{P}'_{li}$, as well as importance values $\sigma_{li} \in \mathbb{R}^{2\times2}$ through a series of blocks involving a correlation lookup, 1D Convolution, Soft Aggregation, Transition Block and Factor Head. Since these operations are out of the scope of the current work, we summarize them with the following relation

$$\Delta_{li}, \sigma_{li} = F(\mathbf{P}^{l\prime}_{ji}, c_i, m_i, m_j) \tag{3}$$

where we compare context features c_i with matching features extracted in frame i and j.

Finally, leveraging the position corrections and their weights, RAMP-VO performs a differentiable bundle adjustment (BA) step, which optimizes the camera poses and depths of each patch to minimize the projection error:

$$\sum_{(l,i)\in\mathscr{E}} \left\| \hat{\mathbf{P}}^{l\prime}_{ji} + \Delta_{li} - \hat{\omega}(T_i, T_j, \mathbf{P}^l_j) \right\|^2_{\sigma_{li}}. \tag{4}$$

The ˆ operation indicates that the center pixel is used, and $\mathscr{E}$ is a bipartite graph that connects frames with patches that are projected into that frame. By using two Levenberg-Marquart optimization steps, the BA layer updates the depth and camera poses in the time window. This operation is fully differentiable, and thus used during training to backpropagate errors. The main innovations of this work are related to how the context and matching features c_j and m_j are generated, and how the BA layer is used to perform *pose forecasting*. These topics are discussed next.

B. Asynchronous and Massively Parallel Encoders

Denote the stream of data $\{x_{k_j}(t_j)\}^T_{j=1}$ captured at timestamps t_j. Here x (henceforth denoted as ”frame”) denotes either a $5\times H\times W$ sized event stack [30], in the case of events, or a $C\times H\times W$ sized image[1]. The variable $k_j \in \{\text{e},\text{i}\}$ denotes the sensor at timestamp t_j.

We encode these data structures using a Recurrent, Asynchronous and Massively Parallel (RAMP) encoder. An overview of the architecture is provided in Figure 2. We employ two different Multi-Scale Fusion modules, with identical structures, to generate either the context or the matching feature maps.

The RAMP encoders first transduce the data stream using pixel-wise encoders starting with a stream of features $\{f^s_{k_j}\}^T_{j=1}$ at three different scales. To encode this data we use separate sensor-specific encoders, one for each scale

$$f^s_{k_j}(t_j) = E^s_{k_j}(x_{k_j}(t_j)) \tag{5}$$

These encoders have a kernel size 1×1, 3×3, and 5×5 respectively, and a stride of 2^s with $s = 0, 1, 2$.

This stream of features is then further encoded in two recurrent stages: (1) intra-sensor fusion, and (2) inter-sensor fusion. In the intra-sensor fusion step, we process features originating from a single sensor and scale with an LSTM operating on individual pixels, inspired by [31] and by the recurrent connections in [32]

$$h^s_{k_j}(t_j) = G_{k_j}(f^s_{k_j}(t_j)) \tag{6}$$

where we have omitted the cell state for the sake of clarity.

The inter-sensor fusion step, instead fuses the hidden states from separate sensors asynchronously using only a single depth-wise convolution in the following way:

$$\Sigma^s_j = H_{k_j}\left([h^s_{k_j}(t_j)\|\Sigma_{j-1}]\right) \tag{7}$$

[1]For color images $C = 3$ and for gray-scale images $C = 1$.

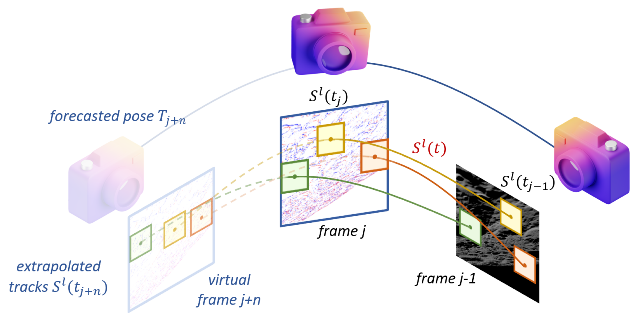


Fig. 3. Illustration of pose forecasting. Through patch extraction, and projection into future frames we construct feature tracks for frames $j, j-1, \ldots$ which we use to construct the splines $S^l(t_j)$. To perform pose forcasting, we extrapolate the feature tracks to time t_{j+n}, and apply bundle adjustment to solve for the forecasted pose T_{j+n}.

where $\|$ denotes concatenation. A different depth-wise convolution H_{k_j} is used each time depending on which sensor the data being fused is generated from.

