Towards Energy-efficient Visual Navigation:

Sensor Quantization and Event-based Vision Pipelines

by

Olivia Christie

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved April 2022 by the
Graduate Supervisory Committee:

Suren Jayasuriya, Chair
Chaitali Chakrabarti
Yezhou Yang

ARIZONA STATE UNIVERSITY

May 2022

ABSTRACT

Visual navigation is a useful and important task for a variety of applications. As the prevalence of robots increase, there is an increasing need for energy-efficient navigation methods as well. Many aspects of efficient visual navigation algorithms have been implemented in the literature, but there is a lack of work on evaluation of the efficiency of the image sensors. In this thesis, two methods are evaluated: adaptive image sensor quantization for traditional camera pipelines as well as new event-based sensors for low-power computer vision.

The first contribution in this thesis is an evaluation of performing varying levels of sensor linear and logarithmic quantization with the task of visual simultaneous localization and mapping (SLAM). This unconventional method can provide efficiency benefits with a trade-off between accuracy of the task and energy-efficiency. A new sensor quantization method, gradient-based quantization, is introduced to improve the accuracy of the task. This method only lowers the bit level of parts of the image that are less likely to be important in the SLAM algorithm since lower bit levels signify better energy-efficiency, but worse task accuracy. The third contribution is an evaluation of the efficiency and accuracy of event-based camera intensity representations for the task of optical flow. The results of performing a learning based optical flow are provided for each of five different reconstruction methods along with ablation studies. Lastly, the challenges of an event feature-based SLAM system are presented with results demonstrating the necessity for high quality and high-resolution event data. The work in this thesis provides studies useful for examining trade-offs for an efficient visual navigation system with traditional and event vision sensors. The results of this thesis also provide multiple directions for future work.

ACKNOWLEDGMENTS

I first want to thank my thesis advisor, Dr. Suren Jayasuriya, for his tremendous guidance and encouragement throughout the years of working with him. He has helped shape my experience and provided many amazing opportunities throughout my undergraduate and graduate years at ASU. I also want to thank both Dr. Chaitali Chakrabarti and Dr. Yezhou Yang for serving on my thesis committee.

I am also grateful to the FM&T team at Astrobotic including Andrew Horchler, Michael Bloom, and in-particular to Chris Owens for his mentorship during my internship at the company. I also want to thank fellow intern, Coleman Glagovich, for his contributions to the work on event-based vision in this thesis.

I am thankful to Victor Torres and Odrika Iqbal for their collaboration on the event-based vision projects and Joshua Rego for his assistance and collaboration with the first chapter of this thesis. In addition, I want to thank the whole Imaging Lyceum team for the positive impact they have had on me during my time with the lab.

TABLE OF CONTENTS

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Chapter 1

INTRODUCTION

Robotics is a growing industry that is useful for a wide variety of applications such as in manufacturing, assembly, transport, and earth and space exploration. As robots become more prevalent and advanced, there is an increasing need for them to be more energy-efficient as well. Traveling robots, or those that use portable batteries, are especially in need of low power solutions [56, 2]. One of the common applications of these types of robots is visual navigation. Visual navigation is the problem of a robot traversing an unknown area without any collisions with use of only a visual sensor. Small robots performing visual navigation are useful for both life-saving tasks such as search and rescue operations as well as fascinating tasks like exploration and everyday tasks like carpet cleaning. All of these systems have in common that they must perform their tasks within deadlines and under power constraints which are not optimal.

While a considerable amount of work has been done on increasing the efficiency of visual odometry algorithms themselves, not much attention has been placed towards the costs of visual sensing. In this thesis, we focus on two methods to aid in this goal: adaptive sensor quantization and novel event-based vision.

Current imaging pipelines for consumer photography produce high-quality images appealing for human vision. The image sensor processing pipeline (ISP) was designed for this purpose. The ISP processes the RAW data coming from the camera sensor to produce a finalized image, usually in PNG or JPEG format. The particular ISP stages vary between manufacturers, however most perform denoising, demosaicing, color transformations and gamut mapping, tone mapping, and compression.

The first chapter of this thesis aims to minimize the ISP pipeline for improved energy efficiency. A key previous paper showed that all of the ISP stages that are tuned for aesthetics are not necessary for many computer vision tasks [5]. This work provided inspiration for evaluating the particular task of visual simultaneous localization and mapping (SLAM) with data from a camera with the ISP removed. Varying bit levels of linear and logarithmic quantization will be evaluated revealing a trade-off between energy-efficiency and task accuracy. These results lead to our new gradient-based quantization scheme which further improves efficiency and performance. Image sensor quantization accounts for 50% of image sensing energy in modern CMOS image sensors [8], so reducing the bit levels yields significant opportunities for energy-efficiency in the pipeline.

Another method for energy-efficient visual navigation, besides changing a traditional image sensor, is to utilize a completely unique type of image sensor. Event cameras output brightness changes in the form of an asynchronous data stream of events, as compared to the discrete frames of traditional cameras. Events are composed of a timestamp, brightness change polarity number, and (x,y) pixel position corresponding to where the event occurred spatially. There are multiple benefits to this new type of camera including, high temporal resolution, high dynamic range, no motion blur, and a low bandwidth [17].

For the asynchronous data representation output from event cameras to be utilized, either new methods for completing vision tasks have to be implemented or the representation must be converted to a traditional frame-based form. Both of these methods are viable solutions that are active recent areas of research. The advantage of converting event data into frames is that the many years of previous work for traditional vision can be utilized. The disadvantage is that converting the event data into simple intensity frames based on a constant duration split reduces some of the benefits such as high temporal resolution and no motion blur. The second

part of this thesis focuses on the second method of converting event data into a traditional representation for the tasks of optical flow and SLAM.

1.1 CONTRIBUTIONS

This thesis focuses on the representation of input data for visual navigation pipelines and the trade-offs between energy-efficiency and task accuracy. The main contributions of this thesis are:

1. In Chapter 3, a study of linear and logarithmic quantization of RAW images with respect to localization accuracy and energy savings for visual SLAM
2. In Chapter 3, a new gradient-based quantization algorithm which quantizes the image spatially at various bit levels for feature-based visual SLAM algorithms
3. In Chapter 4, a study comparing event intensity reconstruction methods with respect to optical flow accuracy and efficiency
4. In Chapter 5, an analysis of estimating visual features in events for the task of visual SLAM

Chapter 2

BACKGROUND

In this chapter an overview of visual navigation and particular methods used in this thesis are provided. This chapter will be organized into three parts: visual features, tasks for visual navigation, and event-based cameras for low-power computer vision.

2.1 VISUAL FEATURES

Visual features are interest points in an image that can be used for a variety of computer vision tasks. These tasks include image alignment (homography), 3D reconstruction, motion tracking, object recognition, indexing and database retrieval, and navigation. Features are typically composed of two parts: a detector and a descriptor. The detector finds the location of the feature and the descriptor gives the feature a unique signature.

In order for a feature to be most useful, it needs to have some key properties. A feature needs to be invariant to transformations. This includes both geometric transformations such as translation, rotation, and scale changes as well as photometric transformations such as brightness and exposure changes. Features also need to be distinctive so that they can be differentiated and matched.

2.1.1 ORB

ORB (Oriented FAST and Rotated BRIEF), an efficient and high performing feature, is a fusion of the FAST (Features from Accelerated and Segments Test) keypoint detector and

BRIEF (Binary Robust Independent Elementary Feature) descriptor with performance enhancing modifications [45].

The FAST detector works by selecting a pixel p in an array and then comparing the brightness of that pixel p to the surrounding 16 pixels that are in a small circle around p. If there exists a set of 12 contiguous pixels in the circle which are all brighter or all darker than the intensity of p by some threshold T, then it is selected as a keypoint. ORB adds an orientation and scale component to these features. An image pyramid, a multiscale representation of a single image, is created and FAST keypoints are found at each level of the pyramid. ORB also assigns an orientation to each keypoint by computing the intensity weighted centroid of the patch around pixel p. The direction of the vector from this pixel p to centroid gives the orientation [44].

The BRIEF descriptor creates a signature in the form of a 256-bit binary vector from the keypoint found by the detector [7]. First the image is smoothed with a Gaussian kernel to add invariance to high-frequency noise. Then a pair of pixels are selected in a defined 31x31 patch around the keypoint. ORB chooses the pair of pixels using a learned greedy search. If the first pixel is brighter than the second, a value of one is assigned to the bit, if it is not brighter a value of zero is assigned:

$$\tau(\mathbf{p};\mathbf{x},\mathbf{y}) := \begin{cases} 1 & : \mathbf{p}(\mathbf{x}) < \mathbf{p}(\mathbf{y}) \\ 0 & : \mathbf{p}(\mathbf{x}) \geq \mathbf{p}(\mathbf{y}) \end{cases} \tag{2.1}$$

where p is the intensity. The feature is defined from 256 of these tests:

$$f_n(\mathbf{p}) := \sum_{1 \leq i \leq n} 2^{i-1} \tau\left(\mathbf{p};\mathbf{x}_i,\mathbf{y}_i\right) \tag{2.2}$$

ORB also adds a rotation invariant component to this process. The orientation of the keypoint is used to steer the descriptor. For any feature set of n binary tests at location (x,y), they define the 2 x n matrix:

$$\mathbf{S} = \begin{pmatrix} \mathbf{x}_1, \ldots, & \mathbf{x}_n \\ \mathbf{y}_1, \ldots, & \mathbf{y}_n \end{pmatrix} \tag{2.3}$$

Using the patch orientation and the corresponding rotation matrix, they construct a steered version of S:

$$\mathbf{S}_\theta = \mathbf{R}_\theta \mathbf{S} \tag{2.4}$$

So the operator becomes:

$$g_n(\mathbf{p}, \theta) := f_n(\mathbf{p}) \mid (\mathbf{x}_i, \mathbf{y}_i) \in \mathbf{S}_\theta \tag{2.5}$$

The angle is discretized, and a lookup table of precomputed BRIEF patterns is constructed [45].

