BIT-VO: Visual Odometry at 300 FPS using Binary Features from the Focal Plane

Riku Murai[1], Sajad Saeedi[2], Paul H. J. Kelly[1]

***Abstract*— Focal-plane Sensor-processor (FPSP) is a next-generation camera technology which enables every pixel on the sensor chip to perform computation in parallel, on the focal plane where the light intensity is captured. SCAMP-5 is a general-purpose FPSP used in this work and it carries out computations in the analog domain before analog to digital conversion. By extracting features from the image on the focal plane, data which is digitised and transferred is reduced. As a consequence, SCAMP-5 offers a high frame rate while maintaining low energy consumption. Here, we present BIT-VO, which is the first 6-Degrees of Freedom visual odometry algorithm which utilises the FPSP. Our entire system operates at 300 FPS in a natural environment, using binary edges and corner features detected by the SCAMP-5.**

I. INTRODUCTION

Vision-based pose estimation algorithms, such as Visual Odometry (VO) and Visual Simultaneous Localisation and Mapping (VSLAM), benefit from higher frame rates; the two main benefits are a reduction in motion blur, and – with smaller frame-to-frame motion – a faster optimisation convergence [1]. However, state of the art algorithms operates at 30-80 frames per second (FPS), as increasing the frame rate increases the volume of data to be processed. Even for fast VOs such as SVO [2], [3], the author recommends a camera which operates at 40-80 FPS. Thus, to reduce the effect of motion blur caused by rapid camera movements, complex algorithms are required. DTAM [4], which creates a dense 3D map, is one such example.

This paper looks at the problem from a different perspective; if we can reduce the volume of data coming from the image sensor, we then have more time for pose estimation. Feature-based VO/VSLAM, such as PTAM [5] and ORB-SLAM [6] compute a sparse set of features from an image and operate using them. Unfortunately, feature extraction is often computationally expensive (ORB-SLAM requires 11ms for 1000 corners), and prevents increased frame rate. The problem is that images are first transferred, and then the features are extracted. Instead, is there a way to stream just the relevant features from the image sensor?

Focal-plane Sensor-processor (FPSP) [7] is a general-purpose vision chip technology which allows user-defined computation in a highly parallel manner on the focal plane of the sensor at high frame rates. For instance, SCAMP-5 can perform: High Dynamic Range Tone Mapping [8], Depth from Focus [9], and FAST Keypoint Detection [10] on the focal plane. The low energy, high frame rate nature of the FPSP, consuming only 1.23W even when operating at its maximum effective frame rate of 100,000 FPS [11], makes the device appealing for high-speed operations. The key to the efficiency of FPSPs – in terms of both power consumption and frame rate – is the ability to reduce the amount of data transferred. As opposed to traditional camera sensors, FPSPs can perform image processing early in the pipeline to deliver a reduced volume of data to later stages – in this paper, just binarised corners and edges. This reduces both bandwidth and energy consumption.

Similar to FPSPs, event-based cameras are another low power, low latency camera technology, which output an asynchronous stream of intensity changes [12]. Many VO/VSLAM algorithms have been implemented using event cameras [13]; however, the bandwidth of data transferred is proportional to the manoeuvre speed – fast motion requires more processing. On the other hand, an FPSP can be programmed to output data at a consistent data rate, thus there is no significant fluctuation in the amount of data transferred under any sort of motion.

The objective of this work is to investigate this approach in estimating the pose of the FPSP in 3D space, predicting motions with all 6-Degrees of Freedom (DoF). The contributions of our work are:

- An efficient BInary feaTure Visual Odometry, BIT-VO, the first 6-DoF visual odometry which utilises the FPSP. Using no intensity information, our proposed method is able to accurately track the pose at 300 FPS, even under difficult situations where the state of the art monocular SLAM fails.
- A robust feature matching scheme, which uses our novel binary-edge based descriptor. Using a small, 44-bit descriptor, our system is able to track the noisy features computed on the focal plane in the SCAMP-5 image sensor itself.
- Extensive evaluation of our system against measurements from a motion capture system, including difficult scenarios such as violently shaking the device 4-5 times a second.

The remainder of the paper is organised as follows. Section II describes the SCAMP-5 and reviews related work. Section III provides an overview of our system, together with the notations used. Section IV and Section V explain the proposed visual odometry algorithm. Section VI details our experimental results. Finally, Section VII concludes our work and discuss directions for the future.

II. BACKGROUND

This section provides a background and literature review on two topics: the SCAMP-5 and VO using unconventional vision sensors.

[1] Imperial College London, Department of Computing
[2] Ryerson University

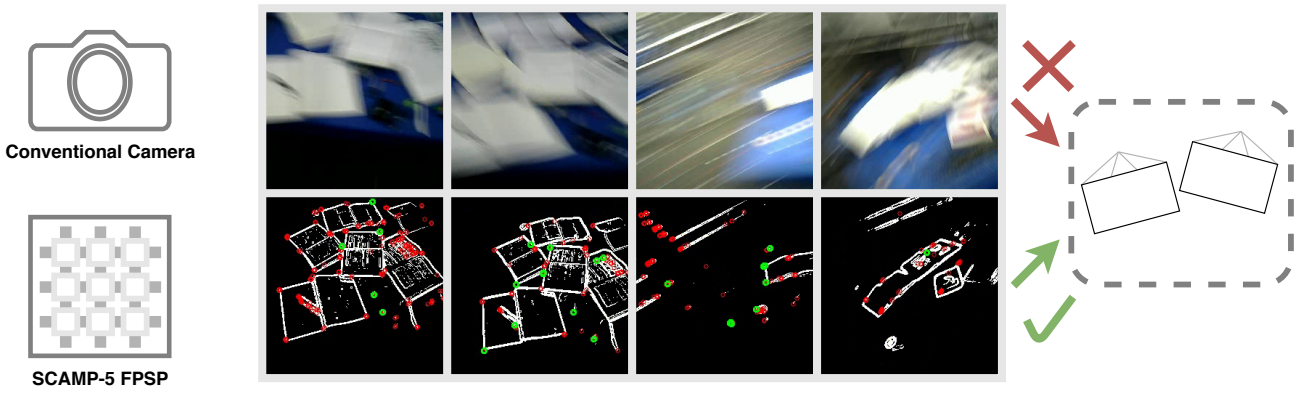

Fig. 1: Comparison of the data used by our proposed VO vs conventional VOs. Our system does not use intensity images (top row) but uses the binary edges and corners (bottom row) extracted by SCAMP-5 at 300 FPS. Notice that the edges, when extracted at a high frame rate, are tolerant against motion blur, and are sharp even when the device is subject to violent motions. For the conventional camera (operat7ing at 20 FPS) , such motion severely blurs the images.