Note that the RAMP encoder fuses each pixel independently and only relies on a simple inter-sensor fusion strategy, and thus can be done efficiently in parallel. This is in contrast to the encoder in [33], which requires sequential processing, and expensive ConvGRU operations at two stages. After generating inter-sensor fused states Σ_j at each time t_j, a hierarchical encoder is used to generate either the context or the matching feature maps, each at $1/4$ the original resolution

$$z_j = K(\{\Sigma_j^s\}_{s=0}^2). \tag{8}$$

Here z_j can be either c_j or m_j, *i.e.* context and matching features, required by the DPVO backend.

Multi-Scale Fusion. The Multi-scale Fusion (MSF) module follows the feature extractors in [1] and consists of a 7×7 convolution with stride 2, four residual blocks (two at $1/2$ and two at $1/4$ of the initial resolution), and a final 1×1 convolution. We use the same configuration in [1], providing full-scale features Σ_j^0 as input to the encoder, but injecting Σ_j^1 and Σ_j^2 after the second and fourth residual blocks respectively. We do so by concatenating these features with the residual block's outputs and adjusting the channels of the next operation to accommodate the additional features.

C. Pose Forecasting

On top of the patch extraction and pose optimization steps described in Section III-A, RAMP-VO incorporates a novel pose forecasting module to bootstrap future camera poses.

After processing a frame k, the set of available patches comprises patches $\mathbf{P}_k^l$ extracted at the latest time t_k, paired with their projections $\mathbf{P}'^l_{ki}$ in previous frames i, together with patches extracted at previous instants t_j combined with their projections $\mathbf{P}'^l_{jw}$ in future and past frames t_w. Each of these sequences can thus be used to fit a continuous-time model of the patch movement along time.

We do so by fitting two cubic univariate splines $S_x^l(t)$ and $S_y^l(t)$, each modeling the motion of the patch center along the x and y axis of the image plane, such that $(\mathbf{x}_i^l, \mathbf{y}_i^l) \sim \mathbf{S}^l(t_i) = (S_x^l(t_i), S_y^l(t_i))$. Given this model, we can then extrapolate the position where the patch will be tracked in the future timestamp t_{k+1} by evaluating the splines in t_{k+1} and assuming the first and second derivatives to be zero at that point.

We consider patches extracted from the last 11 frames, together with their projections, and extrapolate their future locations at t_{k+n}, assuming the depth of these patches to remain constant. We then extrapolate the location of the camera at time t_{k+n} by optimizing its 6 DOF pose through Bundle Adjustment, according to the forecasted patch tracks. In particular, we solve

$$\sum_{(l,i)\in\mathscr{E}} \left\| \mathbf{S}^l(t_{k+n}) - \hat{\omega}(T_i, T_j, \mathbf{P}_j^l) \right\|_{\sigma_{li}}^2. \tag{9}$$

IV. Experiments

Training. We train RAMP-VO on the synthetic TartanAir [**?**] dataset, which we augment with events using VID2E [34]. Given a training sequence, we follow the DPVO [1] training scheme to select and filter frames for training. Additionally, we also enforce a minimum of 1.2M events between every pair of frames and remove additional frames if this condition is not satisfied. Finally, we feed RAMP-VO with both events and frames by interleaving two event stacks for every pair of selected frames. We create the first event stack by stacking the $600,000$ events received just before the mid-timestamp between the pair of images, and the second event stack by aggregating the $600,000$ events preceding the second frame. Since ground truth poses are only given for frames, we do not compute a loss for the mid-frame events.

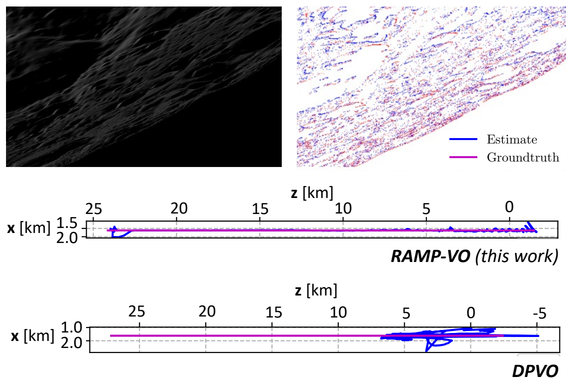
We train RAMP-VO for $350,000$ steps with sequences of 15 images and 30 event stacks each on a Quadro RTX 8000 GPU. The remaining hyperparameters are set as in DPVO [1]. Full training takes around 8 days on our hardware.