2.2 OPTICAL FLOW

Optical flow is defined as the apparent motion of individual pixels on the image plane caused by relative motion between an observer and a scene. Computing optical flow involves the task of estimating per-pixel motion between two consecutive frames. Dense optical flow finds the displacement of all image pixels, while sparse optical flow finds the displacement of a sparse set of features. These displacements are used to calculate motion vectors. The key constraints of traditional optical flow include the need for brightness constancy, small motion between frames, and spatial coherence. The Kanade-Lucas-Tomasi method (KLT) solves the optical flow equations for all pixels in a neighborhood by using least squares. This method assumes that the displacement between two nearby frames is small and approximately constant within a spatial neighborhood [32, 52, 48]. Neural networks have superseded this more traditional method and are currently the state of the art method to compute optical flow. The first convolutional neural network (CNN) approach for optical flow was FlowNet. This method used a U-Net like architecture with encoder and decoder parts [16]. The current state-of-the-art optical flow approach is RAFT, which we will be introduce further in Chapter 4 [50].

2.3 SLAM

Simultaneous Localization and Mapping (SLAM) is a computational problem of constructing and updating a map of an environment, while at the same time keeping track of an agent's location within the map. Or in other words, given a series of observations, estimate the agent's location and a map of the environment. Acquiring a map while simultaneously localizing the position of a robot within this map can be a challenge because localization and mapping are interdependent. A visualization of the problem is shown in Figure 4.

There are many different variations of SLAM algorithms today. The three main paradigms include: filter-based approaches (Kalman filters, Particle filters, and Graph-based methods), global optimization approaches (ORB-SLAM), and learning-based approaches (RatSLAM) [51]. Within these categories, the most common types of sensors used are laser range sensors, cameras, and GPS.

2.3.1 Visual SLAM

Visual SLAM is a very popular type of SLAM due to its cost efficiency and simple sensor configuration. It is simple because cameras are the only sensor used. Cameras are one of the cheapest sensors available that have been used to directly perform SLAM. However, many different types of cameras can be used in visual SLAM at all price points. Some examples are simple monocular cameras (wide angle, fish-eye, spherical), compound eye cameras (stereo and multi), and RGB-D cameras (depth and ToF).

Most visual-based SLAM includes two main components: the front-end and the back-end. The front-end abstracts the sensor data into a form that can be used for the back-end. The back-end performs inference on the abstracted data and computes the position and map.

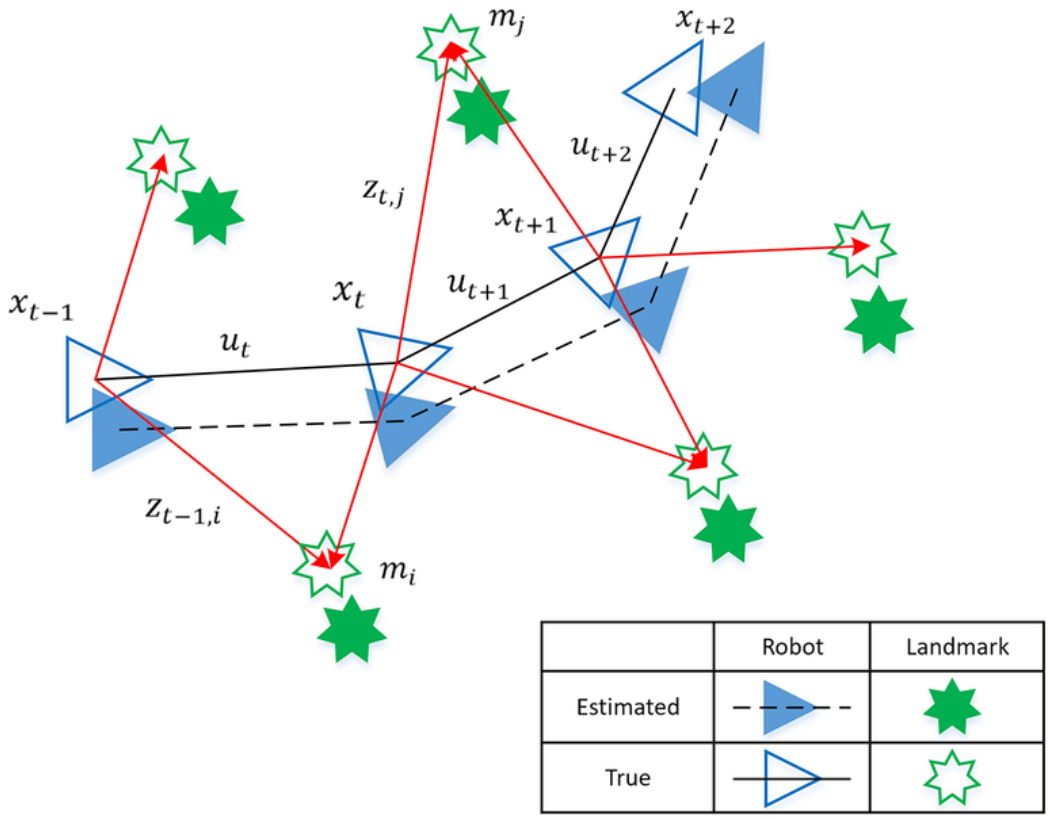


Figure 1. Schematic diagram of Simultaneous Localization and Mapping. Figure from [27]

Visual SLAM can be split into two trade-off types: sparse or dense and feature-based or direct. Sparse SLAM systems use only a subset of all sensor information (pixels) available, while dense SLAM systems use most or all of the sensor information (pixels) from each image frame. Because of this property, the map generated from a sparse SLAM will be a scarce map or point cloud, while that from a dense SLAM constructs a full set of points (not sparse), aiming to complete every pixel or voxel of the target space. Dense SLAM systems require more powerful computers to run the algorithms because they use more information. Feature-based SLAM includes a step of extracting features from the input images and then matching those features to solve the SLAM problem. Conversely, direct SLAM methods directly use the input image without any abstraction. Photometric consistency is used as an error metric for direct methods, while geometric consistency of the feature points is used for feature-based methods.

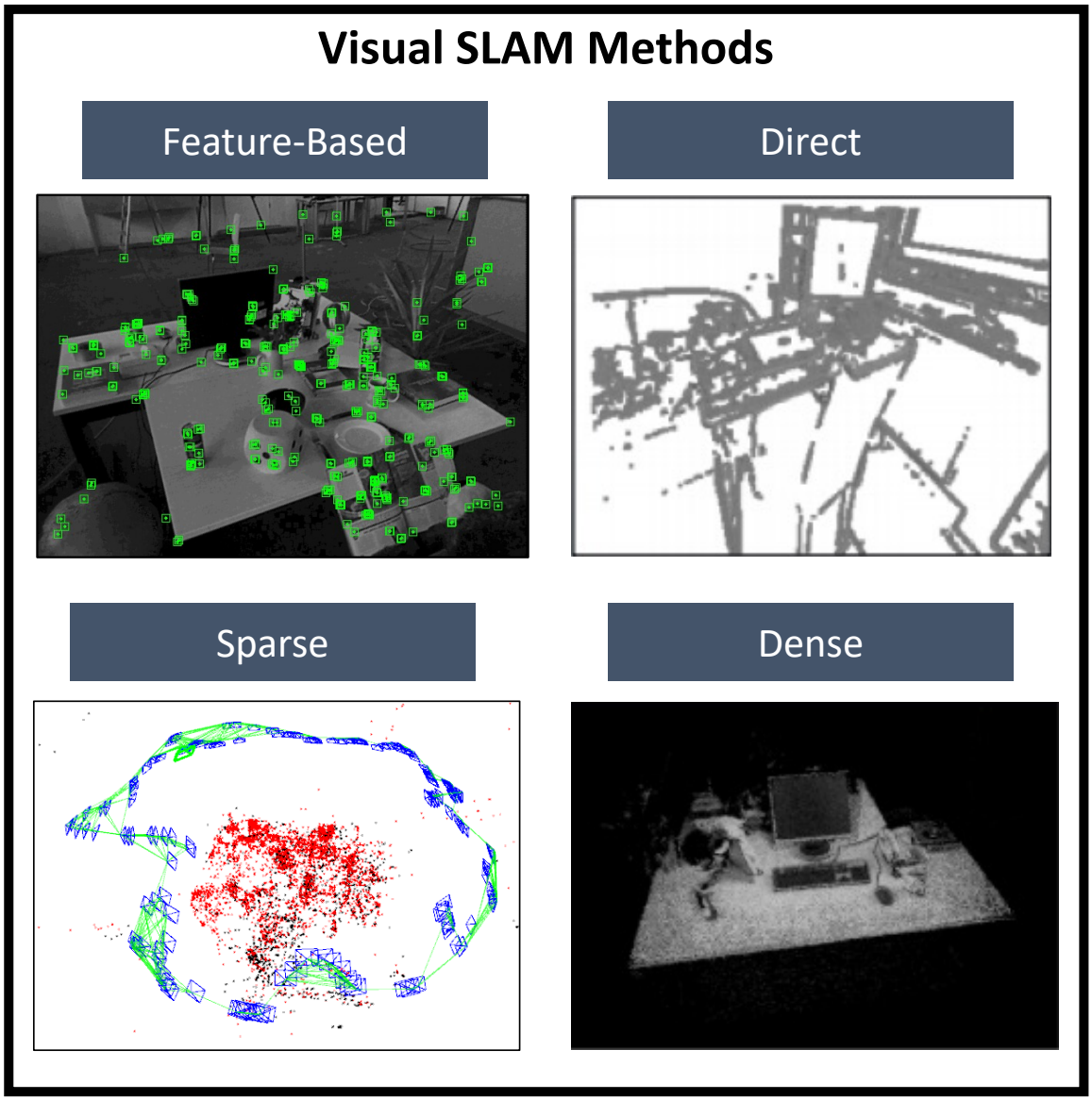


Figure 2. The main categories of visual SLAM methods: feature-based or direct estimation and sparse or dense map generation.