A. SCAMP-5 FPSP

An FPSP is a general-purpose vision chip, where sensors and processor are integrated together on the same silicon [7]. Our work uses the SCAMP-5 vision system [14], which is an FPSP with a resolution of 256×256 pixels. On the focal plane, each one of the 65,536 pixels combines a photodiode with a Processing Element (PE). The device is programmable in a Single Instruction, Multiple Data (SIMD) fashion, where all of the PEs execute the same instruction. Each one of the PEs is capable of storing local data using 7 analog and 13 1-bit registers. Each PE can also perform simple computations such as logical and arithmetic operations. The arithmetic operations occur in the analog domain and directly on the analog registers without the need for digitisation [11]. The nature of analog computation results both in arithmetic operation incurring noise, and values stored on analog registers degrading over time [15]. A PE can communicate with neighbours to its north, east, west or south by copying its register value. Once the computation is complete, data can be read-out from the device in the form of coordinates, binary frames, analog frames, or global data (e.g. regional summation) [14]. In particular, coordinates can be read-out using event-readout, where the cost, in time and energy, is proportional to the number of events rather than the image dimension. This flexibility allows data reduction to occur on the focal plane [11], and, together with highly parallel instruction execution, FPSPs can perceive and process visual information at a very high frame rate.

When developing algorithms for the SCAMP-5 vision system, not only there are inaccuracy and noise in the analog computation, but there are resource constraints which make the porting of a computer vision algorithm rather complicated. The instruction set is limited; a wide range of logical operators are available for the 1-bit registers but only simple arithmetic operators (e.g. addition and subtraction) are available for the analog registers. However, there are few available registers. Furthermore, there is no global memory. A PE's only means of communication is to share data with its adjacent neighbours. Although these factors provide challenges, there have been successful implementations of complex algorithms, for example, a convolutional neural network capable of classifying handwritten digits at 2260 FPS [16], ternary weight CNNs [17], and a tracker for a ground target from a UAV at over 1000 FPS [18].

B. Visual Odometry Using Unconventional Vision Sensors

The term visual odometry (VO) was first coined by Nister et al. [19]. VO is the process of determining the egomotion of a sensor using visual information. Many algorithms have been proposed for VO using conventional vision cameras, for a review, see [20]. With the introduction of unconventional visual sensing technologies such as FPSPs and event cameras, the potential use of such sensors in VO is an active field of research.

Few works exist performing VO using FPSPs; however, none of them are 6-DoF. A 4-DoF VO algorithm using a direct method was proposed in [21]. This approach divides the image into N tiles and estimates an optic flow of each of the tiles efficiently on the focal plane. These vectors are decomposed using ordinary least squares to predict the yaw, pitch, roll and z-axis motion of the device. The computation all occurs on the SCAMP-5, allowing the algorithm to operate at 400-500 FPS. Another 4-DoF VO algorithm which instead uses a feature-based approach was proposed in [22]. The edge features are extracted from the captured image and are aligned against a keyframe using image shifting, scaling, and rotation. All of the image manipulations occur on the focal plane. By measuring the amount of shift, scaling and rotation required to align two images, an estimate of the yaw, pitch, roll and motion in the z-axis is obtained. Given sufficient lighting, the algorithm is capable of operating at over 1000 FPS. The method is extended in [23] to estimate the pose of an agile UAV by performing perspective correction using IMU data. Both VO methods achieve high frame rate by performing all of the computation on the SCAMP-5 device; however, they are limited to 4-DoF tracking, restricting their

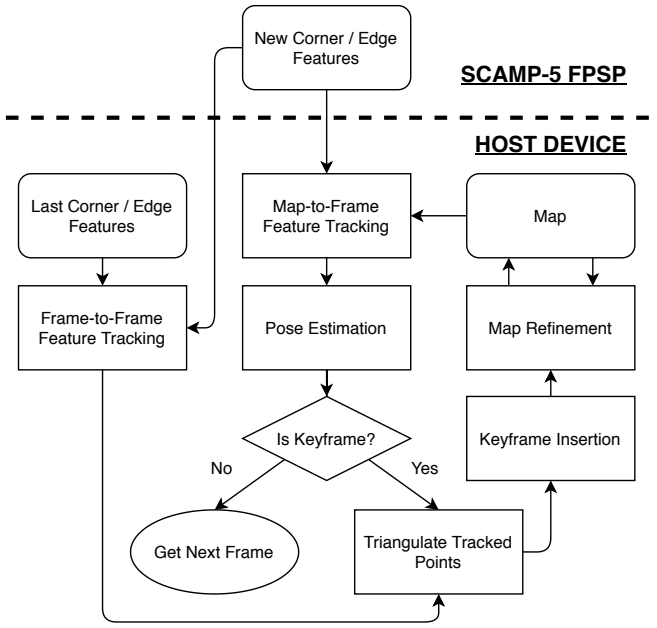

Fig. 2: Tracking and Mapping pipeline. The pipeline runs on an FPSP and a host device, minimising data flow from the sensor to host device,

use cases to platforms which are mechanically constrained to motion in one direction.

