Autonomous Robots
https://doi.org/10.1007/s10514-023-10122-8

High-frame rate homography and visual odometry by tracking binary features from the focal plane

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

Abstract

Robotics faces a long-standing obstacle in which the speed of the vision system's scene understanding is insufficient, impeding the robot's ability to perform agile tasks. Consequently, robots must often rely on interpolation and extrapolation of the vision data to accomplish tasks in a timely and effective manner. One of the primary reasons for these delays is the analog-to-digital conversion that occurs on a per-pixel basis across the image sensor, along with the transfer of pixel-intensity information to the host device. This results in significant delays and power consumption in modern visual processing pipelines. The SCAMP-5—a general-purpose Focal-plane Sensor-processor array (FPSP)—used in this research performs computations in the analog domain prior to analog-to-digital conversion. By extracting features from the image on the focal plane, the amount of data that needs to be digitised and transferred is reduced. This allows for a high frame rate and low energy consumption for the SCAMP-5. The focus of our work is on localising the camera within the scene, which is crucial for scene understanding and for any downstream robotics tasks. We present a localisation system that utilise the FPSP in two parts. First, a 6-DoF odometry system is introduced, which efficiently estimates its position against a known marker at over 400 FPS. Second, our work is extended to implement BIT-VO—6-DoF visual odometry system which operates under an unknown natural environment at 300 FPS.

Keywords Visual odometry · Focal-plane sensor-processor arrays

1 Introduction

Applications that require real-time scene understanding of the environment are often power-constrained. This includes mobile robotics and virtual/artificial reality goggles, for instance. In this work, we explore an unconventional approach to the problem of low-power, high frame rate vision: we compute directly within the light sensor, performing early-stage computation before converting signals to digital form. We demonstrate the power of this approach by computing sparse features efficiently on a representative hardware device. By transferring only the valuable data to later stages of the pipeline, our approach enables a wide range of applications—such as recognition and tracking—to operate with exceptionally high frame rates, low latency, and low power consumption.

The most widespread image formation on an imaging sensor (also referred to as the focal plane) has two phases: exposure and readout. During the exposure, photons are captured by the pixels and then the pixels are read out by the readout system. The readout system is composed of one or several signal amplifiers and analog-to-digital converters (ADC). The digital output of the readout system is then transferred to a host processor via a bus system for further processing. The frame rate and latency of a real-time image processing pipeline is governed by the maximum speed of image formation, and the processing speed. The maximum frame rate of an imaging system depends on four factors: (1) the exposure time, (2) the sensor readout speed, (3) the data transfer rate of the interface, and (4) the number of pixels.

A shorter version of the paper has been presented in IROS 2020.

✉ Paul H. J. Kelly
p.kelly@imperial.ac.uk

Riku Murai
riku.murai15@imperial.ac.uk

Sajad Saeedi
s.saeedi@torontomu.ca

[1] Department of Computing, Imperial College London, London, UK

[2] Toronto Metropolitan University, Toronto , Canada

Published online: 22 July 2023

The bottleneck for a fast and low-power image processing pipeline is created by the need to read input data from the sensor and transfer it to a processing unit. A major slowdown in the frame rate occurs at the readout system when the electric charges, generated by the photons, are converted to digital values since all pixels should go through the readout system (El-Desouki et al., 2009). The readout system also consumes 50-70% of the overall energy of the sensor (Likamwa et al., 2016). To avoid this bottleneck, two measures are required: (1) processing data in analog form to avoid ADC readout delay and energy cost such that the data size is reduced post-processing, and (2) minimizing the amount of data going through the readout system. In other words, data needs to be processed in analog form, on the vision sensor chip, immediately after the data is captured—and, as a result, the volume of data transferred to the host needs to be reduced.

Our focus is on vision-based algorithms for estimating camera pose, such as visual odometry (VO) and Visual Simultaneous Localisation and Mapping (VSLAM). Accurately estimating pose is crucial for scene understanding and these algorithms benefit from operating at high frame rates. Firstly, by capturing images with a shorter exposure time than normal, motion blur is significantly reduced. Secondly, with smaller inter-frame motion, the optimisation problem becomes easier, leading to faster convergence (Handa et al., 2012).

Despite the benefits of high frame rates, most state-of-the-art algorithms operate at a frame rate of 30-60 frames per second (FPS), as naively increasing the frame rate increases the volume of the data required to be processed. Even for fast VO pipeline such as SVO (Forster et al., 2014, 2016), the authors recommend a camera that operates at an effective rate of 40–80 FPS.

In contrast, this paper seeks a way to utilise the advantages of the high frame rate in visual odometry tasks. By efficiently compressing the data from the image sensor to the host device, we can reduce the amount of data transferred and, as a result, the necessary processing power. We focus primarily on feature extraction which occurs at the early stage of the visual odometry/SLAM pipeline. For instance, ORB-SLAM2 (Mur-Artal and Tardós, 2017) requires ∼11ms for feature extraction for 640 × 480 RGB images. The inefficiency arises from the fact that the 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) 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 (Zarándy, 2011). 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 (Carey et al., 2013b), 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 cameras are another low power, low latency camera technology, which output an asynchronous stream of intensity changes (Lichtsteiner et al., 2008). Many VO/VSLAM algorithms have been implemented using event cameras (Gallego et al., 2020); 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. The contributions of our work are:

- A high frame rate camera pose estimation system given some prior knowledge about the scene. Using AprilTags, homography is computed using just the features extracted by SCAMP-5.
- An efficient BInary feaTure Visual Odometry, BIT-VO, the first 6-DoF visual odometry which utilises the FPSP. Given no prior information about the scene, and 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 novel binary-edge based descriptor, which is small and is only 44-bit long. Using noisy features computed on the focal plane of the SCAMP-5 image sensor, our system can track keypoints using this binary descriptor.
- 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. Additionally, a comparison was made between the system and ORB-SLAM2 while varying the exposure time to demonstrate the advantage of a fast frame rate.

The remainder of the paper is organised as follows. Section 2 describes the background and SCAMP-5 FPSP. Section 3 presents a study with a tag tracking with SCAMP-5. Section 4 describes the VO algorithm in unknown environments. Section 5 details our experimental results. Section 6 presents the related work. Finally, Sect. 7 concludes our work and discusses directions for the future.

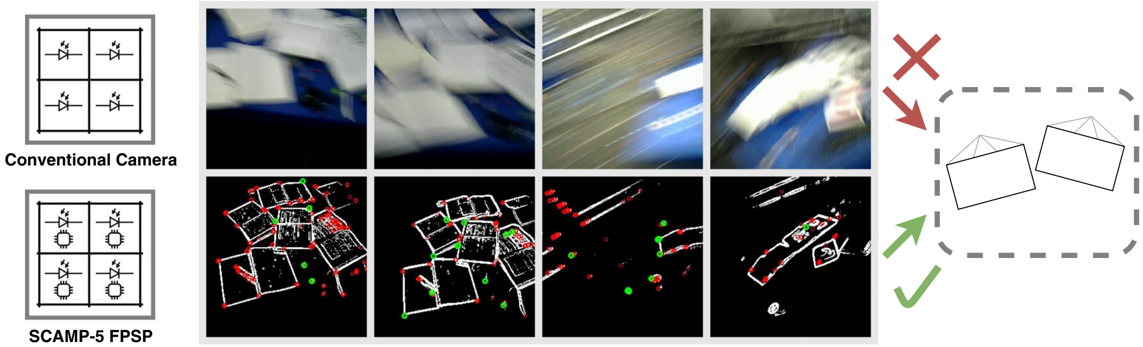


Fig. 1 Comparison of data utilisation in conventional VO algorithms vs the algorithm proposed in this paper; (top row) in conventional VO algorithms, intensity images captured by a conventional camera are used for processing. (bottom row) Unlike the conventional algorithms, the proposed VO algorithm utilises the computing power of the SCMAP-5 focal-plane sensor-processor arrays to extract binary edge and corner features, at 300 FPS, and use them for VO. At such a high frame rate, the edges are sharp and do not suffer from motion blur, even when the camera moves fast. On a conventional camera, running at 20 FPS, such motion results in blurry images (Color figure online)

2 Background

This section provides a background and literature review on these topics: analog computation, vision sensors, and SCAMP-5 FPSP.