2.3.2 ORB-SLAM2

ORB-SLAM2 is an open-source visual sparse feature-based SLAM system that can be performed with either a monocular, stereo, or RGB-D camera [38]. In this thesis, we will concentrate on the monocular ORB-SLAM2 version as that is the most cost-efficient.

ORB-SLAM2 performs four main tasks in parallel: tracking, mapping, re-localization, and loop closing. ORB-SLAM2 is a feature-based algorithm, which means that it detects features with ORB (Oriented FAST and rotated BRIEF) and then uses these features to estimate the location and map. It is highly robust and compact due to a method of picking only certain features and keyframes for reconstruction. This feature-based algorithm is simple and light because it does not store all the positional information. ORB-SLAM2 operates in real-time which is essential for autonomous control of a robot [38].

The main problem of visual SLAM is reconstructing the 3D-environment from the 2D-images. There are three main components involved in solving this problem with a feature-based SLAM.

Features: The first step is to detect features in images using a feature detector and descriptor. ORB-SLAM2 uses the ORB detector and descriptor. ORB is a modified version of the BRIEF (Binary Robust Independent Elementary Features) descriptor and FAST (Features from Accelerated Segment Test) detector. These features were chosen because they are very fast to compute and match but are still invariant to viewpoint [45].

Keyframes: The second step of visual SLAM is to select certain frames called keyframes. Monocular SLAM was previously performed by processing every frame collected from the camera. However, when the camera has not moved there is no new information and so it is not necessary to perform SLAM on all of these frames.

The algorithm performs a survival-of-the-fittest technique in choosing keyframes which

provides unprecedented robustness. The algorithm inserts many keyframes quickly, and then later culls the frames by removing redundant ones. Certain keyframes are selected only when the scene has changed. The computations are then only performed on these keyframes, instead of all of the images captured. This reduces the computational effort of processing consecutive frames with no new information.

Bundle Adjustment/Loop Closure: The last step of the ORB-SLAM algorithm is to perform bundle adjustment and loop closure to refine the visual reconstruction of the 3D structure and viewing parameter estimates. Loop closure occurs when the current path has already been seen. The map is then updated with the correctly associated data information. A bag of words technique is used in ORB-SLAM to determine if the current image frame is similar to a frame already captured.

2.3.3 LSD-SLAM

LSD-SLAM is a semi-dense direct-based SLAM system which utilizes image intensities to estimate location and a semi-dense depth map. It is composed of three main parts: tracking, depth map estimation, and map optimization. The depth map is only created for pixels around large image intensity gradients [14].

2.4 EVENT CAMERAS

In 1991, Misha Mahowald, a graduate student at Caltech, along with Prof. Carver Mead created a "Silicon Retina" which mimicked the neural architecture of an eye [33]. This sparked work on neuromorhpic engineering and event-based cameras.

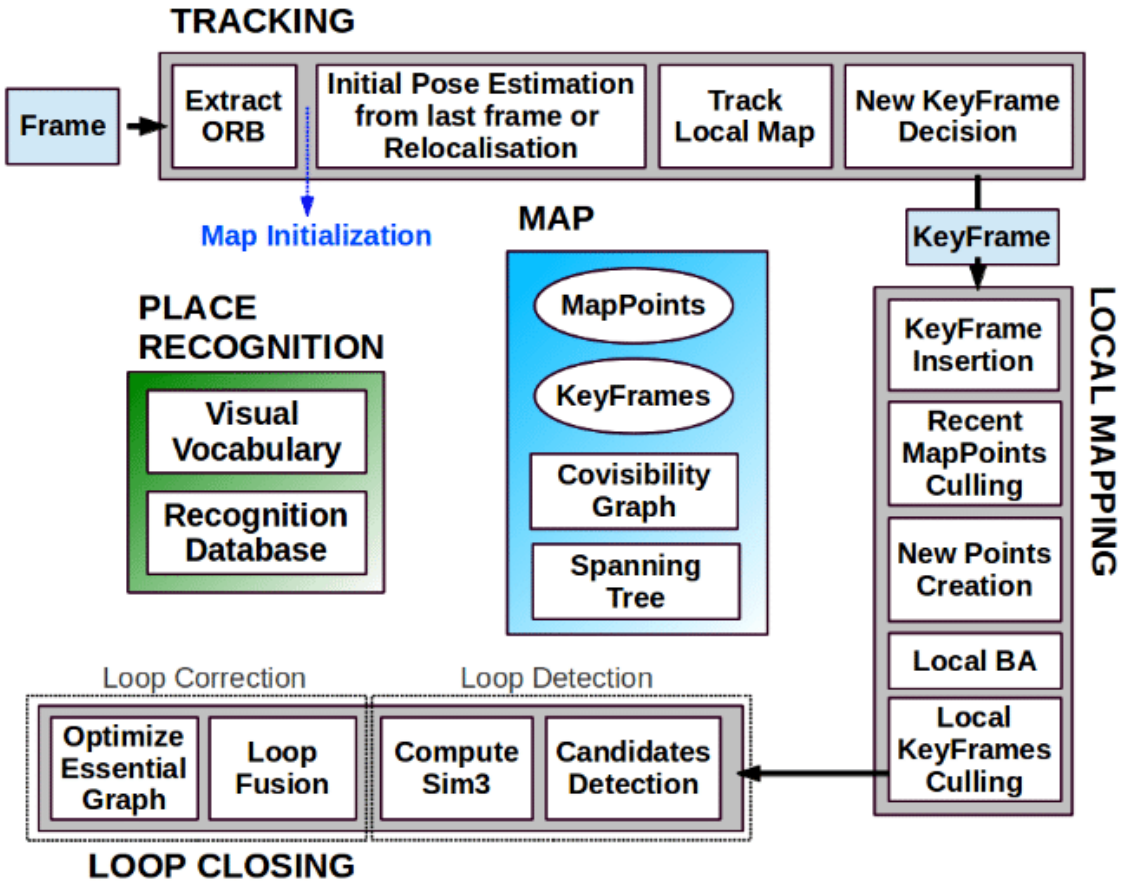


Figure 3. Overview of the ORB-SLAM2 system. Figure from [38]

Event cameras have some unique properties due to their being modeled directly after the human eye. These cameras measure only changes in light intensity. Events are composed of a timestamp, brightness change polarity number, and x, y position corresponding to where the event occurred spatially. Some of properties of event-cameras include a low-latency on the order of microseconds, no motion blur, high dynamic range (140 dB), and ultra-low power (1mW). Event cameras are advantageous for real-time interaction systems like robotics and wearable electronics where there can be a need of operation in low lighting, power, and latency [17].

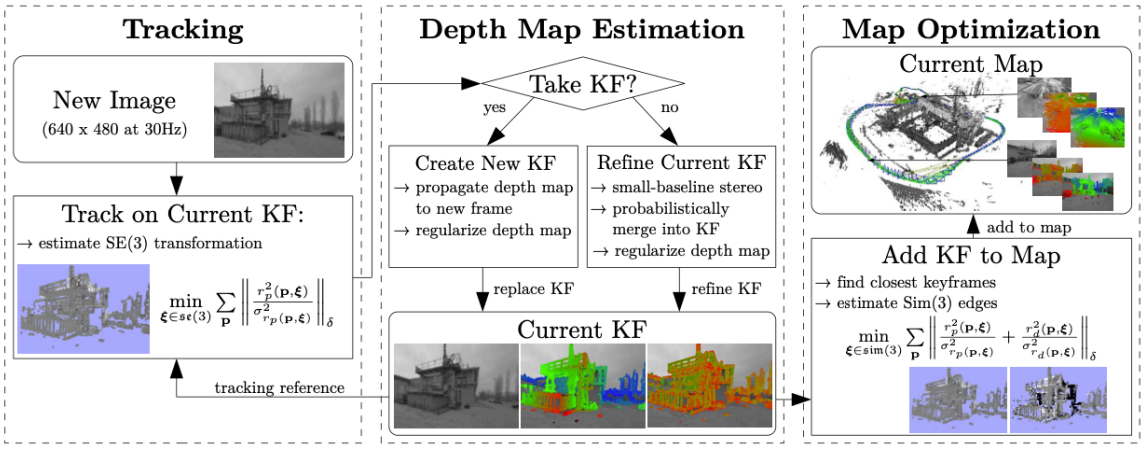


Figure 4. Overview of LSD-SLAM system. Figure from [14]

2.4.1 Voxel Grids

Zihao Zhu et al. introduced an event representation to maintain more information as an input for neural networks [59]. The time domain is discretized and events are inserted into the volume using linearly weighted accumulation similar to bilinear interpolation. Given a set of