Many algorithms along with benchmarking datasets have been proposed for VO using event cameras. For further information about algorithms, challenges and future directions, see the recent review [13] by Gallego et al. Event cameras have also been combined with frame-based cameras [24] to improve VO and other algorithms by detecting features such as edges and corners [13]. While the combined approach shares similarities with the FPSPs architecture, a major difference is that FPSPs is capable of processing data in-situ.

III. System Overview

Our main contribution is a 6-DoF monocular visual odometry which operates in real-time at 300 FPS. An overview of our system flow is summarised in Fig. 2. The initialisation is omitted for simplicity. Feature extractions are performed on the SCAMP-5, while feature tracking and VO operates on the host device which is, for example, a consumer-grade laptop. The system operates only on the binary edge image and corner coordinates, thus no pixel intensity information is ever transferred (Fig. 1). Only using the limited information, we demonstrate that it is possible to create a VO system which is robust against rapid motion.

A. Notations

The world frame is represented with $\mathbf{w}$ and the camera frame is represented with $\mathbf{c}$. The position $\mathbf{p}$ in world frame is denoted as ${}_{\mathbf{w}}\mathbf{p}$. The rigid-body transformation $\mathbf{T_{c,w}} \in \mathbf{SE(3)}$ expresses the transformation from the world to the camera frame. This allows a point in the world frame ${}_{\mathbf{w}}\mathbf{p}$ to be mapped to the camera frame by ${}_{\mathbf{c}}\mathbf{p} = \mathbf{T_{c,w}}\,{}_{\mathbf{w}}\mathbf{p}$.

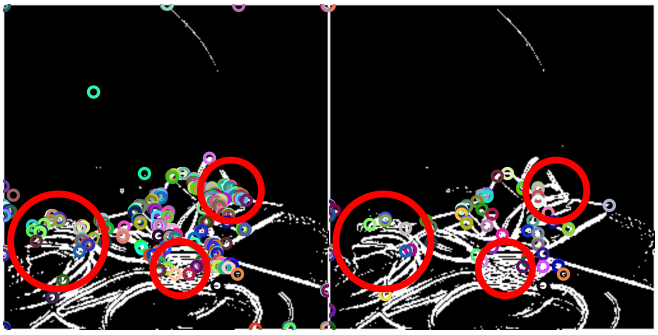

Fig. 3: Illustration of the effect of noisy analog computation. Between two consecutive frames, many corners appear and disappear. The device was mounted on a tripod to ensure stability of the device across multiple frames.

IV. Feature Detection and Matching

This section outlines how features are detected on the FPSP device, and how these features are matched against previous ones on the host device.

A. Feature Detection

Corner and edge features are computed on an FPSP, and it operates at a high frame rate of up to 330 FPS. FAST Keypoint Detector [25] is used for the corner detection. For edge detection, the magnitude of the image gradient is thresholded to find edges [22]. An existing implementation of FAST Keypoint Detector for the SCAMP-5 [10] is used in our work, although suppression of features is disabled as it is not repeatable. Unfortunately, performing repeatable suppression techniques such as non-maximal suppression is difficult due to the noisy analog computation on SCAMP-5. The noisy computation leads to not only incorrect inequality comparisons but also incorrect computation of the compared values. For every incoming frame, SCAMP-5 detects at most 1000 corner features which are read-out using an event-readout. For the binary edge image, the whole 256×256 bit image is transferred, rather than the pixel coordinates. In SCAMP-5, coordinates are expressed as an 8-bit pair, hence, event-readouts are only efficient if the number of events $N_{events} < 4096$. This is only 6.25% of all the available pixels, and, we find that in a typical indoor scene, around 10-15% of the pixels are classified as an edge feature.

B. Feature Matching

The task of matching the corner features is challenging for two reasons: a) feature extraction suffers from noise in analog computation, and b) multiple features are extracted per visual corner. Since SCAMP-5 performs computations using an analog circuit, corners are not reliably extracted in every frame (Fig. 3), causing incorrect association if one uses a naive method such as nearest neighbour matching. For each incoming frame of features, one may search a small local neighbourhood to avoid establishing correspondence if a corner is not extracted because of noise; however, due to (b), it will misidentify correspondence with an incorrect corner feature. This will build up error and result in many incorrect associations, and thus poor visual odometry.

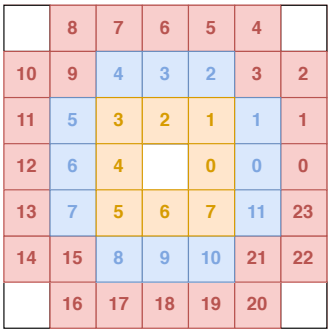

Fig. 4: Descriptor sampling pattern. Different colours denote a different ring, and indices correspond to the bit index.

1) Local Binary Descriptors from Edges: To establish robust correspondences across multiple feature frames, we propose a feature descriptor which only uses the local binary edge information. Our descriptor is tiny – only 44-bit in length thus is space-efficient and is fast to compute. Unlike other binary descriptors [26], [27], [28], we do not have access to the image intensity information. As shown in Fig. 4, three independent rings $\{r1, r2, r3\}$ are formed around a corner of interest. Each element of the ring stores a bit from the corresponding pixel of the binary edge image. A 7×7 patch is used as it fits in a single 64-bit unsigned integer. This allows the patch data to be converted to the rings efficiently using bitwise manipulation. To add a rotation invariance to our descriptor, the orientation of each of the features are computed. Assuming a coordinate frame with the origin set to the corner feature of interest, the intensity gradient magnitude $G(x, y)$ [29] is used to compute the orientation.

$$\theta = \tan^{-1} \frac{\sum_{x,y} yG(x, y)}{\sum_{x,y} xG(x, y)}, \tag{1}$$

where x, y are the coordinates of the 7×7 patch. Since the gradient image is binarised, Eq. 1 is approximated by

$$\theta = \tan^{-1} \frac{\sum_{x,y} yB(x, y)}{\sum_{x,y} xB(x, y)}, \tag{2}$$

where $B(x, y)$ is 1 if image point (x, y) is classified as an edge, and 0 otherwise. The rotation invariance is achieved by bit-rotations of the rings independently [26], based on the orientation θ. At each ring, the number of bits to rotate is determined by $rotate\_by(\theta, r) = \lfloor \theta \cdot \#r/360 \rfloor$ where $r \in \{r1, r2, r3\}$ and $\#r$ is length of the ring. Finally, the descriptor is computed by taking the disjunction of $\{r1, r2, r3\}$ after shifting $r1$ by $\#r2 + \#r3$ and $r2$ by $\#r3$. The descriptors are compared against each other using the Hamming distance, which is performed efficiently using SSE instructions. Although our descriptors are not scale-invariant, they are sufficient for small indoor environments.