2.1 Analog computation

In digital computing, multiple distinct binary signals (bits) are used to represent a state or a number. In contrast, in current-mode analog computing, the stored electrical potential is used to express a value, which can be read as electrical current. The hardware needed to do arithmetic computation on analog signals is considerably less than digital systems. For instance, to add two analog values, currents are joined from two sources representing the original values, while in digital for two 8-bit numbers, an adder needs many transistors—depending on circuit design, between 6 and 28 transistors are required per bit. However, analog computation presents several challenges, including limited-range value representation, limited numerical precision in computation, circuit inaccuracies, noise leakage, and thermoelectric effect (Amant et al., 2014). In the SCAMP-5 design used in this work, values are represented as the charge stored on capacitors, which degrades over time. The algorithms must be aware of the analog nature of the computation and should be resilient to such noises.

2.2 Vision sensors

CCD (charge-coupled device) and CMOS (complementary metal-oxide-semiconductor) are the two mainstream imaging sensor technologies. In both sensors, the main building blocks are: pixel array, readout system, and digital logic. The pixel array converts photons to electric charges. The readout system includes analog to digital conversion. Finally, the digital logic controls the system operation, such as timing and driver. Of these three blocks, on most modern CMOS sensors, the readout system consumes more than 50% of the total sensor power (Likamwa et al., 2016), while the pixel array accounts for a small fraction of the power consumption (Kitamura et al., 2012). The ADC also creates a latency bottleneck. These limitations motivate a new design where analog processing is collocated with the pixel array, reducing and/or eliminating the work to be done by the ADC. Focal-plane sensor-processor Arrays (FPSPs) add a processor per pixel, where it is possible to do some level of processing before going through the ADC. Example FPSPs include ACE400 (Dominguez-Castro et al., 1997), ACE16k (Linan et al., 2002), MIPA4K (Poikonen et al., 2009), and SCAMP (SIMD current-mode analog matrix processor) chips (Dudek and Hicks, 2005)' (Carey et al., 2013b). More details about SCAMP-5 FPSP is presented in the next section.

2.3 SCAMP-5 FPSP

A conventional camera is a 2D array of light-sensitive elements, known as pixels. Focal-plane Sensor Processor Arrays (FPSP), also known as processor-per-pixel arrays (PPA) and cellular-processor arrays (CPA), add a small processor per pixel on the same die (Zarándy, 2011). SCAMP-5 (Dudek and Hicks, 2005) is a 256 × 256 FPSP totaling 65,536 pixels. Each pixel combines a photodiode with a Processing Element (PE). These PEs can execute an instruction simultaneously on their local data, resulting in Single Instruction, Multiple Data (SIMD) parallel processing.

Each PE has the ability to store local data using 7 analog and 13 1-bit registers, as well as perform simple computations such as logical and arithmetic operations. The arithmetic operations are carried out in the analog domain directly on the analog registers, eliminating the need for digitisation (Carey et al., 2013b). Processing in analog, with no digitisation, accelerates the computation; however, this in turn introduces limitations (Carey et al., 2013a). For instance, arithmetic operations become noisy. Moreover, analog values, stored on the registers, degrade gradually. After computation, data can be read out in different forms such as coordinates, binary frames, analog frames, or global data (e.g. regional summation) (Dudek and Hicks, 2005). The device supports event-readout for coordinate readout, where the cost – in time and energy – is proportional to the number of events rather than the image dimension. These features make it possible to perform data reduction and implement coordinate-based algorithms on the focal plane at frame rates much higher than conventional cameras (Carey et al., 2013b).

Due to the small size of the sensor-processor chip, there are limited resources available for each PE. One way to deal with the limited number of registers is to have registers shared among the pixel, but this will reduce the resolution of the sensor (Martel et al., 2015). Further, the instruction set is also constrained; only operations such as addition and subtraction are available, and for example, there is no multiplication. Additionally, there is no central memory, and the PEs can communicate data with their immediate adjacent pixels only. Though using this method, pixels far apart from each other can communicate with each other, but this is at the cost of losing the quality of data because every time that data is copied from one pixel to another, the noise will affect the data. Despite all these constraints, several computer vision algorithms have been implemented on SCAMP-5 FPSP. Examples include FAST keypoint detection (Chen et al., 2017a), 4 DoF visual odometry (McConville et al. 2020; Bose et al. 2017), (Debrunner et al., 2019), localisation (Castillo-Elizalde et al., 2021), target tracking (Greatwood et al. 2017; Liu et al. 2021), and depth estimation (Martel et al., 2017). However, the design and implementation of complex algorithm for FPSP, such as 6 DoF VO/SLAM, remains as challenging and open problems. There exist a few algorithms aiming at running neural networks on SCAMP-5, paving the way for visual odometry and SLAM algorithms that utilise deep neural network. Examples include convolutional neural networks (CNN) with approximated weights (Wong et al. 2020; Debrunner et al. 2018; Stow et al. 2022b), ternary weight CNNs (Bose et al., 2019), and binary weight CNNs (Liu et al. 2020; Bose et al. 2020). Accelerating such networks for on/near sensor implementations will benefit not only odometry and SLAM applications, but also other vision-based AI and robotic applications such as navigation (Stow et al., 2022a). Therefore, developing fast and efficient hardware, compiler, and software (Watanabe et al., 2014) is becoming an active field for on/near sensor applications. Recent trends for such accelerations are based on hardware acceleration, e.g. processing-in-memory (Lin et al., 2018), ISSAC (Shafiee et al., 2016), Eyeriss (Chen et al., 2017b), or limited-precision computations, e.g. MobileNet (Howard et al., 2017), Minerva (Reagen et al., 2016).

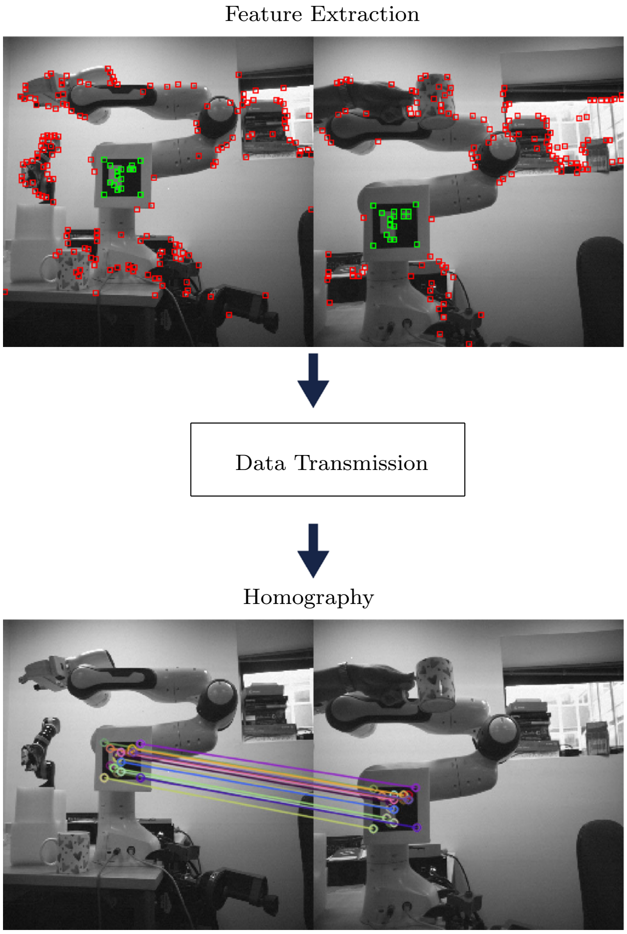


Fig. 2 AprilTag localisation: Feature extraction are performed on FPSP after applying Gaussian kernel to the image. The features are transmitted to a digital processor. Features marked with green squares are used to determine a perspective transformation (Color figure online)