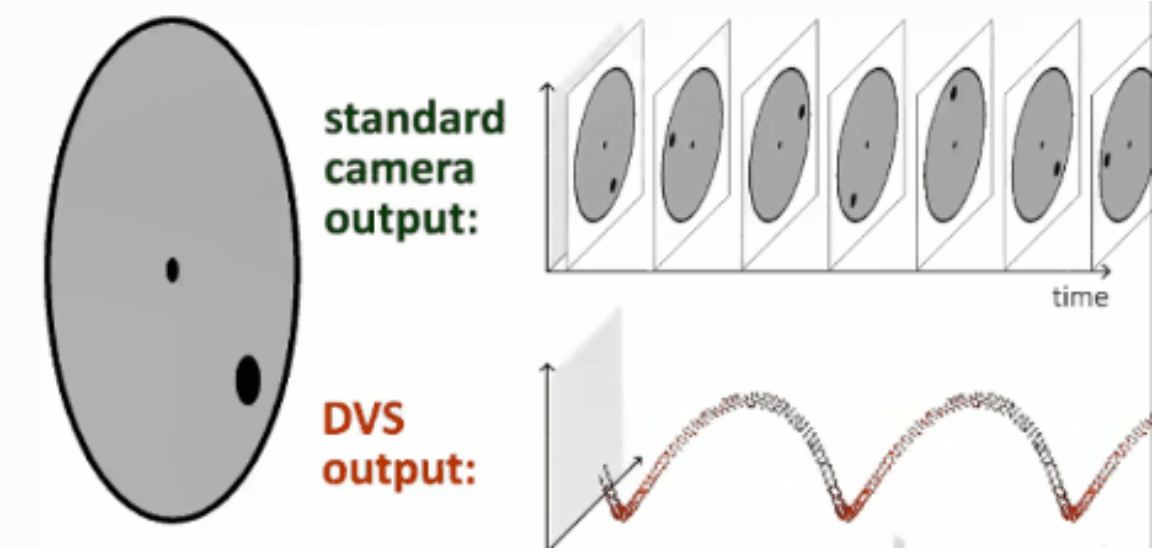


Figure 5. Comparison of operation of an event camera vs a standard camera. The event camera only outputs events when there is a change in brightness. Figure from Davide Scaramuzza

N input events and a set B bins to discretize the time dimension, they scale the timestamps to the range $[0, B-1]$, and generate the event volume as follows:

$$
\begin{aligned}
t_i^* &= (B-1)\,(t_i - t_0)\,/\,(t_N - t_1) \\
V(x,y,t) &= \sum_i p_i k_b\,(x - x_i)\,k_b\,(y - y_i)\,k_b\,(t - t_i^*) \\
k_b(a) &= \max(0, 1 - |a|)
\end{aligned}
\tag{2.6}
$$

where $k_b(a)$ is equivalent to the bilinear sampling kernel defined in Jaderberg et. al. [21].

Chapter 3

ANALYZING SENSOR QUANTIZATION OF RAW IMAGES FOR VISUAL SLAM

In this chapter, various quantization schemes are evaluated along with resolution and frame-rate ablation studies to introduce an energy-efficient imaging pipeline for visual navigation [1].

3.1 MOTIVATION

Simultaneous localization and mapping (SLAM) is one of the most critical algorithms for robotic and embedded platforms performing navigation in the real world. SLAM, using a combination of visual, inertial, and depth sensors, determines a map of the robot's environment while localizing or identifying the position/pose of the robot within that map. However, the energy costs of running SLAM on real-time, mobile platforms can be expensive, limiting battery life for these devices in the wild. Thus, it is important to find energy-efficient pipelines for SLAM that can still obtain good accuracy and performance to enable ubiquitous robotic navigation.

Previous research for energy-efficient SLAM has utilized techniques such as motion planning and dynamic power management [34]. Real-time SLAM systems leverage efficient feature detection and description, local tracking and mapping, and parallel thread computing for fast performance [37]. Most of these approaches have concentrated on increasing computational efficiency after receiving sensor data.

Monocular visual SLAM is an emerging algorithm which has reduced the number and types of sensors necessary to a single visual camera, and has shown good localization results [37,

[1] This work was originally presented at the 2020 IEEE International Conference on Image Processing [10].

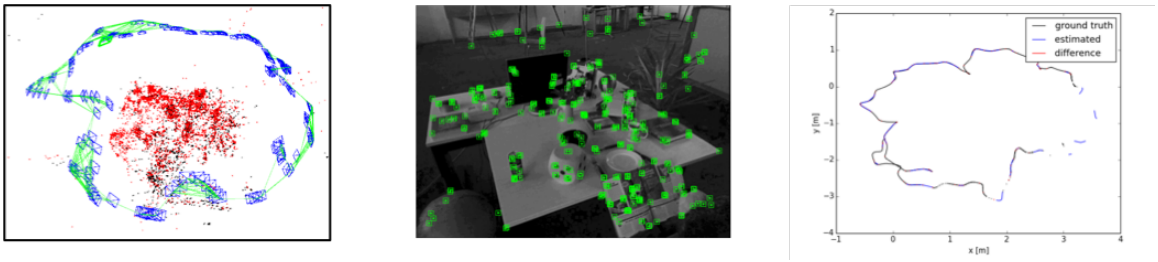
38]. Advantages include being lightweight in the hardware, applicable for mobile cameras and embedded platforms with low size, weight and power (SWaP). However, not much research has looked at the energy costs of image sensing itself for visual SLAM. In particular, image sensor processing (ISP) pipelines which convert RAW images to JPG/PNG images are normally tuned for creating highly aesthetic and visually pleasing images. It is unknown if this processing is needed for machine vision algorithms such as SLAM, and what optimizations can improve energy-efficiency without sacrificing accuracy. For instance, reduced ISP pipelines have been shown to save up to 75% of image sensing energy for other vision tasks [5].

In this chapter, we investigate the effectiveness of visual SLAM on RAW images, without ISP processing, at varying types and levels of quantization. Image sensor quantization accounts for 50% of image sensing energy in modern CMOS image sensors [8], yielding significant opportunities for energy-efficiency in the pipeline. Our specific contributions include: (1) comparing linear and logarithmic quantization of RAW images with respect to localization accuracy for visual SLAM, and (2) introducing a new gradient-based quantization algorithm which quantizes the image spatially at various bit levels that outperforms both linear and logarithmic quantization for feature-based visual SLAM algorithms. We validate these contributions by testing two state-of-the-art visual SLAM algorithms on seven video datasets. This, to the best of our knowledge, is the first study to explore visual SLAM performance on RAW and varying quantized images.

3.2 BACKGROUND

The image signal processing (ISP) pipeline is a method which converts the RAW data from a camera into a digital image form. There are multiple steps involved in this process as shown in Figure 7 including: white balancing, demosaicing, denoising, color transforms, tone

(a)



(b)

Figure 6. (a) Experimental pipeline for analyzing quantization for Visual SLAM. The original dataset is run through CRIP to get RAW quantized images that are used for both SLAM methods. (b) ORB-SLAM2: Left - Scene mapping and camera trajectory. Center - Feature detection for a single video frame. Right - Output camera trajectory compared to the ground truth.

reproduction, and compression. A RAW image is usually mosaiced and 12 bit. Mosaicing is performed with a Bayer filter, a color filter array which arranges RGB colors on the photosensors. Demosaicing combines these separate colors into a traditional three channel RGB image. White balancing, denoising, and color transforms are all processes to make the images visually look better. Compression transforms a high bit value image into a lower bit value image. This reduces the cost of storage and transmission of the data.

3.3 RELATED WORK

Simultaneous localization and mapping (SLAM) has been an active area of research for over 30 years [12, 1], with recent advances in monocular visual SLAM algorithms [37, 38, 14,

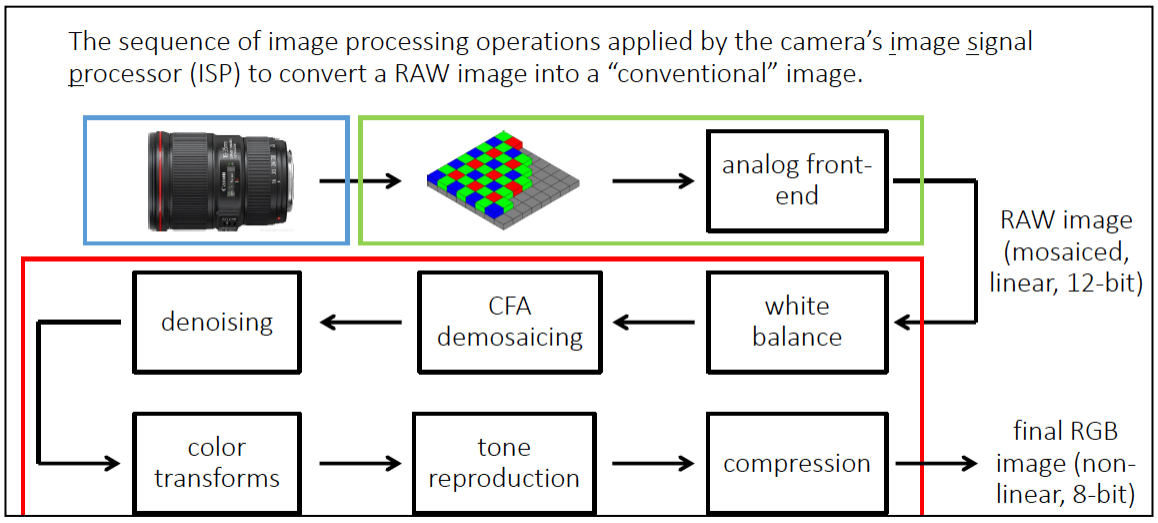


Figure 7. Image signal processing (ISP) pipeline. Figure courtesy of Ioannis Gkioulekas

23, 13]. For energy-efficient SLAM, eSLAM achieves real-time performance on low-power platforms by optimizing feature extraction and matching, yielding $41 - 71\times$ energy improvements and $1.7 - 3\times$ frame rate speed up [30]. Further, hardware acceleration such as a FPGA-based ORB feature extractor for SLAM reduced the energy consumption by 83% and reduced the latency by 41% compared to an Intel i5 CPU [15]. We are primarily concerned with optimizing the image sensing energy prior to the visual SLAM algorithm, and our methods are complementary with these systems.