2) Frame-to-Frame Matching: The high frame rate of our system enables efficient frame-to-frame feature matching. Given frames, $\{F_1, \ldots, F_n\}$, a local neighbourhood around a feature in F_i is matched against features in F_{i+1}. Similarly, features in F_{i+1} are matched against features in F_{i+2}. By following these matches, features in F_i can be matched against any arbitrary frames, assuming that they stayed in sight. This enables feature tracking to take advantage of the small inter-frame motion. By searching a small radius of $3-5$ pixels, a feature which minimises the Hamming distance is selected as a candidate. If the descriptor distance to the candidate exceeds a threshold, the candidate does not form a match. In our implementation, the threshold is set to 10.

3) Map-to-Frame Matching: All of the visible map points are back-projected onto the image plane to find correspondences. Again, only a small radius is searched. Map points are observed by multiple keyframes, thus, they store multiple descriptors each. Similar to ORB-SLAM [6], the most descriptive descriptor is selected by finding a descriptor which minimises the median distance to all others.

V. VISUAL ODOMETRY

Our VO system uses information obtained through feature tracking to predict the 3D point-cloud structure of the scene and the pose of the SCAMP-5. Like PTAM [5], localisation and mapping are interleaved. The 3D map points are generated through triangulation of features, and the pose of the SCAMP-5 is estimated through minimisation of the reprojection error. The non-linear optimisation is solved using the Levenberg-Marquardt algorithm, implemented using the Ceres Solver [30]. Since the inter-frame motion is small, the non-linear optimisation is fast to converge, requiring at most 10 iterations.

A. Pose Estimation

Given a set of 3D map points and its correspondences on an image plane, poses can be estimated by minimising the reprojection error, which can be formulated as [31]:

$$\mathbf{T}_{\mathbf{c},\mathbf{w}} = \underset{\mathbf{T}_{\mathbf{c},\mathbf{w}}}{\operatorname{argmin}} \frac{1}{2} \sum_i \rho \left(\|\mathbf{u}_i - \pi(\mathbf{T}_{\mathbf{c},\mathbf{w}}\, {}_{\mathbf{w}}\mathbf{p}_i)\|^2 \right), \tag{3}$$

where the error between the projected 3D points $\pi(\mathbf{T}_{\mathbf{c},\mathbf{w}}\, {}_{\mathbf{w}}\mathbf{p}_i)$ and the corresponding feature coordinates $\mathbf{u}_i$ are minimised. $\rho(\cdot)$ is the Huber loss function which reduces the effect of outlying data [32]. Unlike PTAM [5] or ORB-SLAM [6], the velocity model is not used in pose estimation. At such high frame rate, inter-frame motion is small, thus the previous pose is a sufficiently good estimate of the current position. Furthermore, the addition of velocity model leads to worse initialisation if the camera motion violates this assumption, which occurs often during violent motion.

B. Map Refinement

Every map point keeps a reference of the keyframes that it was observed by. These relationships form a graph, which is used in the structures-only bundle adjustment, where the pose estimates for each of the keyframes remain fixed, and only the positions of the map points are optimised. This is solved robustly using the Huber loss and at the end of the optimisation, map points are removed if their residual exceeds the Huber functions tuning constant.

TABLE I: Absolute Trajectory Error of different sequences, computed using evo [35]. The total length of the trajectory, Root Mean Square Error and Median Error is reported.

Sequence	Length[m]	RMSE [m]	Median [m]
Long	68.5	0.108	0.078
Rapid Shake	5.6	0.015	0.011
Jumping	32.9	0.056	0.040
Circle	38.3	0.128	0.084

TABLE II: Absolute Trajectory Error comparison of using our proposed descriptor and using rotated BRIEF, computed using evo [35]. The total length of the trajectory, Root Mean Square Error and Median Error is reported.

Descriptor	Length[m]	RMSE [m]	Median [m]
Ours	38.3	0.128	0.084
Rotated BRIEF	38.3	0.123	0.107

C. Initialisation

The 5-point algorithm [33] with RANSAC [34] is used to perform bootstrapping. This gives a relative pose estimate, which is used to triangulate the initial 3D map. Features in the reference frame are tracked using frame-to-frame tracking until there are sufficient disparities. Disparities are computed by taking the median of the features pixel displacements. If it is greater than 20 pixels, relative pose estimation and triangulation is attempted. Upon triangulation, if any 3D map point has a parallax of fewer than 5 degrees, or is behind of either of the two cameras, they are removed from the map. Once over 100 map points are successfully triangulated, the system is initialised.

D. Keyframe Selection

To select which frames are suitable as a keyframe, similar to PTAM [5] and SVO [2], [3], the selection process is based on the displacement of the camera relative to the depth of the scene. A keyframe is inserted when all of the following conditions are satisfied: a) At least 200 frames have passed since the previous keyframe insertion, b) at least 50 features are tracked, and c) Euclidean distances between the current frame and all the other keyframes are greater than 12% of the median scene depth. When a frame is selected as a keyframe, first, 2D-3D correspondences are established through the back-projection of the map points into the image plane. This links the map point to the keyframes that observed it. For the features which are not yet triangulated, Frame-to-Frame tracker is inspected to see if there are any successful matches which satisfy the epipolar constraint. If not many matches are found (< 30 matches), brute-force matching of all the features is performed between the current and the last keyframe. This process ensures that a sufficient number of map points are created at every keyframe insertion.