Datasets. We use the Stereo DAVIS [35] dataset to compare our method against the state-of-the-art. Data provided in [35] was recorded indoors with a hand-held stereo DAVIS240C, with poses provided by a motion caption system.

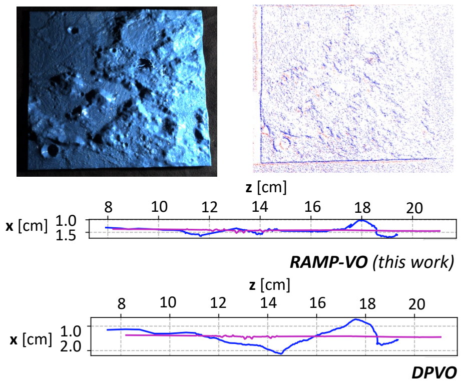
Moreover, since datasets acquired under challenging motion and lighting are scarce in the existing literature, we also propose two additional benchmarks. These datasets feature landing on the Moon's surface, a scenario involving fast motion, high-dynamic range, and untextured terrains.

Malapert landing. Malapert landing consists of 20 minutes of data, divided into 2 sequences from a simulation of the Malapert south Moon region. We use the planets and satellites simulator PANGU [2] to generate realistic descent trajectories, each 250 km long in altitude and 40 km in translation, featuring partial or complete darkness. Ground truth poses of the spacecraft's center of mass are provided at 5Hz, together with 640×480 synchronized RGB images. We generate synthetic events using Vid2E [36] with default settings.

[2]https://pangu.software/



(a) Malapert dataset



(b) Apollo dataset

Fig. 4. Preview, and qualitative trajectory comparison on Malapert dataset (a), and Apollo dataset (b). Note that while the Malapert sequence is measured in kilometers, the Apollo sequence, recorded at a miniature scale of the Moon's surface, is in centimeters.

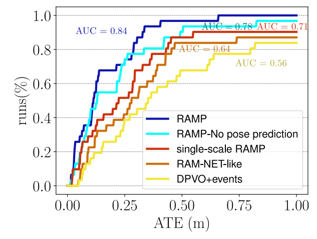
Apollo Landing. Apollo landing is a real dataset consisting of a total of 5 minutes of recording, split into 6 trajectories, featuring both vertical and lateral descent trajectories on a 260×260 cm scale replica of the Apollo 17 landing site. Frames and events are recorded with a beam-splitter setup featuring a 20Hz FLIR BFS-U3-12263c RGB camera and a Prophesee Gen4 event camera, similar to the one used in [4]. Poses are recorded with an OptiTrack motion capture system. We downsample frames and events to a resolution of 640×480 before processing.

Baselines. We evaluate the proposed RAMP-VO architecture against several VO state-of-the-art methods making use of images only (I), events only (E) as well as based on both images and events (I+E). We follow the evaluation in [4] and select ORB-SLAM2 [19] and DPVO [1] as image-only baselines, while we use EDS [4] for comparison against methods fusing images and events, being the only VO system of this kind in the literature. Since we are the first to propose an end-to-end learnable VO system for event cameras, we also implement an event-only DPVO baseline, EDPVO, that directly processes event stacks, as well as one that processes images and events concatenated together, DPVO+events.

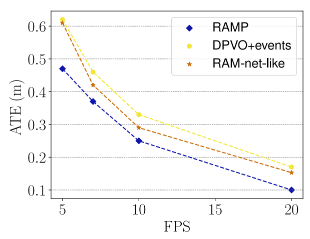
A. Effects of RAMP blocks

We start by analyzing the individual contribution of RAMP-VO modules on the TartanAir [?] dataset. In this section, we adopt the evaluation protocol in [1], which analyzes the percentage of sequences from the TartanAir test set below a given absolute trajectory error threshold, producing plots like the one in Fig. 5. This is done to discount the effect of individual diverging sequences, which report abnormally high trajectory errors, and would skew the average. We summarize the results by computing the the area under the curve (AUC), and use it to compare between ablations.

Results: From the compared methods, DPVO+events has the lowest performance with an AUC of 0.56, followed by the baseline with a RAM-Net-like encoder with 0.64. Our single-scale RAMP-VO has an AUC of 0.71, a 11% increase over the RAM-Net encoder, despite RAM-Net using multiple scales. Transitioning to the multi-scale variant of RAMP-VO, we achieve an AUC of 0.78, which improves by 22% on the RAM-Net encoder. Finally, we analyze the impact of using pose forecasting for initializing the camera pose before performing bundle adjustment. Adding pose forecasting improves the AUC to 0.84.