The image sensor processing (ISP) pipeline utilizes demosaicing, denoising, color transforms, white balancing, and tone mapping to achieve high quality aesthetic images. However, for energy-efficiency, some smartphone cameras can bypass the ISP to produce RAW images. Liu et al. proposed an ISP that selectively disables stages depending on application needs [31]. The work most aligned with ours concerns reconfigurable ISP pipelines for energy-efficient computer vision [5]. This work shows how reduced ISP pipelines lead to vision accuracy-energy tradeoffs and save 75% of sensing energy with a minimalistic pipeline using logarithmic quantization. In our work, we leverage these insights and apply them to the particular case

of visual SLAM. We introduce a new spatially-varying quantization method to improve the performance of visual SLAM over logarithmic quantization.

3.4 METHOD

3.4.1 Simulating RAW Data:

The main challenge to evaluating the effectiveness of Visual SLAM on RAW data is the availability of suitably labeled datasets at varying quantization. We leverage the Configurable & Reversible Imaging Pipeline (CRIP) from [5] which can reverse JPEG/PNG images back to RAW format. CRIP was shown to have average pixel error of 1.064% and the PSNR was 28.81 dB as compared to real RAW images [5], lending confidence to the validation of our algorithms on this data. Using CRIP, we can convert visual SLAM data available online. For energy-efficiency, we turn off the ISP, including demosaicing, denoising, white balancing, color transforms, and tone mapping. This allows the sensor hardware to go straight from image sensor ADC to the SLAM algorithm, eliminating the ISP chip. Since visual SLAM, especially real-time systems, commonly work in grayscale intensity, most ISP optimizations are not critical. We chose to not include color stages as the visual SLAM algorithms work on grayscale intensity for real-time performance benefits. Images were taken in normal light conditions, and thus denoising was not necessary. For tone mapping, we utilize the fact that logarithmic quantization approximates the benefits of tone mapping [5] to avoid this stage. We also tested some data with and without demosaicing to investigate whether visual SLAM's effectiveness is affected by the Bayer color filter on the sensor inducing intensity changes on the original mosaiced images. In addition to this pipeline, resolution and frame rate were explored, however these could be simulated without the need for CRIP.

3.4.2 Linear and Logarithmic Quantization.

The image sensor analog-to-digital (ADC) converters operate on each pixel, and the typical linear ADC's energy cost is exponential in the number of bits in its output. Thus, image capture energy can be reduced via lower bit depths in sensor quantization, which can be achieved via successive-approximation (SAR) ADCs [53]. Quantization can be either linear or nonlinear. The nonlinear distribution of quantization levels can better represent images as the non-uniform probability distribution function for intensities in natural images is log-normal [43]. A central insight of Buckler et al. [5] was that log quantization uniformly mapped this distribution to equal bit values, thus performing approximate tone mapping of the images without the ISP. This yields beneficial accuracy/energy trade-offs across several computer vision benchmarks. In our experiments, we test the effectiveness of uniform linear and logarithmic quantization from 8 bits down to 2 bits for visual SLAM.

3.4.3 Gradient-based Quantization.

In addition to linear and logarithmic quantization, we introduce a new form of quantization based on image gradients to help improve the accuracy-energy tradeoff. Our algorithm encodes regions with high-intensity gradient with higher bit values and lower gradient regions with lower values. Since most visual features contain gradient energy, this method preserves these features while downgrading non-salient regions at low bits. This yields significant energy savings in average bit depth across an image.

Gradient-based quantization relies on sensing the image gradient for pixels locally, and could theoretically be implemented in image sensor hardware. Focal-plane processing can compute basic functions such as edge detection and gradients in analog on the sensor [11], as

well as optical pixels including Angle Sensitive Pixels [9] and event-based sensors [29]. While there is potential to implement this in hardware, for this study, this method is simulated using the preprocessed images for each quantization level.

Our algorithm is the following:

$$I_{GQ}[m,n] = I_{b[m,n]}[m,n], \tag{3.1}$$

$$b[m,n] = \min(\lceil \frac{W[m,n]}{\max(W[m,n])} * 7 \rceil + b_{min}, b_{max}). \tag{3.2}$$

where $W[m,n] = \sum_{(i,j)\in N(m,n)} \nabla I_{RAW}$ is the total gradient energy of a neighborhood $N(m,n)$ around pixel (m,n), ∇I_{RAW} is the image gradient magnitude, $b[m,n]$ is the bitmap which maps a pixel to a quantization bit depth, and $I_b[m,n]$ is the corresponding logarithmic quantized pixel at that bit depth. We use the gradient of the image using a 5×5 kernel. We use a 3×3 neighborhood, and shift all pixels between 3 and 8 bits precision using $b_{min} = 3, b_{max} = 8$. In Figure 8, we show an example frame which has been quantized using our method. The red inlet shows an area where high gradient intensity is mapped to higher bit quantization, the green inlet surrounding the dice shows edge information, which is a mix of high and low quantization, and the blue inlet surrounding the floor with low gradient intensity is mapped to lower bit quantization.

3.4.4 Visual SLAM benchmarks.

We deploy two benchmarks for Visual SLAM: ORB-SLAM2 [38], and LSD-SLAM [14], both of which are open-source real-time monocular SLAM systems. ORB-SLAM2 is a feature-based algorithm which detects features with ORB features, and then estimates the location and sparse depth map based on these features. ORB-SLAM2 performs four main tasks in parallel:

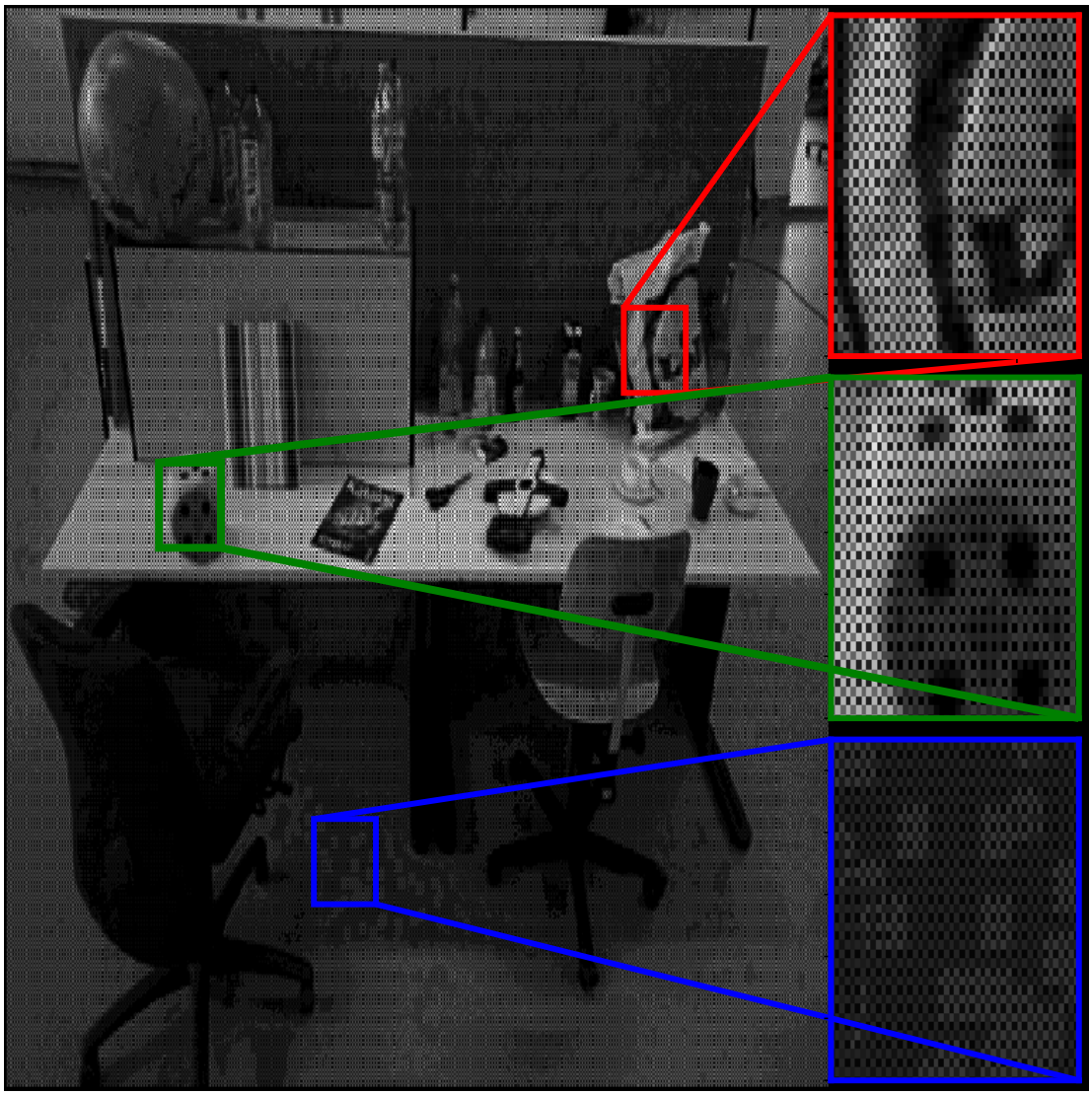


Figure 8. Gradient-based quantization frame. The three inlets show differences in bit range with red (high gradient), green (mix of high and low), and blue (low gradient).

tracking, mapping, re-localization, and loop closing. It is highly robust and compact via careful selection of only certain features and keyframes for reconstruction [38]. LSD-SLAM is a direct-based algorithm which utilizes the image intensities to estimate the location and semi-dense depth map. It is composed of three main parts: tracking, depth map estimation, and map optimization. The depth map is only created for pixels around large image intensity gra-

dients [14]. We chose these two benchmarks as exemplars for feature-based and direct-based Visual SLAM algorithms.

3.5 EXPERIMENTAL RESULTS

3.5.1 Dataset and Metrics.

The TUM RGB-D benchmark dataset [49] was used to evaluate the accuracy of camera localization while running our imaging pipelines. This dataset provides sequences along with ground truth trajectory obtained with an external motion capture system. We utilize 7 videos from this dataset for our experiments, which although is smaller in scale, is on roughly the same order of videos evaluated as compared to the original ORB-SLAM [37].