3 Visual odometry against a known marker

With the potential of SCAMP-5 FPSP in fast and low-power processing in robotics and other applications that require tracking of objects or determining the pose of a camera, this simple pipeline is a demonstration of how one can easily perform tracking given prior knowledge of the scene. The experiment is based on tag detection and tracking, i.e. calculating a homography between consecutive frames. Here we use AprilTags (Olson, 2011), visual fiducials designed to be recognised easily, with applications in camera pose estimation and multi-robot systems. Our goal is to design a high-speed camera localisation system using AprilTags that can be used for applications that require accurate and fast pose estimation. Our tag-based tracking is an alternative approach which is much faster than conventional systems and consumes very low energy, enabling other applications in wearable devices with low power budget. The tag tracking has the following main steps (Fig. 2):

- **Feature Extraction**: This block extracts all FAST features (Rosten and Drummond, 2006) in a SIMD fashion (Chen et al., 2017a). FAST features are used in many computer vision application such as pose estimation and image stitching. Prior to feature detection, the image is convolved with Gaussian and Laplacian kernels in the focal plane.
- **Data Transmission**: Only sparse FAST corners are transmitted to a digital microcontroller, next to the camera, for further processing. Transmitting sparse data reduces the latency of the pipeline.
- **Homography**: On the microcontroller, a simple feature-density based algorithm was used to identify the tags in the image. Given the features from two consecutive tags, a homography is computed to recover the perspective transformation between the two frames (Olson, 2011), (Hartley and Zisserman, 2004). The pose of the camera can then be calculated easily using the homography (Malis and Vargas, 2007).

The tag detection runs at 402 Hz on SCAMP-5. This includes the combined computation in the focal plane and the accompanying microcontoller. The computation in the focal plane runs at 1848 fps (including the Gaussian kernel convolution and readout time for 300 features), and the microcontroller here creates a bottleneck to compute the homography. In fact, using a faster digital processor, would enable a faster frame rate. The average energy used for a pair of frames to compute the homography is 3.64 mJ.

A sequence of tracking data from SCAMP-5 was recorded to compare the error of the homography computation between a CPU based system (Intel Core i7-4712) and SCAMP-5. The average reprojection error of the homography implemented on SCAMP-5 is 0.381 pixels for 277 frames. The same metric for the complete digital system is 0.353 pixels, using the same feature selection threshold. This indicates that by using the new architecture, faster computation at low power consumption is achieved, but at a cost of minor accuracy loss due to the computations in analogue.

4 Visual odometry under unknown environment

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. 3. The initialisation is omitted for simplicity. Feature extractions are performed on SCAMP-5, while feature tracking and VO operates on the host device which is, for example, a consumer-grade laptop. The system exclusively operates on the binary edge image and corner coordinates, without transferring any pixel intensity information (as shown in Fig. 1). Despite using limited information, we demonstrate the feasibility of creating a robust VO system against rapid motion.

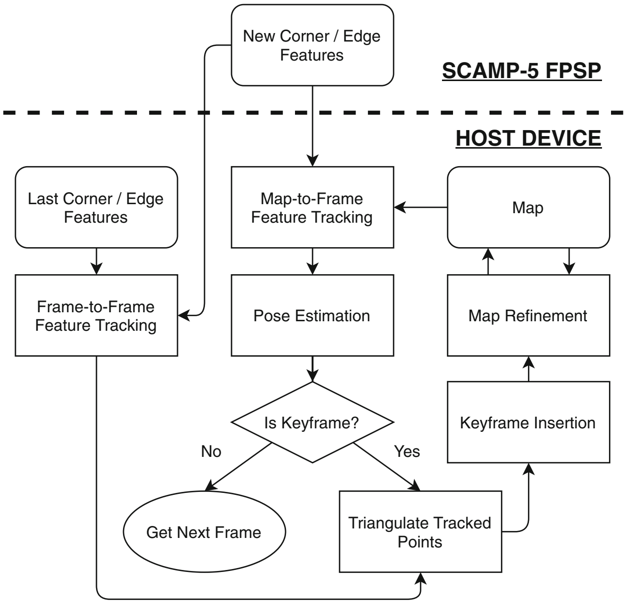

New Corner / Edge Features
SCAMP-5 FPSP
HOST DEVICE
Last Corner / Edge Features
Map-to-Frame Feature Tracking
Map
Frame-to-Frame Feature Tracking
Pose Estimation
Map Refinement
Is Keyframe?
No
Yes
Keyframe Insertion
Get Next Frame
Triangulate Tracked Points

Fig. 3 Tracking and Mapping pipeline. The pipeline runs on an FPSP and a host device, minimising data flow from the sensor to host device (Color figure online)

4.1 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.

4.1.1 Feature detection

The feature detection on the FPSP device involves using the FAST keypoint detector and binary edge detector, which are computed at a high frame rate of 330 FPS. An existing implementation of FAST Keypoint Detector for SCAMP-5 (Chen et al., 2017a) is used, with the suppression of features disabled due to the noisy analog computation. For edge detection, the magnitude of the image gradient is thresholded to find edges (Bose et al., 2017). For each frame, at most 1000 corner features are detected and are read-out as pixel coordinate using an event-readout. On contrary, the whole 256 × 256 bit binary image is transferred for edge features. In SCAMP-5, coordinates are expressed as an 8-bit pair, hence, event-readouts are only efficient if the number of

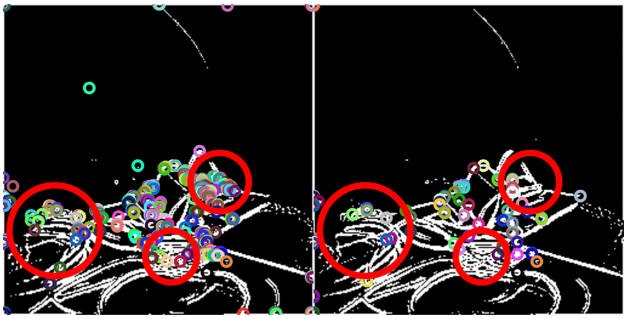
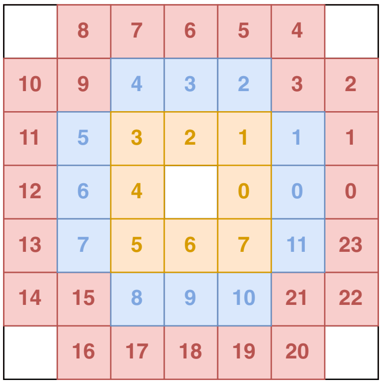


Fig. 4 Illustration of the effect of noisy analog computation. Between two consecutive frames, many corners appear and disappear. The device was mounted on a tripod to ensure stability of the device across multiple frames (Color figure online)



	8	7	6	5	4	
10	9	4	3	2	3	2
11	5	3	2	1	1	1
12	6	4		0	0	0
13	7	5	6	7	11	23
14	15	8	9	10	21	22
	16	17	18	19	20	

Fig. 5 Descriptor sampling pattern. Different colours denote a different ring, and indices correspond to the bit index (Color figure online)

events $N_{events} < 4096$. This is only 6.25% of all the available pixels, and we found that in majority of the cases, edge image exceeds this threshold.

4.1.2 Feature matching

Matching the corner features extracted from SCAMP-5 across multiple frames is challenging for two reasons: (a) feature extraction suffers from noise in analog computation, and (b) multiple features are extracted per visual corner. Due to the noisy nature of analog computation, corners are not repeatably extracted at every frame as shown in Fig. 4, resulting in incorrect data association if a naive method such as the nearest neighbour is used for the feature matching. This is problematic as incorrect data association degrades the accuracy and reliability of a visual odometry system. To address this issue, we devise a novel binary descriptor which can be used for feature matching.