(a) Ablation Study



(b) Low Framerate VO

Fig. 5. Comparisons of the ablated models on the full TartanAir test set (a). We show here the importance of using a RAMP encoder as in RAMP-VO over a sequential, single-scale encoder and feed-forward encoder. The RAMP encoder is better at maintaining memory than the RAM-Net-like encoder, as highlighted in low framerate VO experiments (b) on the *carwelding* sequences of TartanAir

Timing Results: To further motivate the use of the proposed Asynchronous and Massively Parallel encoders, as opposed to the hierarchical and sequential RAM Net encoders [29], we now assess their performance in terms of processing speed. We time the two encoders on the ablation test set by recording the time required to extract features Σ_j from a single frame (or event), averaged over the full dataset, on a Quadro RTX 8000 GPU. RAM Net takes 370 ms on average, while the proposed RAMP Net only takes 47 ms, resulting in a $7.9\times$ speedup. By processing features pixel-wise, our encoder can indeed exploit much higher parallelism than RAM Net.

B. Low Framerate VO

By using events, our VO system can rely only on low framerate images, and predict poses in the blind-time between frames. To test this capability, we design an experiment on the *carwelding hard p003* and *carwelding easy p007* sequences of the TartanAir test set, with images and event

TABLE I
AVERAGE ABSOLUTE TRAJECTORY ERROR ON THE MALAPERT AND APOLLO DATASETS

	Input	Malapert [km]		Apollo [cm]		
		cam-1	cam-2	rec-1	rec-3	rec-4
DPVO	I	73,1	48,2	0,9	0,3	1,3
DPVO	I+E	40,1	34,0	0,9	0,2	1,1
RAMP-VO SS (Ours)	I+E	1,1	9,4	**0,7**	0,2	1,0
RAMP-VO (Ours)	I+E	**0,6**	**4,3**	0,8	**0,2**	**0,9**

TABLE II
AVERAGE ABSOLUTE TRAJECTORY ERROR (M) ON STEREO DAVIS

	Input	Bin	Boxes	Desk	Monitor
ORB-SLAM2 [19]	I	**2.5**	7.0	9.3	10.3
DPVO [1]	I	3.6	6.7	7.0	11.5
DPVO [1]	E	5.8	7.2	7.8	14.4
EDS [4]	I+E	2.6	5.8	5.0	8.0
DPVO [1]	I+E	4.2	5.3	4.9	6.2
RAMP-VO (Ours)	I+E	3.1	**5.1**	**3.1**	**4.2**

stacks arriving at 20 Hz, where we artificially reduce the framerate of the images by subsampling them. We then evaluate the trajectory error for DPVO+events, RAMP-VO, and RAMP-VO with the RAM-Net-like encoder on 20 Hz ground truth poses, and report the results in Fig. 5 (b).

As can be seen, compared to DPVO+events, the RAM-Net-like and RAMP encoder achieve a lower absolute trajectory error, especially at lower framerates. This is because both methods propagate information asynchronously through recurrency throughout the deadtime between two frames, increasing the stability of the VO system. However, we see that the RAMP encoder consistently outperforms the RAM-Net-like encoder, demonstrating its superiority.

C. Results on Space Data

Next, we validate RAMP-VO on the low light and low frame frames Malapert and Apollo landing datasets, where we also demonstrate our model's generalizability. We compare both single- and multi-scale architectures against DPVO and DPVO+events, and report the average absolute trajectory error over 5 runs.

Results: In Table I, we start by analyzing the performance on the challenging Malapert landing dataset. RAMP-VO is able to recover accurate poses that deviate only 0.2% to 1.7% from the ground truth poses, if we consider that trajectories cover 250km in distance. DPVO, instead, can not capture a valid trajectory leading to ATE errors of several kilometers, equivalent to 20% to 30%. When events are added, DPVO decreases the error by 45.14% to 29.46%, highlighting the importance of events in dark regions. We report a qualitative comparison on a Malapert sample in Figure 4

On the Apollo landing dataset, both RAMP-VO versions (multi- and single-scale, or SS) outperform image and image+event DPVO baselines by up to 30.77% reaching an error from 2% to 6% compared to the ground truth. Contrary to Malapert, Apollo scenes feature clear frames and events, which explains the lower performance improvement given by the events. Similarly, single-scale and multi-scale RAMP-VO achieve similar results on Apollo, differently from Malapert where errors are around two times higher for single-scale. This suggests that when information is scarce, like in Malapert low-light environments, having the ability to focus on both global and fine-grained details improves robustness.