Our error metric is absolute trajectory error (ATE) defined as the difference between points of the true and the estimated trajectory [49]. The true and estimated poses are matched via timestamps and then aligned using a similarity transform [19], as the scale of monocular SLAM is unknown. Then ATE is calculated as a root mean squared error.

While computational speed is another important metric, we do not report latency as we found that the per-frame processing time for all pipelines were roughly the same at 21-33ms on average. To quantify the expected energy savings of our imaging pipeline, we follow the model of [5] to compute the expected value of the ADC energy readout.

$\mathrm{E}\left[\mathrm{ADC}_{-\mathrm{energy}}\right] = \sum_{m=1}^{2^n} p_m e_m$ where n is the number of bits, $2n$ is the total number of levels, m is the level index, p_m is the probability of level m occurring, and e_m is the energy cost of running the ADC at level m.

3.5.2 Initialization, Tracking and Features.

Visual SLAM algorithms can suffer from issues with initializing at the beginning of the video, as well as maintaining tracking over the entire video. We observed that the number of features extracted while performing ORB-SLAM2 affects the performance of our quantization pipeline. As the bit level decreased in our quantization pipelines, the features were increased in order to preserve fast initialization and accurate tracking. We found that 4 bit linear and logarithmic quantization required an increase of four-hundred features, and logarithmic quantization lower than four bits required an increase of two-hundred features per bit level.

3.5.3 Analysis

Resolution. For ORB-SLAM2, we simulated resolutions ($640 \times 480, 533 \times 400, 427 \times 320, 320 \times 240$) with ATE errors (0.29 cm, 3.48 cm, 6.46 cm, 10.9 cm) respectively for the fr2-xyz video. For the same video with the same resolutions, LSD-SLAM reported ATE errors (3.44 cm, 3.89 cm, 3.87 cm, 7.95 cm). We found similar trends for other datasets, but both SLAM algorithms failed to initialize and track below a resolution of 320×240. These experiments demonstrate that as the resolution decreases, visual SLAM accuracy degrades until it does not track for low resolutions. This means that image sensor subsampling such as windowing, ROIs, or binning would not be effective for these visual SLAM algorithms.

Frame Rate. We simulated frame rates of 30 FPS, 15 FPS, and 7.5 FPS by subsampling frames. For ORB-SLAM2 on two example videos, the ATE was constant down to 7.5 FPS. Below 7.5 FPS, however, it either failed to track or the error increased significantly. LSD-SLAM was sensitive to lower frame rates, with failure to initialize and track after 15 FPS. We

observed similar trends in other videos. We hypothesize that low FPS causes feature matches to be more distant in time due to the frame subsampling, causing tracking issues.

Quantization. Since quantization is one of our primary mechanisms for saving energy in an image sensor, we conducted extensive tests on three types of quantization: (1) linear quantization, (2) logarithmic quantization, and (3) our new gradient-based quantization. For each of these quantization, we converted images to RAW and then quantized with varying the number of bits from 8 down to 2 using the CRIP pipeline.

Linear quantization: Linear quantization resulted in an increasing error trend with ORB-SLAM2, see Figure 9. The results show an average ATE of 7.52 cm at eight bits, 11.2 cm at seven and six bits, 4.23 cm at four bits, and 23.51 cm at the lowest working bit level of four bits. The average ATE was taken over four videos as two videos failed to initialize and track and one video resulted in very high error. None of the LSD-SLAM videos that were linearly quantized were able to initialize and track. To analyze this, it is helpful to look at the logarithmic quantization results for LSD-SLAM to draw comparisons.

Logarithmic quantization: In Figure 11, we show the average ATE results over all datasets for our logarithmic quantization pipelines. ORB-SLAM2 was generally more robust to logarithmic quantization and shows an expected trend of increasing ATE as the bit value decreases, with a low ATE of 1.70 at 8 bits that increases to 4.23 at 4 bits, then jumps to 14.22 at 3 bits. Logarithmic quantization outperformed linear quantization because of the approximate tone mapping effect that occurs due to the statistics of pixel values in natural images (which was also observed in [5]).

The results for LSD-SLAM, shown in Figure 12, are less consistent and in general, show poor performance for any RAW image pipeline. The minimum ATE achieved was 47.37 at 4 bits while the maximum ATE of 82.08 occurred at 3 bits. These averages are much higher for

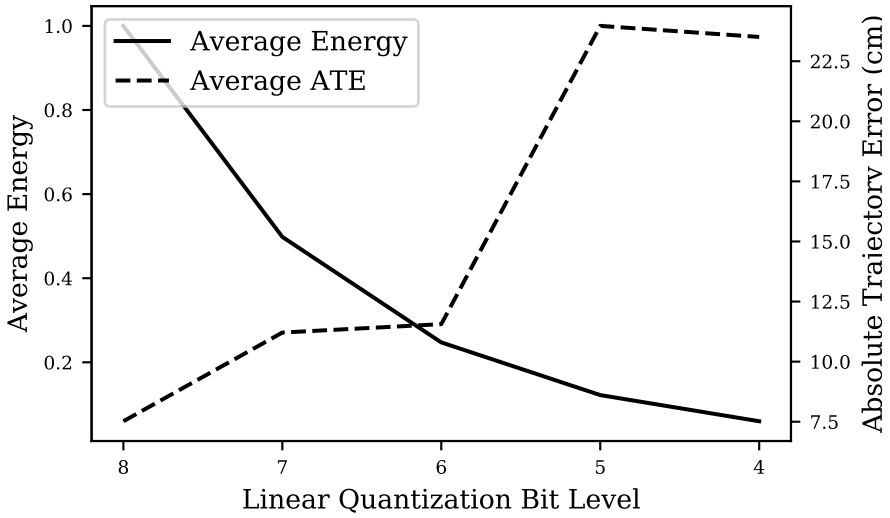


Figure 9. Relative average ADC energy savings normalized to 8 bits and average ATE for ORB-SLAM2 over seven datasets with 4-8 bit linear quantization

each pipeline than those measured for ORB-SLAM2. However, we note that log quantization still outperforms linear quantization, which failed to initialize.

We believe there are two mechanisms at play for the performance of LSD-SLAM. First, the approximate tone mapping of log quantization affects image intensities and contrast, and thus enables an intensity-based method like LSD-SLAM [14] to perform better with log quantization than linear quantization. However, even in the log quantized RAW images, the Bayer pattern likely causes false textures to appear in flat regions, and causes LSD-SLAM errors. We note that although we are operating with no ISP in this chapter, we did try demosaicing on log quantized images and were able to achieve a more consistent performance for LSD-SLAM.

Gradient-based quantization: For ORB-SLAM2, our gradient-based quantization algorithm leads to further gains in energy efficiency. As shown in Table 1, the average bit value of each video is consistently between 4 and 5 bits with an overall average of 4.41 bits. Even with this relatively lower bit average of the images, visual SLAM achieves an average ATE of 1.81.

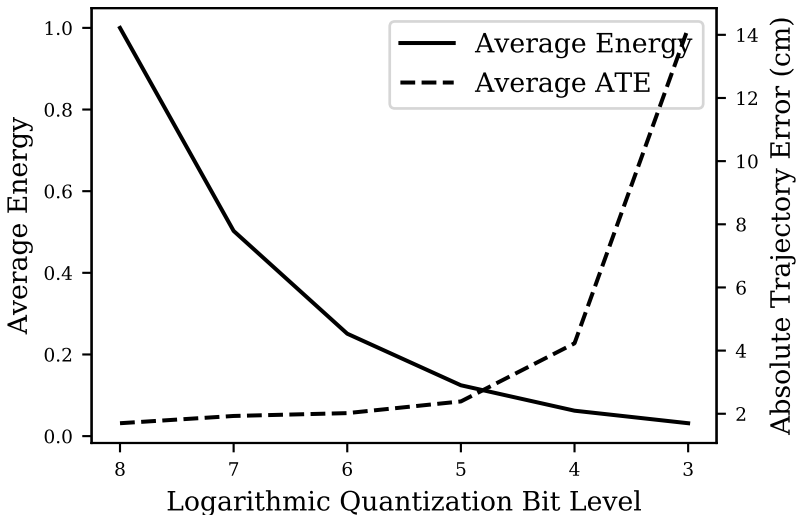


Figure 10. Relative average ADC energy savings normalized to 8 bits and average ATE for ORB-SLAM2 over seven datasets with 3-8 bit logarithmic quantization.

This ATE is comparable to 7 and 8 bit logarithmic quantization pipelines with ATEs 1.93 and 1.70 respectively, saving effectively 3-4 bits in energy.

We note that the average bit level is low because a majority of pixels contain flat gradient information. Edges or high gradient pixels are a much sparser set of the total. Keeping only these high gradient pixels at higher bit ranges allows for features to still be detected easily while saving energy. This method shows promise as a balanced pipeline for performance and energy efficiency.

For LSD-SLAM, we do not see similar benefits for gradient-based quantization, like our previous quantization experiments. Since LSD-SLAM does not use features but rather intensity differences, including high bit regions in an otherwise low bit quantized image does not improve the performance as well as it does for feature-based methods.