4.1.3 Local binary descriptors from edges

We propose a feature descriptor that solely utilises local binary edge information to establish reliable correspondences across multiple feature frame. Our descriptor is tiny—only 44-bit in length thus is space-efficient and is fast to compute. Unlike other binary descriptors such as LBP (Ojala et al., 2002), BRIEF (Calonder et al., 2010), and BRISK (Leutenegger et al., 2011), we do not have access to the image intensity information. Our approach involves forming three independent rings $\{r1, r2, r3\}$ around a corner of interest, as depicted in Fig. 5, each containing a bit from the corresponding pixel of the binary edge image. We use a 7x7 patch that can be stored in a single 64-bit unsigned integer, making it possible to efficiently convert patch data to rings 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 $G(x, y)$ (Rosin, 1999) 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 (Ojala et al., 2002), based on the orientation θ. At each ring, the number of bits to rotate is determined by:

$$rotate\_by(\theta, r) = \lfloor \theta \cdot \#r/360 \rfloor \tag{3}$$

where $r \in \{r1, r2, r3\}$ and $\#r$ is length of the ring. The descriptor d is computed by:

$$d = (r1 << (\#r2 + \#r3)) \mid (r2 << \#r3) \mid r3 \tag{4}$$

where $<<$ operator is bit-wise shift, and $|$ operator is bit-wise or. 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.

4.1.4 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 with features in any other frames, as long as they remain visible. As the frame rate of the camera is high, inter-frame motion is small. 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, we have empirically chosen the threshold is to be 10.

4.1.5 Map-to-frame matching

All of the visible map points are projected onto the image plane to find correspondences. Again, only a small radius is searched. Each map points stores multiple descriptors as they are observed across multiple keypoints. Similar to ORB-SLAM2 (Mur-Artal and Tardós, 2017), we select the most descriptive descriptor for each map point by finding the descriptor that minimises the median distance to all other descriptors.

4.2 Visual odometry

This section summarises the implementation details of our VO system; however, it is kept brief as it is very similar to the standard VO systems like PTAM (Klein and Murray, 2007). Set of 3D map points of the scene is used to estimate the pose of SCAMP-5 by minimising the reprojection error. After every keyframe insertion, structure-only bundle adjustment (Strasdat et al., 2012) is carried out to refine the 3D map. Both of these nonlinear problems are solved using the Levenberg-Marquardt algorithm, which is implemented using Ceres Solver (Agarwal et al., 2010). As the inter-frame motion is small due to the high frame rate, the non-linear optimisation converges quickly, usually requiring no more than 10 iterations.

4.2.1 Bootstrapping

The bootstrapping process employs the 5-point algorithm (Nistér, 2004) with RANSAC (Fischler and Bolles, 1981) to obtain a relative pose estimate and triangulate the initial 3D map. The reference frame features are tracked using frame-to-frame tracking until sufficient disparities are obtained, with disparities computed by taking the median of the features' pixel displacements. If the disparity is greater than 20 pixels, relative pose estimation and triangulation are attempted. Triangulated 3D map points with a parallax of fewer than 5 degrees or behind either of the two cameras are removed from the map. The system is initialised once more than 100 map points are successfully triangulated.

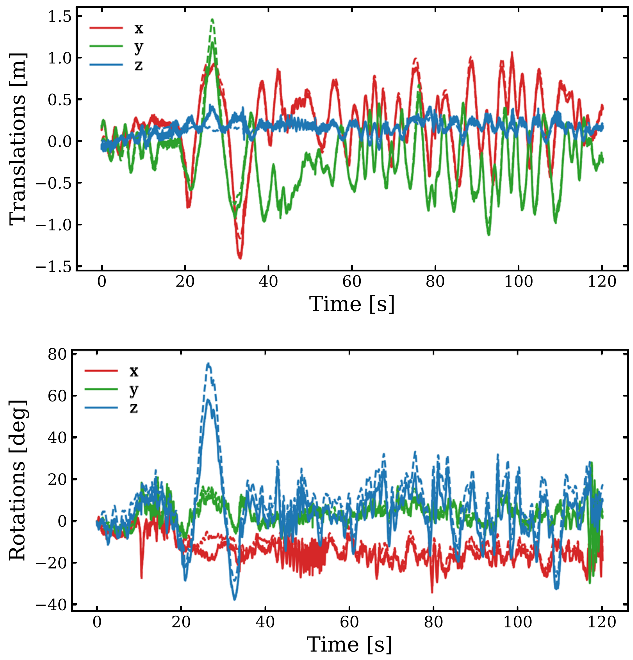


Fig. 6 **Top:** Estimated x, y, z translations for "Long" sequence. **Bottom:** Estimated x, y, z rotations for "Long" sequence. Solid lines show our estimate and dotted line are the ground truth (Color figure online)

4.2.2 Keyframe selection

To determine which frames are suitable as keyframes, we adopt a selection process similar to PTAM (Klein and Murray, 2007) and SVO (Forster et al., 2014), (Forster et al., 2016). This process is based on the camera displacement relative to the depth of the scene. A frame is designated as a keyframe if it meets the following criteria:

(a) at least 200 frames have elapsed 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 projection of the map points into the image plane, linking each map point to the keyframes that observed it. If some features have not yet been triangulated, the Frame-to-Frame tracker is checked to see if any matches satisfies the epipolar constraint. If fewer than 30 matches are found, brute-force matching of all the features is performed between the current and the last keyframe. This ensures that an adequate number of map points are generated with every keyframe insertion.

Table 1 Absolute Trajectory Error of different sequences, computed using evo (Grupp, 2017). 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

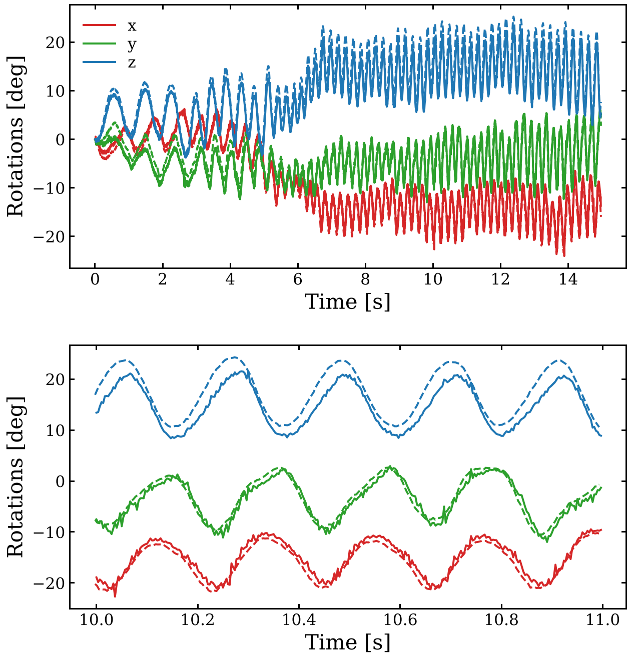


Fig. 7 **Top:** Estimated x, y, z rotations for "Rapid Shake" sequence. **Bottom:** Close-up view of rotation estimates for "Rapid Shake" sequence. Solid lines show our estimate and dotted line are the ground truth. Our method is able to track rapid rotations accurately (Color figure online)

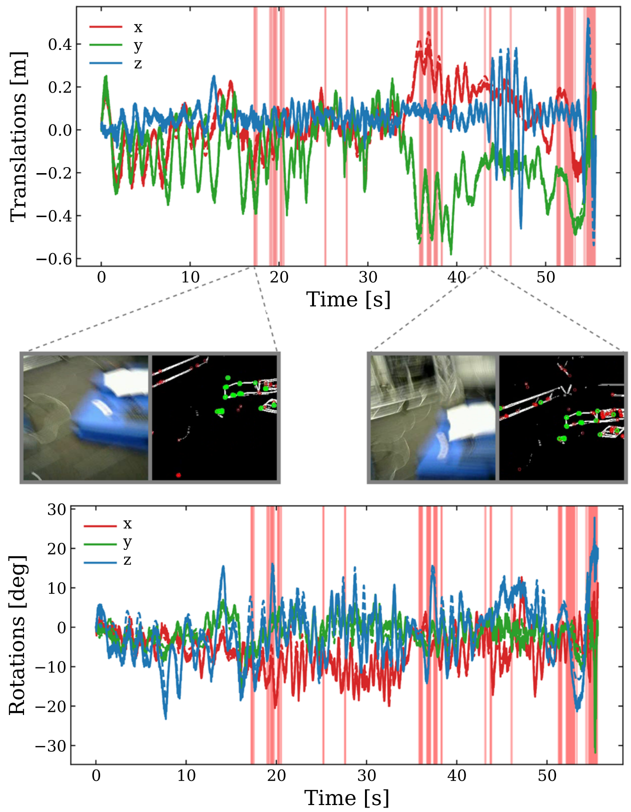


Fig. 8 **Top:** Estimated x, y, z rotations for "Jumping" sequence. **Bottom:** Estimated x, y, z rotations for "Jumping" sequence. Solid lines show our estimate and dotted line are the ground truth. The pink region indicates that the ORB-SLAM2 lost track due to rapid motion (Color figure online)