D. Comparison with State of the Art

We conclude by analyzing the performance of the proposed RAMP-VO architecture against state-of-the-art methods on the Stereo DAVIS [35] dataset. Results are reported in Table II. This benchmark represents a completely new scenario compared to the TartanAir training setting, as Stereo DAVIS features gray-scale frames, a lower 180×240 resolution, and articulate motion different from the large and straight movements of TartanAir. Similar to the Apollo landing dataset, RAMP-VO is thus required to generalize from simulated to real sensor data. We compare our architecture following the benchmark in [4], processing each time a $100,000$ event stacks at the same rate as frames.

Results: As showed in Table II, the proposed RAMP-VO outperforms all other baselines, both image- and event-based, with the exception of only one sequence where ORB-SLAM2 [19] performs better. It does this while being trained entirely on synthetic images and events. Notably, RAMP-VO consistently surpasses the DPVO image-only baseline and achieves better performance than EDS, which is thus far the sole prior event- and frame-based VO. This performance improvement holds true especially on challenging sequences. Indeed, on sequences such as Desk and Monitor, RAMP-VO outperformes EDGS by a factor of 2-3.

It is worth noting how effective processing and fusion of event data is particularly important to achieve high performance on this benchmark. Indeed, naive adaptations of DPVO for event processing fall short, while specialized event fusion techniques, like ours, can achieve better generalization and transfer to real-world data.

V. CONCLUSION

In this work, we introduce RAMP-VO, an end-to-end VO system tailored for challenging environments such as those encountered during lunar descents. RAMP-VO exploits asynchronous and massively parallel encoders to efficiently and effectively fuse asynchronous event data into synchronous frames, achieving a $7.9\times$ speedup and 22% improvement over state-of-the-art asynchronous encoders. Moreover, by incorporating pose forecasting and events, RAMP-VO reduces the trajectory error of both existing deep-learning-based solutions by up to 52%, as well as model-based VO methods fusing images and events by up to 20%. Experiments show that RAMP-VO can transfer zero-shot from training only performed on synthetic data, while still outperforming the other baselines. This work represents a new milestone in event data fusion for VO, and we believe it can spark new interest in the use of event cameras and learning-based approaches for robust navigation.

References

[1] Z. Teed, L. Lipson, and J. Deng, "Deep patch visual odometry," *arXiv preprint arXiv:2208.04726*, 2022.

[2] F. Mahlknecht, D. Gehrig, J. Nash, F. M. Rockenbauer, B. Morrell, J. Delaune, and D. Scaramuzza, "Exploring event camera-based odometry for planetary robots," pp. 8651–8658, 2022.

[3] A. R. Vidal, H. Rebecq, T. Horstschaefer, and D. Scaramuzza, "Hybrid, frame and event based visual inertial odometry for robust, autonomous navigation of quadrotors," *CoRR*, vol. abs/1709.06310, 2017. [Online]. Available: http://arxiv.org/abs/1709.06310

[4] J. Hidalgo-Carri'o, G. G. Bonet, and D. Scaramuzza, "Event-aided direct sparse odometry," *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5771–5780, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:248227281

[5] Z. Teed and J. Deng, "Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras," *Advances in neural information processing systems*, vol. 34, pp. 16 558–16 569, 2021.

[6] W. Wang, D. Zhu, X. Wang, Y. Hu, Y. Qiu, C. Wang, Y. Hu, A. Kapoor, and S. Scherer, "Tartanair: A dataset to push the limits of visual slam," IEEE, pp. 4909–4916, 2020.

[7] 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 T-PAMI.*, 2020.

[8] H. Kim, S. Leutenegger, and A. J. Davison, "Real-time 3d reconstruction and 6-dof tracking with an event camera," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VI 14*. Springer, 2016, pp. 349–364.

[9] 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*, vol. 2, pp. 593–600, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:16588072

[10] G. Gallego, C. Forster, E. Mueggler, and D. Scaramuzza, "Event-based camera pose tracking using a generative event model," 2015, arXiv:1510.01972.

[11] I. Alzugaray and M. Chli, "Asynchronous corner detection and tracking for event cameras in real time," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3177–3184, 2018.

[12] E. Mueggler, C. Bartolozzi, and D. Scaramuzza, "Fast event-based corner detection," in *British Machine Vision Conference (BMVC)*, 2017.