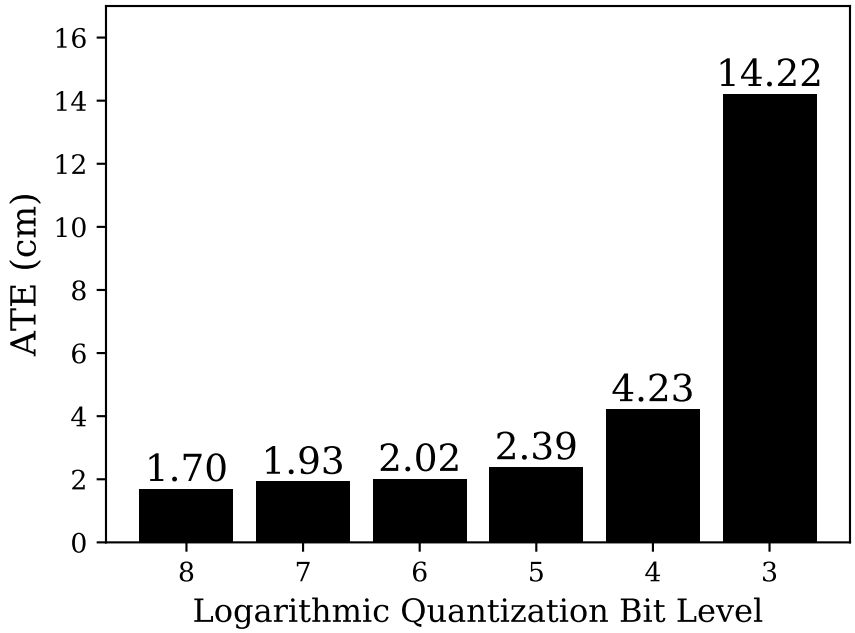


Figure 11. Average ATE for logarithmic quantization for ORB-SLAM2.

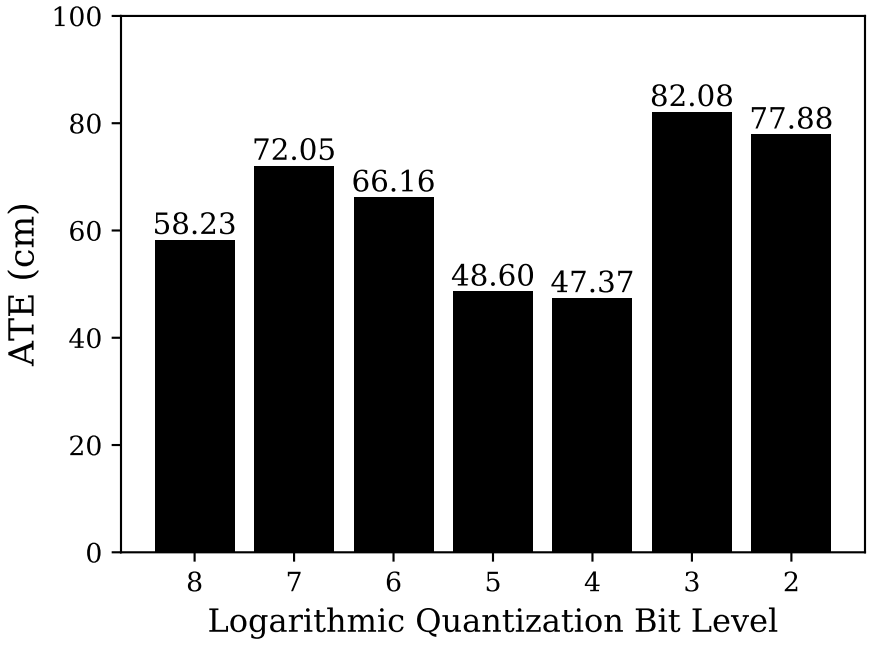


Figure 12. Average ATE for logarithmic quantization for LSD-SLAM.

	fr1 xyz	fr2 xyz	fr1 floor	fr1 desk	fr2 desk	fr3 long office	fr2 desk person	Avg.
Avg. bit	4.55	4.42	4.31	4.55	4.38	4.23	4.42	**4.41**
ATE (ORB)	1.04	0.28	2.9	1.8	0.82	4.9	0.94	**1.81**
ATE (LSD)	4.06	3.72	71.1	69.9	88.7	159.4	45.1	**63.1**

Table 1. Gradient-based Quantization Results

3.6 DISCUSSION

Visual SLAM was investigated on RAW images without ISP processing and varying sensor quantization. The results indicate that for feature-based visual SLAM algorithms, namely ORB-SLAM2, using RAW images with logarithmic quantization at low bit levels can be energy-efficient and high performing. In particular, the novel gradient-based quantization algorithm achieved effectively 3-4 bits in energy savings without sacrificing performance. However, we note that our results on LSD-SLAM are not as conclusive since the intensity-based SLAM method does not rely on feature mapping. It remains as future work to try and adapt sensor quantization schemes that can benefit these direct-mapping methods. It would also be of interest to test our methods on a deep learning SLAM algorithm like DeepSLAM [28]. Also, there is an opportunity to optimize the SLAM algorithm itself for RAW data to extract the maximum performance while maximizing energy-efficiency.

Chapter 4

INTERMEDIATE EVENT REPRESENTATIONS FOR COMPUTING OPTICAL FLOW

In this chapter, we will analyze using an event-based camera for energy-efficient visual navigation. The event-based camera is a highly efficient sensor with power consumption on the order of 1 mW. This property of event cameras increases the efficiency of any algorithm in the input phase as compared to using a traditional sensor. The particular task we will evaluate is optical flow, which can be used for visual navigation algorithms. Because of the unique format of event data, events need to be reconstructed into an intensity representation in order to exploit traditional vision-based optical flow. There are many methods for event-based intensity estimation in the literature, however it is unknown which method is best for particular tasks in-terms of both accuracy and efficiency. We evaluate five event reconstruction methods in-relation to the task of optical flow. The results show that the performance of the task is directly related and the efficiency is inversely related to the complexity and photometric consistency of the reconstruction method.

4.1 MOTIVATION

The goal of this chapter is to evaluate a traditional computer vision task, optical flow, with event-based vision. Due to the novel representation of event data, in order to have compatibility with traditional algorithms, the events will first be reconstructed into intensity frames. This intermediate event representation is useful in bridging the knowledge gap between traditional and event camera knowledge and research. Using the end-task of optical flow, five event

reconstruction methods are evaluated. Both computational complexity and task accuracy is considered in the comparison.

One application of optical flow is motion estimation in visual odometry [36]. Camera motion and depth maps are estimated from the resulting flow field vectors of optical flow. Other applications include improving video quality (motion stabilization and super resolution), segmentation, and tracking.

4.2 RELATED WORK

Event-based optical flow estimation methods range from iterative asynchronous methods [3] inspired by the Lucas-Kanade algorithm [48], plane fitting methods that use the plane-like shape of spatio-temporal event streams [4], and variational optimization based approaches that use image data as well as events [39]. There are also learning-based approaches that use a U-Net architecture to directly estimate sparse optical flow [58, 26, 57]. In our work we will concentrate on a traditional vision learning-based optical flow method, RAFT, that computes dense optical flow [50].

An approach related to this chapter, E-RAFT was published after my work on this area, it extends RAFT for event-based vision [18]. This work builds on the RAFT architecture with changes made for working with events. To utilize events without reconstruction with a CNN, the events are split into short sequences and represented as volumetric voxel grids. A voxel grid discretizes the time dimension, but retains most of the temporal data through bilinear interpolation. E-RAFT introduces a differentiable warm-starting method which initializes the flow estimate from the last prediction instead of initializing it with zero as in the original RAFT. They report a 23% increase of performance compared to previous event-based optical flow methods.

Although this is a compelling approach to perform optical flow with direct events, in this chapter we evaluate event reconstruction methods and their ability to extract optical flow. This is an important approach because intermediate event representations are useful for many downstream applications, not just optimized for one task. Intermediate event representations also enable the use of events with traditional vision tasks without any modifications.

4.3 APPROACH

4.3.1 Reconstruction

Because events are a fairly new representation, in order to utilize traditional optical flow, we first reconstruct the events into an intensity frame. Two simple integration based and three learning based reconstruction methods are used.

High-pass Filter: One of the reconstruction techniques used, the high-pass filter, is represented by:

$$L^{k+1}(x,y) = \exp(-\alpha\Delta t)L^k(x,y) + p \tag{4.1}$$

Where alpha is the cutoff frequency of the high-pass filter and delta t is the time since the last event at the same pixel. This is the event only method purposed by Scheerlinck [46]. An example of an image created with this method is shown in Figure 15 on the top left.

Leaky Integrator: The second method used is the leaky integrator:

$$L^{k+1}(x,y) = \beta L^k(x,y) + p, \beta \in [0,1], p \in \{-1,1\} \tag{4.2}$$

Where L is the log image intensity and p is the event polarity. With beta equal to one correlates to direct integration of the events.An example of an image created with this method is shown in Figure 15 on the top right.

E2VID: E2VID is a learning based reconstruction technique that uses no hand-crafted priors to reconstruct intensity images directly from an event stream. The network architecture as shown in Figure 13 has a recurrent UNet-like form. The method incorporates a perceptual loss to encourage the reconstructions to match natural image statistics. It is trained on a large simulated dataset [41]. An example of an image created with this method is shown in Figure 15 on the middle left.

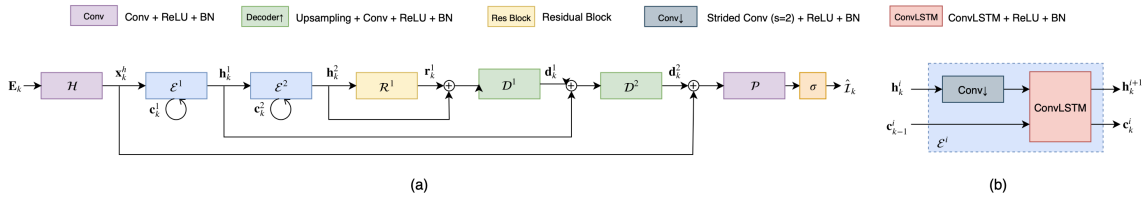


Figure 13. E2VID architecture composed of (a) recurrent encoder layers and followed by (b) residual blocks and decoder layers. Figure from [41]

FireNet: FireNet is a faster version of E2VID which uses 10x less FLOPs and is 99.6 percent smaller. The network architecture as shown in Figure 14 is a fully convolutional recurrent neural network with gated recurrent units and residual blocks [47]. An example of an image created with this method is shown in Figure 15 on the middle right.