5 Experiments

We have evaluated our proposed system against ground-truth data from the Vicon motion capture system. The Vicon motion capture system is a high quality motion tracking system that is able to track the 6DoF motion of object, via tracking markers attached to them, at very high rates with submilimiter accuracy (Vicon, 1984). As our method is a monocular VO, the estimated trajectory is scaled and aligned to the ground truth data. Experiments have been conducted with SCAMP-5 (Dudek and Hicks, 2005). 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-SLAM2 (Mur-Artal and Tardós, 2017) lose track when subject to dynamic motions. Field of view 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 the project page.[1]

5.1 Accuracy and robustness

The"Long" test sequence involves repeatedly traversing a 68.5m test area, with numerous features appearing and dis-

[1] https://rmurai.co.uk/projects/BIT-VO/

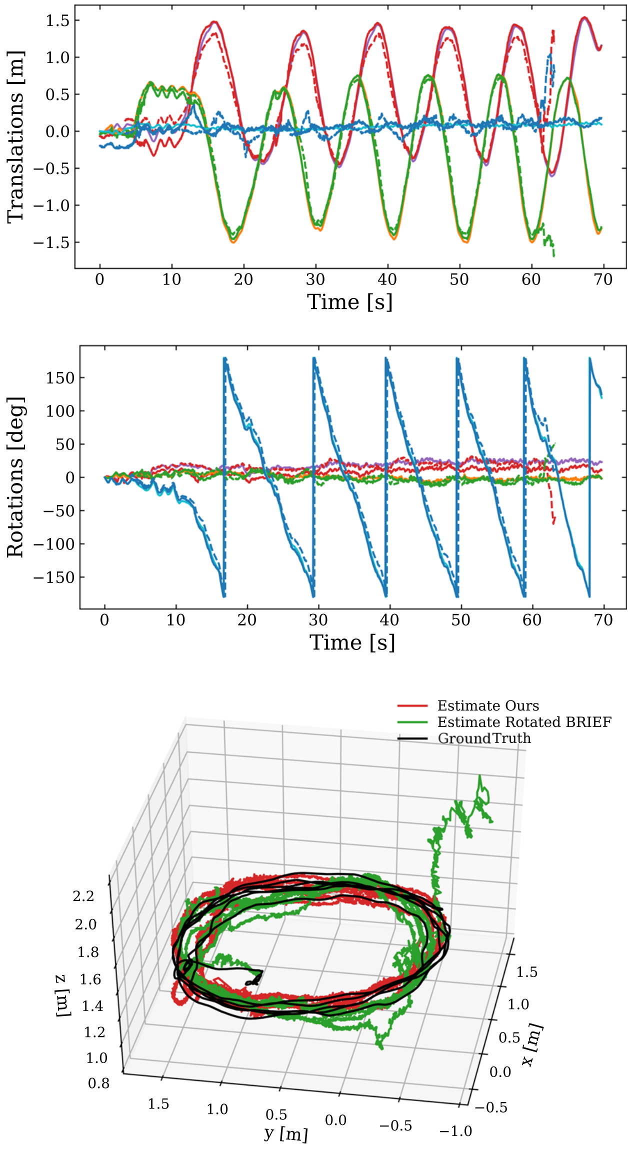


Fig. 9 **Top:** Estimated x, y, z translation for "Circle" sequence. **Middle:** Estimated x, y, z rotations for "Circle" sequence. Solid lines show results from using our proposed descriptor, while dotted lines used rotated BRIEF. The estimated data x, y, z is plotted using red, green, blue and the 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. **Bottom:** Estimated 3D trajectory of "Circle" sequence using our proposed method: our pipeline (in red), rotated BRIEF descriptors (in green), and ground truth (in black) (Color figure online)

appearing from the view of SCAMP-5. The translation and rotation of our system over time are depicted in Fig. 6.

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 (Sturm et al., 2012) as summarised in Table 1. Similar to a 4-DoF VO for SCAMP-5 (Bose et al., 2017), our system is able 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.

Table 2 Absolute Trajectory Error comparison of using our proposed descriptor and using rotated BRIEF, computed using evo (Grupp, 2017). 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

5.2 Comparison against a 4-DoF algorithm

To the best of the authors' knowledge, other than BIT-VO, there is no other 6-DoF odometry algorithm using FPSPs. However, we have designed an experiment to perform a comparison between BIT-VO and a 4 DoF algorithm described in (Debrunner et al., 2019). To have a fair comparison, we have constrained the motion of the BIT-VO. SCAMP-5 was mounted on a Turtlebot Waffle Pi mobile robot. The robot was exhibited yaw motions only. Because recording camera frames is not possible, we repeated the motion and performed live yaw estimation for both algorithms. The 4DoF algorithm is based on tracking intensity values. BIT-VO is based on tracking features. Feature-based odometry algorithms in general are more robust than intensity tracking based algorithms. The comparisons verify this hypothesis. Figure 10-(top) shows the ground-truth and yaw angles estimated via BIT-VO after the initialisation of the algorithm. Figure 10-(bottom) shows the same using the 4-DoF algorithm. From the graphs, it is evident the BIT-VO is able to track better. The RMS tracking error for BIT-VO is 0.0063 radians (0.3621 deg), and the tracking error for the 4-DoF algorithm is 0.1351 radians (7.7446 deg). To account for variations in the motion of the robot, this experiment was repeated five times. On average, the RMS for BIT-VO is 13.78× less than that of the 4 DoF algorithm.

5.3 Comparison against visual SLAM

In this section, we present the advantage of our high frame rate VO, running at 300 FPS, compared with ORB-SLAM2 (Mur-Artal and Tardós, 2017) running on images coming from a webcam at 20 FPS, as well as an Intel RealSense D445. For a fair comparison, in all runs, the images were cropped to 256 × 256 pixels, to match the resolution of SCAMP-5. One limitation when comparing with SCAMP-5 is that it is not possible to record images off SCAMP-5, as the processing is done on the chip. Any attempt to save images will defy the purpose of the focal-plane sensing/processing. Therefore, the images are saved

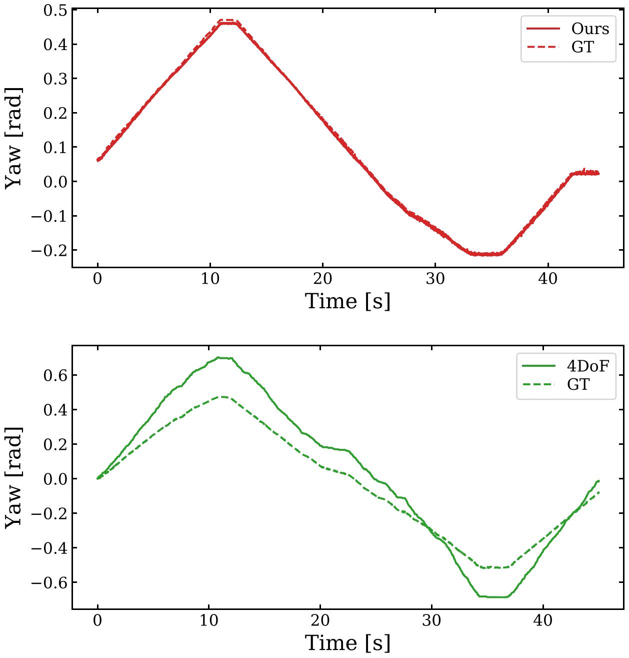


Fig. 10 Comparison between BIT-VO and a 4-DoF tracking algorithm on a constrained motion. **Top:** Ground-truth and estimates of yaw angle using BIT-VO. **Bottom:** Ground-truth and estimates of yaw angle using a 4-DoF tracking algorithm (Color figure online)

using the conventional camera that is the subject of the comparison. Figure 8 demonstrates that the camera undergoes an aggressive translation in the z-direction, with a maximum translation of 80 cm, see seconds 42 to 48, solid blue line. This "Jumping" motion, causes the ORB-SLAM2 to lose tracking features when experimenting on the webcam. Figure 8 highlights this in pink. The corresponding images demonstrate the motion blur caused by the jumping motion. SCAMP-5 survives the aggressive jumping motion.