VI. EXPERIMENTS

We have evaluated our proposed system against ground-truth data from the Vicon motion capture system. As our method is a monocular VO, the estimated trajectory is scaled and aligned to the ground truth data. Experiments have been conducted with the SCAMP-5 [14]. Raw intensity images are not recorded by SCAMP-5, because in this case, SCAMP-5 would act as a conventional camera, with a reduced frame rate. Thus, a direct comparison against other VO/VSLAM using a monocular camera or SCAMP-5 is not possible. Instead, a webcam was attached to SCAMP-5 to demonstrate that systems using a typical camera such as ORB-SLAM [6] lose track when subject to dynamic motions. Field of view

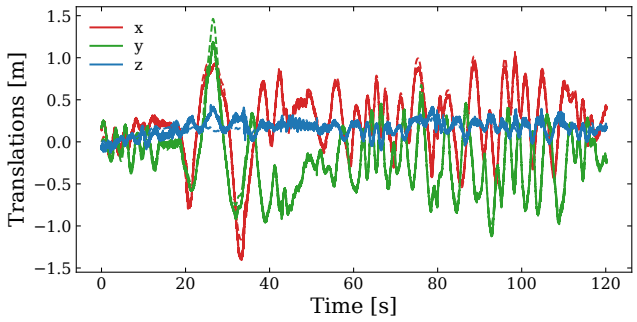

Fig. 5: Estimated x, y, z translations for "Long" sequence. Solid lines show our estimate and dotted lines are the ground truth.

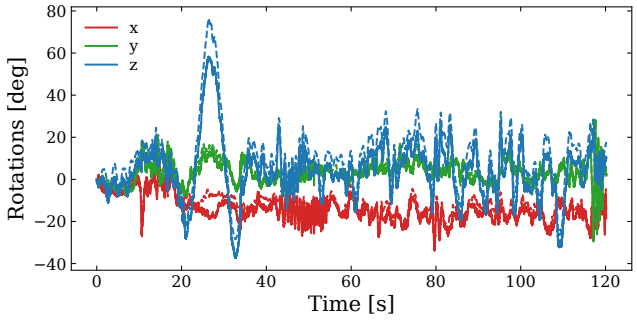

Fig. 6: Estimated x, y, z rotations for "Long" sequence. Solid lines show our estimate and dotted line are the ground truth.

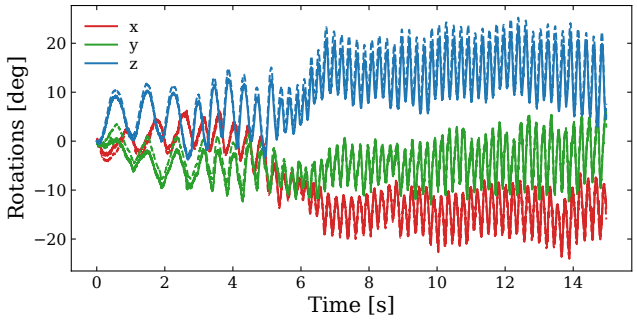

Fig. 7: Estimated x, y, z rotations for "Rapid Shake" sequence. Solid lines show our estimate and dotted line are the ground truth.

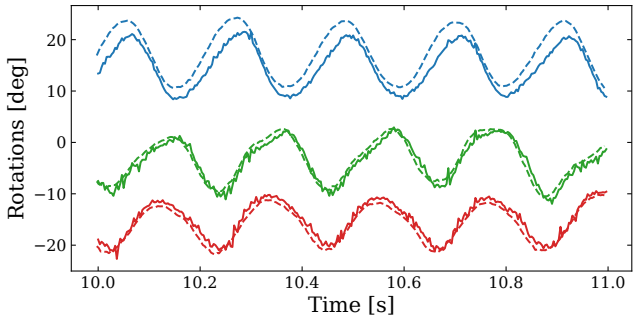

Fig. 8: Close-up view of rotation estimates for "Rapid Shake" sequence. Our proposed method is capable of tracking rapid rotation accurately.

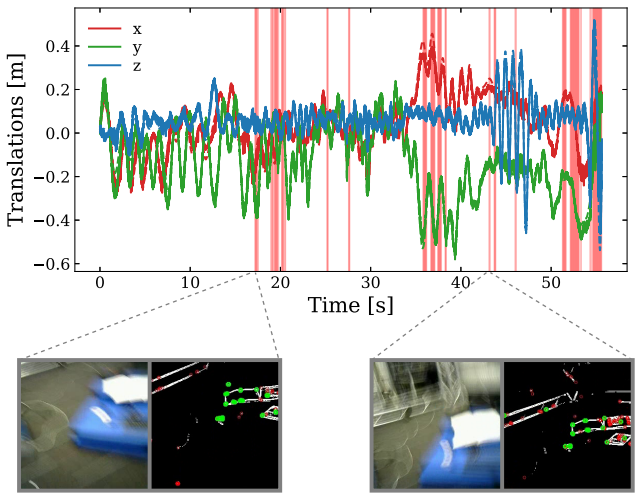

Fig. 9: Estimated x, y, z rotations for "Jumping" sequence. Solid lines show our estimate and dotted line are the ground truth. The pink region indicates that the ORB-SLAM lost track due to rapid motion.