[13] V. Vasco, A. Glover, and C. Bartolozzi, "Fast event-based harris corner detection exploiting the advantages of event-driven cameras," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 4144–4149.

[14] P. Chiberre, E. Perot, A. Sironi, and V. Lepetit, "Detecting stable keypoints from events through image gradient prediction," *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 1387–1394, 2021.

[15] D. Gehrig, H. Rebecq, G. Gallego, and D. Scaramuzza, "Asynchronous, photometric feature tracking using events and frames," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 750–765.

[16] B. Kueng, E. Mueggler, G. Gallego, and D. Scaramuzza, "Low-latency visual odometry using event-based feature tracks," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 16–23.

[17] A. Zihao Zhu, N. Atanasov, and K. Daniilidis, "Event-based visual inertial odometry," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5391–5399.

[18] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE TRO*, vol. 31, no. 5, pp. 1147–1163, 2015.

[19] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," *IEEE TRO*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.

[20] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 3, pp. 611–625, 2017.

[21] Z. Teed and J. Deng, "Raft: Recurrent all-pairs field transforms for optical flow," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*. Springer, 2020, pp. 402–419.

[22] I. Alonso and A. C. Murillo, "EV-SegNet: Semantic segmentation for event-based cameras," in *CVPRW*, 2019.

[23] S. Tulyakov, D. Gehrig, S. Georgoulis, J. Erbach, M. Gehrig, Y. Li, and D. Scaramuzza, "Time lens: Event-based video frame interpolation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 16 155–16 164.

[24] S. Tulyakov, A. Bochicchio, D. Gehrig, S. Georgoulis, Y. Li, and D. Scaramuzza, "Time lens++: Event-based frame interpolation with parametric non-linear flow and multi-scale fusion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 17 755–17 764.

[25] L. Sun, C. Sakaridis, J. Liang, Q. Jiang, K. Yang, P. Sun, Y. Ye, K. Wang, and L. V. Gool, "Event-based fusion for motion deblurring with cross-modal attention," in *European Conference on Computer Vision*. Springer, 2022, pp. 412–428.

[26] Y. Hu, J. Binas, D. Neil, S.-C. Liu, and T. Delbruck, "Ddd20 end-to-end event camera driving dataset: Fusing frames and events with deep learning for improved steering prediction," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2020, pp. 1–6.

[27] S. Tulyakov, A. Bochicchio, D. Gehrig, S. Georgoulis, Y. Li, and D. Scaramuzza, "Time lens++: Event-based frame interpolation with parametric non-linear flow and multi-scale fusion," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[28] N. Messikommer, C. Fang, M. Gehrig, and D. Scaramuzza, "Data-driven feature tracking for event cameras," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 5642–5651.

[29] D. Gehrig, M. Rüegg, M. Gehrig, J. H. Carrio, and D. Scaramuzza, "Combining events and frames using recurrent asynchronous multi-modal networks for monocular depth prediction," 2021.

[30] S. Mostafavi I., L. Wang, Y.-S. Ho, and K.-J. Y. Yoon, "Event-based high dynamic range image and very high frame rate video generation using conditional generative adversarial networks," in *Conference of Computer Vision and Pattern Recognition (CVPR)*, 2019.

[31] M. Cannici, M. Ciccone, A. Romanoni, and M. Matteucci, "A differentiable recurrent surface for asynchronous event-based data," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*. Springer, 2020, pp. 136–152.

[32] M. Gehrig and D. Scaramuzza, "Recurrent vision transformers for object detection with event cameras," in *Conference of Computer Vision and Pattern Recognition (CVPR)*, 2023.

[33] M. Gehrig, W. Aarents, D. Gehrig, and D. Scaramuzza, "Dsec: A stereo event camera dataset for driving scenarios," in *IEEE RA-L*.

[34] D. Gehrig, M. Gehrig, J. Hidalgo-Carrió, and D. Scaramuzza, "Video to events: Recycling video datasets for event cameras," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, June 2020.

[35] Y. Zhou, G. Gallego, H. Rebecq, L. Kneip, H. Li, and D. Scaramuzza, "Semi-dense 3D reconstruction with a stereo event camera," in *European Conference of Computer Vision (ECCV)*, 2018, pp. 242–258.

[36] D. Gehrig, M. Gehrig, J. Hidalgo-Carrió, and D. Scaramuzza, "Video to events: Recycling video datasets for event cameras," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, June 2020.