Several experiments were performed running ORB-SLAM2 on images captured with an Intel RealSense D455 camera – state-of-the-art camera used in robotics and computer vision application. D445 is a global shutter camera, hence, does not suffer from rolling shutter affect. To have a fair comparison by accounting for the differences in the field of view and lenses of D445 and SCAMP-5, the field of view of D445 was reduced to the field of view of SCAMP-5. Then both cameras were taped together, such that they are both observing the same scene. The exposure times of both cameras were configured to be the same, i.e. 4 ms. At this low exposure, with rapid motion, ORB-SLAM2 fails to track multiple times, despite having sharp edge images. This is due to having less intensity information. Figure 12 shows an intensity image and its edge image captured via D445 at 4ms exposure. ORB-SLAM2 is a complete SLAM algorithm with localisation and loop closure components, but BIT-VO is only an odomtery algorithm.

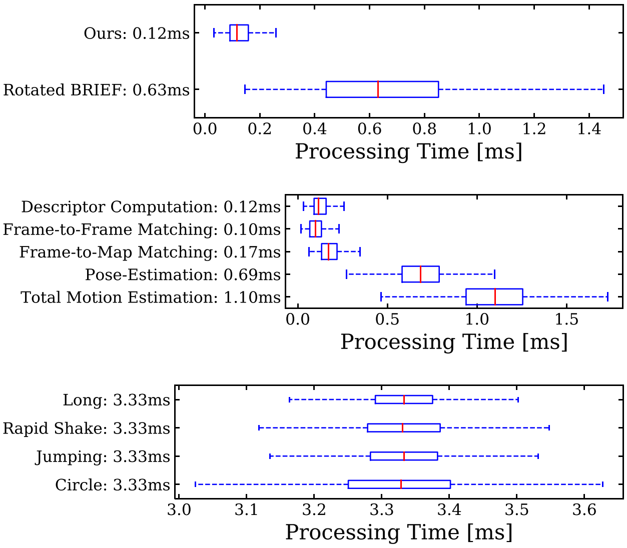


Fig. 11 Runtime breakdown of the system. **Top:** Comparison of the processing time of our descriptor against rotated BRIEF. **Middle:** Breakdown of the processing time required by our motion-estimation. **Bottom:** Processing time per frame while running the system online on different sequences. Note that the bottleneck is SCAMP-5, which outputs features at 300 FPS (Color figure online)

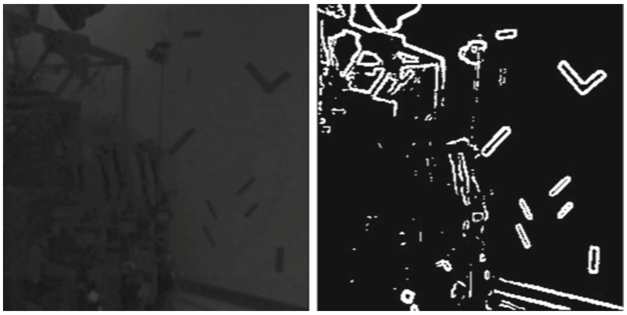


Fig. 12 An intensity image **(Left)** and its edges **Right** captured via D445 at 4ms exposure (Color figure online)

The qualitative experiment[2] demonstrates that ORB-SLAM2 fails to track multiple time, while BIT-VO is able to provide stable estimates.

5.4 Rotation invariance

The rotation invariance, presented in Sec. 4.1.3, helps to produce stable results. To verify this, we simply enabled and disabled the rotation invariance and compared the results, while doing pure role motion after initialisation of BIT-VO. The video of the experiments demonstrates the qualitative results[3]

[2] https://youtu.be/P6DgSxd61iY

[3] https://youtu.be/qPtt2LYzkQw

5.5 Comparison against other descriptors

Another possible choice of descriptor would have been to employ other binary descriptors like BRIEF (Calonder et al., 2010) or BRISK (Leutenegger et al., 2011). However, these methods construct the descriptor by comparing pixel intensities. As a comparison to our approach, BRIEF descriptor was modified by using XOR operation instead of pixel intensity comparison. To achieve rotation invariance, we adopt the same methodology as ORB (Rublee et al., 2011), where the feature's orientation is calculated using Eq. 2.

To compare our descriptor against BRIEF, we have recorded the output features from 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 256-bit long rotated BRIEF from OpenCV (Bradski, 2000) was used for the experiments. Figure 9 shows that there are no major differences in the two approaches, apart from 60 s onward where VO using rotated BRIEF fails. Bottom Fig. 9 depict the 3D trajectories of our approaches together with the ground truth. We notice that there are high-frequency noises present in our trajectories. The low resolution of SCAMP-5 camera means there is large round-off error in the pixel position of the features. Futhermore, due to the noise present in the analog computation of each frame, a different set of corners is extracted for the same visual scene, which leads to incorrect correspondences of the feature. This results in a shaky trajectory. When the correct features are again extracted, through descriptor matching, incorrect matches are removed. Table 2 presents a comparison of the absolute trajectory error using two different descriptors. To ensure a fair comparison, measurements after 58 s were excluded for rotated BRIEF, when it failed to track the trajectory. The results show no significant difference in the tracking accuracy when using either descriptor. However, our descriptor has a significant advantage in terms of computational efficiency, as shown in Fig. 11. The runtime for computing the descriptors per frame was measured offline over 10 iterations for the "Circle" sequence for both our descriptor and rotated BRIEF. The median runtime for our approach was more than five times faster than rotated BRIEF.

5.6 Runtime evaluation

Breakdown of the runtime of the motion-estimation which occurs on the host-device is provided in Fig. 11. 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 when executed offline, 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. The keyframe insertion combined with motion estimation exceeds the time budget of 3.33ms when operating at 300 FPS. However the excess is resolved within one or two frames. For latency-critical applications, it is possible to offload the keyframe insertion onto a different thread. The runtime of the different sequences when operating the system live is reported as well. We execute SCAMP-5 at 300 FPS, not at full capacity of 330 FPS for stable frame rates. Execution of the feature extraction on SCAMP-5 is our bottleneck which limits the overall frame rate of BIT-VO. The rest of the pipeline is capable of running at a much higher frame rate, thus, our approach is applicable to the next-generation FPSP devices which may have much faster computation.

Finally, "Circle" sequence has the largest inter-quantile-range, as it required more keyframe insertions when compared to other sequences.

6 Related works: visual odometry using unconventional vision sensors

While there are a few works utilizing FPSP for visual odometry, there is none performing 6-DoF. Bose et al. proposed a feature-based VO algorithm, estimating yaw, pitch, and roll rotations, as well as the translation along the z-axis (Bose et al., 2017). They first extract edge features and then align them with a keyframe. The alignment is done via shift, scale, and rotation operations, all performed on the sensor-processor chip. The 4-DoF algorithm can run up to 1000 FPS, under sufficient lighting. The algorithm later was extended and deployed on a UAV (Greatwood et al., 2018). A similar approach was used in McConville et al. (2020). Debrunner et al. proposed a VO algorithm, running at 400-500 FPS, capable of estimating yaw, pitch, and roll rotations in addition to the translation along z-axis Debrunner et al. (2019). Their method is a direct approach, using image intensity directly. They divide the focal plane into tiles and estimate the optic flow for each tile. The optic flow vectors are used to estimate the 4-DoF motion using ordinary least squares. In summary, the high-speed odometry in these 4-DoF algorithms has been made possible by sensing and computing on the same chip.