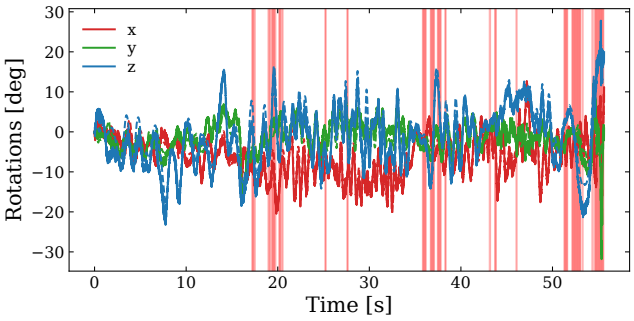

Fig. 10: Estimated x, y, z rotations for "Jumping" sequence. Solid lines show our estimate and dotted line are the ground truth. The pink region indicates that the ORB-SLAM lost track due to rapid motion.

between the two devices are different, hence, for fairness, best efforts were made to ensure both devices observe the same scene. All host computations were made on a laptop, with 4-core Intel i7-6700HQ CPU at 2.60GHz. Mapping and tracking used a single core, with visualisation, and communication with SCAMP-5 using an extra core each.

Due to the nature of SCAMP-5, we cannot use existing frame-by-frame video datasets for comparison. Thus, we evaluate our system against 4 different recordings: Long, Rapid Shake, Jumping, and Circle sequences. The test scene consisted of typical tabletop objects such as desktop monitor and books. Videos of the live running system is available on https://rmurai0610.github.io/projects/BIT-VO.

A. Accuracy and Robustness

The "Long" sequence repeatedly travels the test arena for 68.5m, where many features continuously enter and leaves the sight of the SCAMP-5. Fig. 5 and Fig. 6 illustrates the translation and rotation of our system over time. We notice a small rotational drift along the z-axis; however, there is no other significant drift, with small RMSE of 0.108m for the Absolute Trajectory Error [36] as summarised in Table I. Similar to a 4-DoF VO for SCAMP-5 [22], our system is able

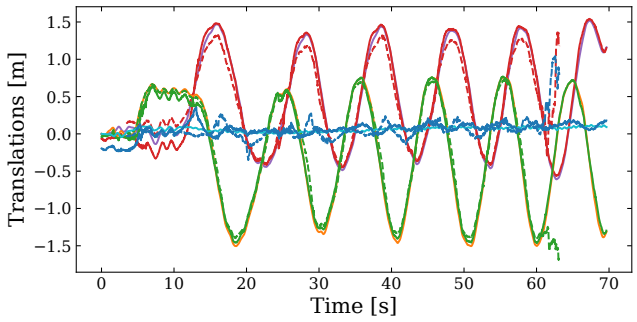

Fig. 11: Estimated x, y, z translation for "Circle" sequence. Solid lines show results from using our proposed descriptor, while dotted lines used rotated BRIEF. The estimated data x, y, z is plotted using red, green, blue and the ground truth data x, y, z is plotted using purple, orange, cyan respectively. Initialisation of rotated BRIEF version occured after our method.

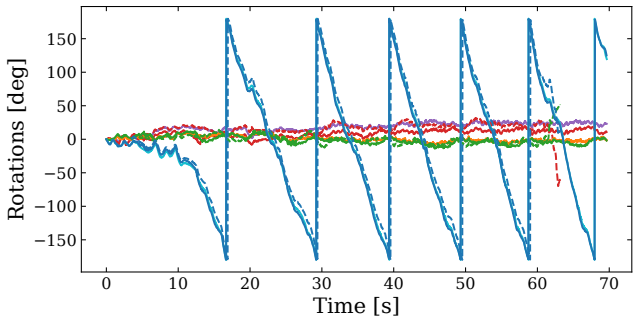

Fig. 12: Estimated x, y, z rotations for "Circle" sequence. Solid lines show results from using our proposed descriptor, while dotted lines used rotated BRIEF. The estimated data x, y, z is plotted using red, green, blue and the ground truth data x, y, z is plotted using purple, orange, cyan respectively. Note rotations along z-axis wraps as full 360 degrees loops are made.

to track violent rotations, as shown in Fig. 7. The system was subject to 4-5 shakes per second but was able to accurately track rotations along all three axes. A magnified view is provided in Fig. 8.

B. Comparison Against Visual SLAM

In the "Jumping" sequence, the device is subject to violent translational motions of up to 80cm caused by jumping as seen in seconds 42-48 of Fig. 9.

Fig. 9 and Fig. 10 highlights the advantage of operating at 300 FPS. Our VO pipeline is compared against ORB-SLAM [6] which uses data from a webcam operating at 20 FPS. The images are cropped to match the resolution of SCAMP-5 which is 256×256. Due to the nature of the FPSP, frames are not recorded on the device, but rather with a webcam which we have attached to the device. The pink highlighted regions in Fig. 9 is where ORB-SLAM has lost track. As shown, the images captured on the webcam suffers from motion blur, while the features from SCAMP-5 does not.

C. Comparison Against Other Descriptors

An alternative option would have been to use other binary descriptors such as BRIEF [27] or BRISK [28]; however, these methods use pixel intensity comparisons to build the descriptor. To use BRIEF descriptor in our setup, the pixel

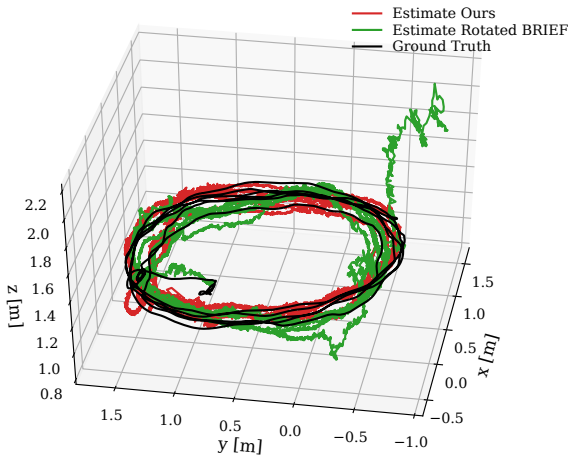

Fig. 13: Estimated 3D trajectory of "Circle" sequence using our proposed method: our pipeline (in red), rotated BRIEF descriptors (in green), and ground truth (in black).