Event cameras are also closely related to FPSPs, albeit with the distinction that FPSPs can perform computation on the chip. There has been a lot of research conducted on odometry and tracking on event cameras. Kim et al. (2014), Kim et al. (2016). Using just an event-stream, the algorithm proposed in Rebecq et al. (2016) is capable of creating a semi-dense 3D map and operating on a CPU in real-time. The method works under challenging scenarios such as aggressive motions and illumination changes. It is also possible to augment the events with intensity values, by combining the hardware of event and conventional cameras (Brandli et al., 2014b). The dynamic and active pixel Vision Sensor (DAVIS) (Brandli et al., 2014a) transfers not only the asynchronous stream of events but the synchronously captured frames too. Using DAVIS, Kueng et al. (2016) performs visual odometry by extracting features from the frames, and tracking the features using the events through a variation of the Iterative Closest Point. For a comprehensive review of the recent developments regarding the algorithms on event cameras, please refer to Gallego et al. (2020).

7 Conclusion

In this work, we demonstratedna novel visual odometry algorithm for FPSPs used in robotics and computer vision applications, that are constrained by power budget and require very low latency. Examples of such applications include self-driving cars, wearable devices, pervasive computing, and IoT. We 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. We demonstrated that by processing data in the focal plane and limiting data movement only to important features, we can increase the frame rate and reduce the overall power consumption significantly while maintaining an equivalent accuracy.

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 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 (EPSRC) [grant number EP/K008730/1] and Natural Sciences and Engineering Research Council of Canada (NSERC). 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. We would like to thank Matthew Z. Wong for his assistance in tag detection for homography, and Yingying Li, Abrar Ahsan, and Ali Babaei for their assistance in creating Figs. 10 and 12.

References

Agarwal, S., & Mierle, K., Others (2010) Ceres Solver. http://ceres-solver.org, Retrieved Accessed: 2019-06-06

Amant, RS., Yazdanbakhsh, A., Park, J., Thwaites, B., Esmaeilzadeh, H., Hassibi, A., Ceze, L., & Burger, D. (2014). General-purpose code acceleration with limited-precision analog computation. In: ACM/IEEE International Symposium on Computer Architecture (ISCA), pp 505–516

Bose, L., Chen, J., Carey, SJ., Dudek, P., & Mayol-Cuevas, W. (2017). Visual odometry for pixel processor arrays. In: IEEE International Conference on Computer Vision (ICCV), pp 4614–4622

Bose, L., Chen, J., Carey, SJ., Dudek, P., & Mayol-Cuevas, W. (2019). A camera that CNNs: Towards embedded neural networks on pixel processor arrays. In: IEEE International Conference on Computer Vision (ICCV), pp 1335–1344

Bose, L., Dudek, P., Chen, J., Carey, S. J., & Mayol-Cuevas, W. W. (2020). Fully embedding fast convolutional networks on pixel processor arrays. *European Conference on Computer Vision ECCV, Springer, Lecture Notes in Computer Science, 12374*, 488–503.

Bradski, G. (2000). The OpenCV Library. Dr Dobb's Journal of Software Tools

Brandli, C., Berner, R., Yang, M., Liu, S. C., & Delbruck, T. (2014). A 240× 180 130 db 3 μs latency global shutter spatiotemporal vision sensor. *IEEE Journal of Solid-State Circuits, 49*(10), 2333–2341.

Brandli, C., Berner, R., Yang, M., Liu, S. C., & Delbruck, T. (2014). A 240×180 130 dB 3 μs latency global shutter spatiotemporal vision sensor. *IEEE Journal of Solid-State Circuits, 49*, 2333–2341.

Calonder, M., Lepetit, V., Strecha, C., & Fua, P. (2010). BRIEF: Binary robust independent elementary features. In: European conference on computer vision (ECCV), Springer, pp 778–792

Carey, S. J., Barr, D. R., Wang, B., Lopich, A., & Dudek, P. (2013). Mixed signal SIMD processor array vision chip for real-time image processing. *Analog Integrated Circuits and Signal Processing, 77*(3), 385–399.

Carey, SJ., Lopich, A., Barr, DR., Wang, B., Dudek, P. (2013b). A 100,000 fps vision sensor with embedded 535GOPS/W 256×256 SIMD processor array. In: IEEE Symposium on VLSI Circuits (VLSIC), pp 182–183

Castillo-Elizalde, H., Liu, Y., Bose, L., & Mayol-Cuevas, W. (2021). Weighted node mapping and localisation on a pixel processor array. In: IEEE international conference on robotics and automation (ICRA)

Chen, J., Carey, S., & Dudek, P. (2017a). Feature extraction using a portable vision system. In: vision-based agile autonomous navigation of UAVs Workshop, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)

Chen, Y. H., Krishna, T., Emer, J. S., & Sze, V. (2017). Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks. *IEEE Journal of Solid-State Circuits, 52*(1), 127–138.

Debrunner, T., Saeedi, S., & Kelly, P. (2018). AUKE: automatic kernel code generation for an analogue SIMD focal-plane sensor-processor array. *ACM Transactions on Architecture and Code Optimization (TACO), 1*(12), 1–25.

Debrunner, T., Saeedi, S., Bose, L., Davison, AJ., Kelly, PHJ. (2019). Camera tracking on focal-plane sensor-processor arrays

Dominguez-Castro, R., Espejo, S., Rodriguez-Vazquez, A., Carmona, R. A., Foldesy, P., Zarándy, Á., Szolgay, P., Szirányi, T., & Roska, T. (1997). A 0.8/spl mu/m CMOS two-dimensional programmable mixed-signal focal-plane array processor with on-chip binary imaging and instructions storage. *IEEE Journal of Solid-State Circuits, 32*(7), 1013–1026.

Dudek, P., & Hicks, P. J. (2005). A general-purpose processor-per-pixel analog SIMD vision chip. *IEEE Transactions on Circuits and Systems I: Regular Papers, 52*(1), 13–20.

El-Desouki, M., Deen, M., Fang, Q., Liu, L., Tse, F., & Armstrong, D. (2009). CMOS image sensors for high speed applications. *Sensors, 9*, 430–44.

Fischler, M. A., & Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM, 24*(6), 381–395.

Forster, C., Pizzoli, M., & Scaramuzza, D. (2014). SVO: Fast semi-direct monocular visual odometry. In: IEEE International Conferenc Robotics and Automation (ICRA), pp 15–22

Forster, C., Zhang, Z., Gassner, M., Werlberger, M., & Scaramuzza, D. (2016). SVO: semi-direct visual odometry for monocular and multi-camera systems. *IEEE Transactions on Robotics, 33*(2), 249–265.

Gallego, G., Delbruck, T., Orchard, GM., Bartolozzi, C., Taba, B., Censi, A., Leutenegger, S., Davison, A., Conradt, J., Daniilidis, K., & Scaramuzza, D. (2020). Event-based vision: A survey. IEEE Transactions on Pattern Analysis and Machine Intelligence pp 1–26

Greatwood, C., Bose, L., Richardson, T., Mayol-Cuevas, W., Chen, J., Carey, SJ., & Dudek, P. (2017). Tracking control of a UAV with a parallel visual processor. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp 4248–4254

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

Grupp, M. (2017). evo: Python package for the evaluation of odometry and SLAM. https://github.com/MichaelGrupp/evo

Handa, A., Newcombe, RA., Angeli, A., Davison, AJ. (2012). Real-time camera tracking: When is high frame-rate best? In: European Conference on Computer Vision (ECCV), Springer, pp 222–235

Hartley, R. I., & Zisserman, A. (2004). *Multiple View Geometry in Computer Vision*. Cambridge University Press.

Howard, AG., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., & Adam, H. (2017). MobileNets: Efficient convolutional neural networks for mobile vision applications. arXiv:1704.04861

Kim, H., Handa, A., Benosman, R., Ieng, SH., & Davison, A. (2014). Simultaneous mosaicing and tracking with an event camera. In: Proceedings of the British Machine Vision Conference, BMVA Press

Kim, H., Leutenegger, S., & Davison, AJ. (2016). Real-time 3D reconstruction and 6-DoF tracking with an event camera. In: European Conference on Computer Vision (ECCV), pp 349–364

Kitamura, K., Watabe, T., Sawamoto, T., Kosugi, T., Akahori, T., Iida, T., Isobe, K., Watanabe, T., Shimamoto, H., Ohtake, H., Aoyama, S., Kawahito, S., & Egami, N. (2012). A 33-megapixel 120-frames-per-second 2.5-watt CMOS image sensor with column-parallel two-stage cyclic analog-to-digital converters. *IEEE Transactions on Electron Devices, 59*(12), 3426–3433.

Klein, G., & Murray, D. (2007). Parallel tracking and mapping for small AR workspaces. In: IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR), pp 1–10

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

Leutenegger, S., Chli, M., & Siegwart, RY. (2011). BRISK: Binary robust invariant scalable keypoints. In: International conference on computer vision (ICCV), pp 2548–2555

Lichtsteiner, P., Posch, C., & Delbruck, T. (2008). A 128×128 120 dB 15 μs latency asynchronous temporal contrast vision sensor. *IEEE Journal of Solid-State Circuits, 43*(2), 566–576.

Likamwa, R., Hou, Y., Gao, Y., Polansky, M., & Zhong, L. (2016). RedEye: Analog ConvNet image sensor architecture for continuous mobile vision. International Symposium on Computer Architecture (ISCA) pp 255–266

Lin, X., Rivenson, Y., Yardimci, N. T., Veli, M., Luo, Y., Jarrahi, M., & Ozcan, A. (2018). All-optical machine learning using diffractive deep neural networks. *Science, 361*(6406), 1004–1008.

Linan, G., Espejo, S., Dominguez-Castro, R., & Rodriguez-Vazquez, A. (2002). Architectural and basic circuit considerations for a flexible 128×128 mixed-signal SIMD vision chip. *Analog Integrated Circuits and Signal Processing, 33*(2), 179–190.

Liu, Y., Bose, L., Chen, J., Carey, S., Dudek, P., & Mayol-Cuevas, W. (2020). High-speed light-weight cnn inference via strided convolutions on a pixel processor array. In: Proceedings of the British Machine Vision Conference (BMVC)

Liu, Y., Bose, L., Greatwood, C., Chen, J., Fan, R., Richardson, T., Carey, S. J., Dudek, P., & Mayol-Cuevas, W. (2021). Agile reactive navigation for a non-holonomic mobile robot using a pixel processor array. *IET Image Processing, 15*(9), 1883–1892.

Malis, E., & Vargas, M. (2007). Deeper understanding of the homography decomposition for vision-based control. Research Report RR-6303, INRIA

Martel, JN., Chau, M., Cook. M., & Dudek, P. (2015). Pixel interlacing to trade off the resolution of a cellular processor array against more registers. In: European Conference on Circuit Theory and Design (ECCTD), pp 1–4

Martel, J. N., Müller, L. K., Carey, S. J., Müller, J., Sandamirskaya, Y., & Dudek, P. (2017). Real-time depth from focus on a programmable focal plane processor. *IEEE Transactions on Circuits and Systems I: Regular Papers, 65*(3), 925–934.

McConville, A., Bose, L., Clarke, R., Mayol-Cuevas, W., Chen, J., Greatwood, C., Carey, S., Dudek, P., & Richardson, T. (2020). Visual odometry using pixel processor arrays for unmanned aerial systems in GPS denied environments. Frontiers in robotics and AI 7(126)

Mur-Artal, R., & Tardós, J. D. (2017). ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras. *IEEE Transactions on Robotics, 33*(5), 1255–1262. https://doi.org/10.1109/TRO.2017.2705103

Nistér, D. (2004). An efficient solution to the five-point relative pose problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 26*(6), 0756–0777.

Ojala, T., Pietikainen, M., & Maenpaa, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 24*(7), 971–987.

Olson, E. (2011). AprilTag: A robust and flexible visual fiducial system. In: IEEE International Conference on Robotics and Automation (ICRA), pp 3400–3407

Poikonen, J., Laiho, M., & Paasio, A. (2009). MIPA4k: A 64×64 cell mixed-mode image processor array. In: IEEE International Symposium on Circuits and Systems (ISCAS), pp 1927–1930

Reagen, B., Whatmough, P., Adolf, R., Rama, S., Lee, H., Lee, SK., Hernández-Lobato, JM., Wei. G., & Brooks, D. (2016). Minerva: Enabling low-power, highly-accurate deep neural network accelerators. In: ACM/IEEE Annual International Symposium on Computer Architecture (ISCA), pp 267–278

Rebecq, H., Horstschäfer, T., Gallego, G., & Scaramuzza, D. (2016). 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.

Rosin, P. L. (1999). Measuring corner properties. *Computer Vision and Image Understanding, 73*(2), 291–307.

Rosten, E., & Drummond, T. (2006). Machine learning for high-speed corner detection. In A. Leonardis, H. Bischof, & A. Pinz (Eds.), *Computer Vision-ECCV 2006* (pp. 430–443). Heidelberg: Springer, Berlin Heidelberg, Berlin.

Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. (2011). ORB: An efficient alternative to SIFT or SURF. In: International Conference on Computer Vision (ICCV), pp 2564–2571

Shafiee, A., Nag, A., Muralimanohar, N., Balasubramonian, R., Strachan, JP., Hu, M., Williams, RS., & Srikumar, V. (2016). ISAAC: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars. In: ACM/IEEE Annual International Symposium on Computer Architecture (ISCA), pp 14–26

Stow, E., Ahsan, A., Li, Y., Babaei, A., Murai, R., Saeedi, S., & Kelly, P. H. J. (2022). Compiling cnns with cain: focal-plane processing for robot navigation. *Autonomous Robots*. https://doi.org/10.1007/s10514-022-10053-w

Stow, E., Murai, R., Saeedi, S., & Kelly, P. H. J. (2022). Cain: Automatic code generation forÂ simultaneous convolutional kernels onÂ focal-plane sensor-processors. In B. Chapman & J. Moreira (Eds.), *Languages and Compilers for Parallel Computing* (pp. 181–197). Cham: Springer International Publishing.

Strasdat, H., Montiel, J. M., & Davison, A. J. (2012). Visual SLAM: Why filter? *Image and Vision Computing, 30*(2), 65–77.

Sturm, J., Engelhard, N., Endres, F., Burgard, W.,& Cremers, D. (2012). A benchmark for the evaluation of RGB-D SLAM systems. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp 573–580

Vicon, S. (1984). Vicon systems. Retrieved 2021-04-24 https://www.vicon.com/software/tracker/

Watanabe, Y., Oku, H., & Ishikawa, M. (2014). Architectures and applications of high-speed vision. *Optical Review, 21*(6), 875–882.

Wong, MZ., Guillard, B., Murai, R., Saeedi, S., & Kelly, PH. (2020). AnalogNet: Convolutional neural network inference on analog focal plane sensor processors. arXiv preprint arXiv:2006.01765

Zarándy, A. (2011). *Focal-plane sensor-processor chips*. New York: Springer.

Riku Murai received M.Eng in Computing in 2019 from the Imperial College London. He is currently a PhD student in the Department of Computing at Imperial College London. His research interests include robotics and computer vision. In particular, the use of novel hardware and distributed computations.

Sajad Saeedi is an Assistant Professor at Toronto Metropolitan University. He received his PhD in Electrical and Computer Engineering from the University of New Brunswick, Fredericton Canada. He is currently working on semantic perception, bringing deep learning advances to robotic systems. His research interests include robotic perception, collaborative robotic systems, and artificial intelligence.

Paul H. J. Kelly has been on the faculty at Imperial College London since 1989, has a BSc in Computer Science from UCL (1983) and has a PhD in Computer Science from the University of London (1987). He leads Imperial's Software Performance Optimisation research group, working on domain-specific compiler technology.