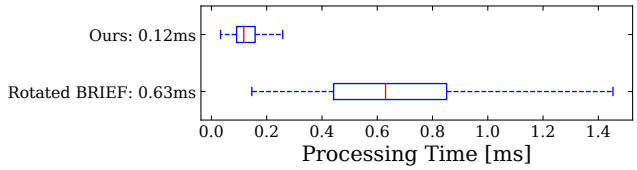

Fig. 14: Comparison of the processing time of our descriptor against rotated BRIEF.

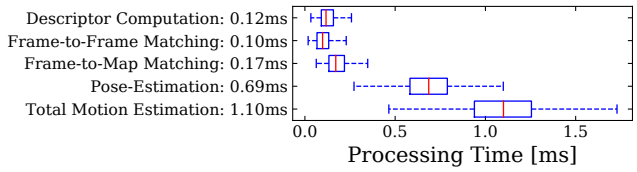

Fig. 15: Breakdown of the processing time required by our motion-estimation.

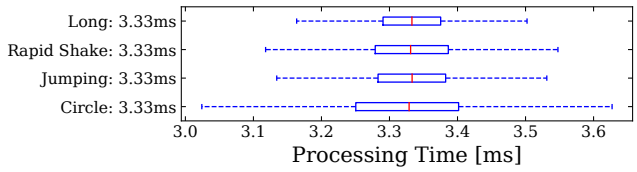

Fig. 16: Processing time per frame while running the system online on different sequences. Note that the bottleneck is the SCAMP-5, which outputs features at 300 FPS.

intensity comparison is replaced with an XOR operation. For the rotation invariance, we follow the same approach as ORB [37], with the orientation of the feature computed using Eq. 2.

To compare our descriptor against BRIEF, we have recorded the output features from the SCAMP-5. The Vicon room was explored in a circular motion while pointing the camera towards the centre of the room. A modified version of rotated BRIEF from OpenCV [38] was used for the experiments, which is a 256-bit long descriptor. Fig. 11 and Fig. 12 shows that there are no major differences in the two approaches, apart from 60 seconds onward where VO using rotated BRIEF fails. Fig. 13 depict the 3D trajectories of our approaches together with the ground truth. We notice that there are high-frequency noises present in our trajectories. For each frame, the noise from analog computation causes a different set of corners to be extracted. The difference causes incorrect correspondences, thus different formulation of the optimisation problem. This results in a shaky trajectory. When the correct features are again extracted, through descriptor matching, incorrect matches are removed, thus as demonstrated our system does not build up the noise over time. Table II compares the absolute trajectory error of using two different descriptors. For a fair comparison, we have excluded all measurements after 58 seconds for rotated BRIEF, such that the failed trajectory is excluded from the metric computation. We observe no significant difference in the accuracy of tracking in using either of the descriptors. The main advantage of our descriptor is visible in Fig. 14. The "Circle" sequence was executed offline using our descriptor and rotated BRIEF for 10 iterations each, and the time required to compute descriptor per frame is reported. Looking at the median, our approach is more than 5 times faster than rotated BRIEF.

D. Runtime Evaluation

Breakdown of the runtime of the motion-estimation is provided in Fig. 15. The timing is measured offline over 10 iterations of the "Circle" sequence. Our motion estimation is highly efficient, and the median time required to estimate the pose is 1.10ms, which translates to a frame rate of over 900 FPS. Currently, our system does not separate map-refinement onto different thread during keyframe insertion. The median of processing time for keyframe insertion is 3.17ms, with 2.22ms, 3.98ms at 0.25, 0.75 quantile respectively. When operating at 300 FPS, time budget is only 3.33ms, thus keyframe insertion combined with motion-estimation exceeds our allowance. However, within one or two frames, the excess is resolved. For a latency-critical application, it is possible to offload the keyframe insertion to a different thread. The runtime of the different sequences when operating the system live is reported in Fig. 16. We execute SCAMP-5 at 300 FPS, not at full capacity of 330 FPS for stable frame rates. As shown, our system is limited by the frame rate of the SCAMP-5, not by our VO algorithms. "Circle" sequence has the largest inter-quantile-range, as it required more keyframe insertions when compared to other sequences.

VII. CONCLUSION

We have presented BIT-VO, which is capable of performing VO at 300 FPS by using binary edges and corners computed on the focal plane. Our system is simplistic and minimal, yet it is sufficient to work in challenging conditions, highlighting the advantage of operating at high effective frame rates. In the proposed pipeline, a robust feature matching scheme using small 44-bit descriptors was implemented. FPSP's analog computation introduces noise to the values, but the proposed method is able to distinguish the noisy features. In future, we plan to incorporate a noise model for the computation of the FPSP, to improve the accuracy of

the algorithms. One of the key challenges in working with FPSPs is benchmarking of VO/VSLAM against conventional methods. If full intensity images are recorded from an FPSP for benchmarking purposes, the FPSP would not be able to operate at its high frame rate. A possible solution is to create an automated system to repeat the exact same trajectory multiple times.

This work will inform the design of future FPSP devices with higher computational capability, light sensitivity and pixel count. The programmable nature of the FPSP device, in contrast to, for example, event cameras, offers the prospect of higher accuracy, and enhanced robustness through greater adaptivity.

ACKNOWLEDGEMENTS

This research is supported by the Engineering and Physical Sciences Research Council [grant number EP/K008730/1]. We would like to thank Piotr Dudek, Stephen J. Carey, and Jianing Chen at the University of Manchester for kindly providing access to SCAMP-5.

REFERENCES

[1] A. Handa, R. A. Newcombe, A. Angeli, and A. J. Davison, "Real-time camera tracking: When is high frame-rate best?" in *European Conference on Computer Vision*. Springer, 2012, pp. 222–235.

[2] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *2014 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2014, pp. 15–22.

[3] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "SVO: Semi-Direct Visual Odometry for Monocular and Multi-Camera Systems," *IEEE Transactions on Robotics*, vol. 33, no. 2, pp. 249–265, 2016.

[4] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "DTAM: Dense tracking and mapping in real-time," in *2011 international conference on computer vision*. IEEE, 2011, pp. 2320–2327.

[5] G. Klein and D. Murray, "Parallel Tracking and Mapping for Small AR Workspaces," in *Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*. IEEE Computer Society, 2007, pp. 1–10.

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

[7] A. Zarándy, *Focal-plane sensor-processor chips*. Springer New York, 2011.

[8] J. N. Martel, L. K. Müller, S. J. Carey, and P. Dudek, "A real-time high dynamic range vision system with tone mapping for automotive applications," in *International Workshop on Cellular Nanoscale Networks and their Applications (CNNA)*, 2016, pp. 1–2.

[9] J. N. Martel, L. K. Müller, S. J. Carey, J. Müller, Y. Sandamirskaya, and P. Dudek, "Real-time depth from focus on a programmable focal plane processor," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 65, no. 3, pp. 925–934, 2017.

[10] J. Chen, S. J. Carey, and P. Dudek, "Feature extraction using a portable vision system," in *Vision-based Agile Autonomous Navigation of UAVs Workshop, 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

[11] S. J. Carey, A. Lopich, D. R. Barr, B. Wang, and P. Dudek, "A 100,000 fps vision sensor with embedded 535GOPS/W 256x256 SIMD processor array," in *VLSI Circuits (VLSIC), 2013 Symposium on*. IEEE, 2013, pp. C182–C183.

[12] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128×128 120 dB 15 μs Latency Asynchronous Temporal Contrast Vision Sensor," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 2, pp. 566–576, 2008.

[13] G. Gallego, T. Delbruck, G. M. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. Davison, J. Conradt, K. Daniilidis, and D. Scaramuzza, "Event-based vision: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2020.

[14] P. Dudek and P. J. Hicks, "A general-purpose processor-per-pixel analog SIMD vision chip," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 52, no. 1, pp. 13–20, 2005.

[15] S. J. Carey, D. R. Barr, B. Wang, A. Lopich, and P. Dudek, "Mixed signal SIMD processor array vision chip for real-time image processing," *Analog Integrated Circuits and Signal Processing*, vol. 77, no. 3, pp. 385–399, 2013.

[16] M. Z. Wong, B. Guillard, R. Murai, S. Saeedi, and P. H. Kelly, "AnalogNet: Convolutional Neural Network Inference on Analog Focal Plane Sensor Processors," *arXiv preprint arXiv:2006.01765*, 2020.

[17] L. Bose, J. Chen, S. J. Carey, P. Dudek, and W. Mayol-Cuevas, "A camera that CNNs: Towards embedded neural networks on pixel processor arrays," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019, pp. 1335–1344.

[18] C. Greatwood, L. Bose, T. Richardson, W. Mayol-Cuevas, J. Chen, S. J. Carey, and P. Dudek, "Tracking control of a uav with a parallel visual processor," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 4248–4254.

[19] D. Nister, O. Naroditsky, and J. Bergen, "Visual odometry," in *2011 International conference on computer vision and Pattern Recognition*, 2004, p. 652–659.

[20] D. Scaramuzza and F. Fraundorfer, "Visual odometry [tutorial]," *IEEE robotics & automation magazine*, vol. 18, no. 4, pp. 80–92, 2011.

[21] T. Debrunner, S. Saeedi, L. Bose, A. J. Davison, and P. H. J. Kelly, "Camera Tracking on Focal-Plane Sensor-Processor Arrays," 2019.

[22] L. Bose, J. Chen, S. J. Carey, P. Dudek, and W. Mayol-Cuevas, "Visual Odometry for Pixel Processor Arrays," in *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct. 2017, pp. 4614–4622.

[23] C. Greatwood, L. Bose, T. Richardson, W. Mayol-Cuevas, J. Chen, S. J. Carey, and P. Dudek, "Perspective Correcting Visual Odometry for Agile MAVs using a Pixel Processor Array," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 987–994.

[24] C. Brandli, R. Berner, M. Yang, S.-C. Liu, and T. Delbruck, "A 240×180 130 dB 3 μs Latency Global Shutter Spatiotemporal Vision Sensor," *Solid-State Circuits, IEEE Journal of*, vol. 49, pp. 2333–2341, 10 2014.

[25] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *European conference on computer vision*. Springer, 2006, pp. 430–443.

[26] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 7, pp. 971–987, 2002.

[27] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "BRIEF: Binary Robust Independent Elementary Features," in *European conference on computer vision (ECCV)*. Springer, 2010, pp. 778–792.

[28] S. Leutenegger, M. Chli, and R. Y. Siegwart, "BRISK: Binary Robust Invariant Scalable Keypoints," in *2011 International conference on computer vision*. Ieee, 2011, pp. 2548–2555.

[29] P. L. Rosin, "Measuring corner properties," *Computer Vision and Image Understanding*, vol. 73, no. 2, pp. 291–307, 1999.

[30] S. Agarwal, K. Mierle, and Others, *Ceres Solver*, [Accessed: 2019-06-06]. [Online]. Available: http://ceres-solver.org

[31] H. Strasdat, J. M. Montiel, and A. J. Davison, "Visual SLAM: why filter?" *Image and Vision Computing*, vol. 30, no. 2, pp. 65–77, 2012.

[32] Z. Zhang, "Parameter estimation techniques: A tutorial with application to conic fitting," *Image and Vision Computing*, vol. 15, no. 1, pp. 59–76, 1997.

[33] 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. 0756–777, 2004.

[34] 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, pp. 381–395, 1981.

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

[36] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 573–580.

[37] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *2011 International conference on computer vision*. Ieee, 2011, pp. 2564–2571.

[38] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.