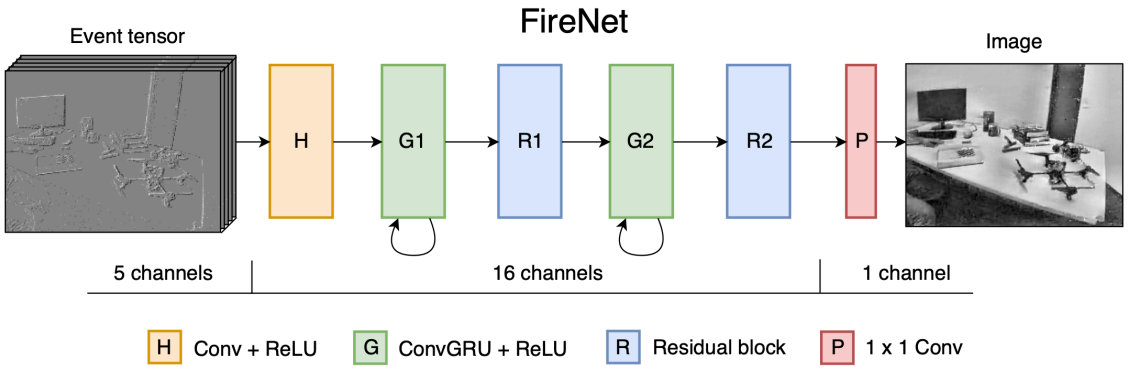


Figure 14. FireNet architecture composed of convolutional layers, convolutional gated recurrent units, and residual blocks. Figure from [47]

VNet: VNet is a minimal CNN we developed composed of five convolution layers with a ReLU activation and a final sigmoid layer Table 2. Normalized voxel grids are used as the event input. This representation is used to evaluate the necessity of using relatively large and complicated networks like E2VID and FireNet for reconstruction, particularly for the task of optical flow. An example of an image created with this method is shown in Figure 15 on the bottom.

Table 2. VNet architecture

Num. Layers	Conv. Filters	Conv. Activation	Last Layer Activation
5	3 x 3	ReLU	Sigmoid

4.3.2 RAFT

Deep learning methods for optical flow have resulted in more robust results and faster inference time as compared to traditional optimization methods. RAFT (Recurrent All-Pairs Field Transforms for Optical Flow) is the current state-of-the-art method for estimating dense optical flow. It uses both convolutional neural network (CNN) and recurrent neural network (RNN) architectures and is split into three stages: feature extraction, visual similarity, and iterative updates. See Figure 16 for an overview of the system. Two consecutive frames are used as input to the network. The feature extraction step is composed of two convolutional neural networks with shared weights. A context network of the same style is also used to extract features, but from only the first image. The visual similarity step calculates the inner product of all pairs of feature maps. This results in 4D correlation volumes that specify small and large pixel displacements. A correlation pyramid which is used to create multi-scale image similarity features is created from these correlation volumes. In the iterative update a sequence

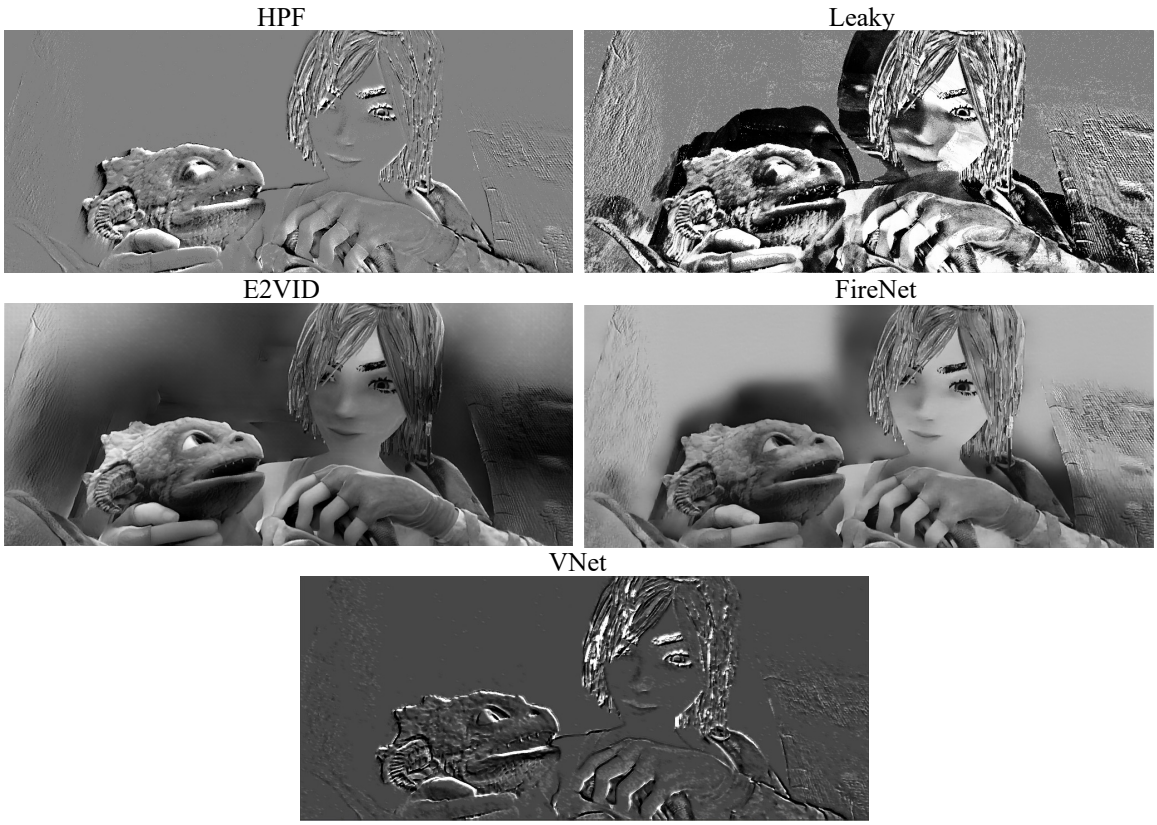


Figure 15. Visualization of event reconstruction techniques on the simulated event Sintel dataset.

of Gated Recurrent Unit (GRU) cells mimic an iterative optimization algorithm producing an optical flow prediction. The input to the GRU cells are the previous hidden state and the flow, correlation, and context features. Finally a convex upsampling module is used as the optical flow is output at 1/8 the resolution of the initial image. The L1 distance between ground truth and predicted flow is used as the loss function [50].

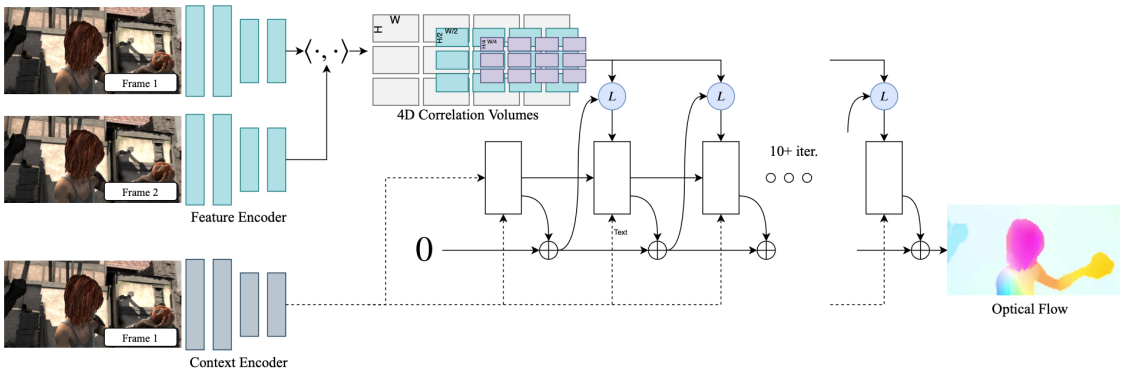


Figure 16. RAFT architecture composed of feature encoder, correlation layer, and update operator. Figure from [50]

4.3.3 RAFT on Event Reconstructions

My approach extends the RAFT framework for event-based vision. The event images are first reconstructed with the techniques described, then the optical flow is computed with RAFT. The pipeline for this approach is shown in Figure 17. The accuracy and efficiency of these event reconstruction techniques are evaluated. The results in Section 4.6 show an increased accuracy by training the neural network based event reconstructions with the vision task.

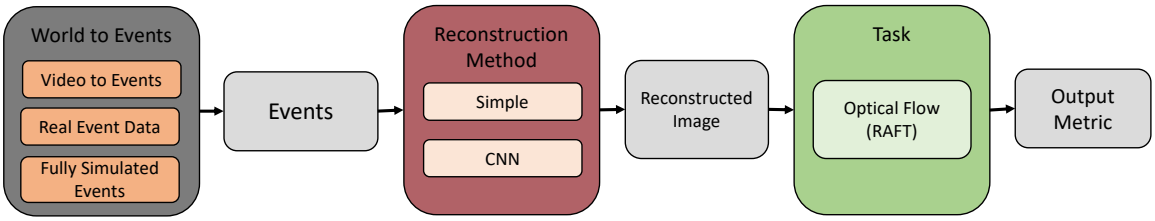


Figure 17. Experimental pipeline for analyzing event reconstruction techniques on the task of optical flow.

4.4 DATASET AND METRICS

The MPI-Sintel dataset was used for evaluation in this chapter [[6]]. Sintel is an open source animated CGI short film. Because the dataset is simulated, optical flow groundtruth is also available corresponding to the image timestamps. The dataset includes a variety of sequences with realistic atmospheric effects [55]. A sample of this dataset is shown in Figure 18

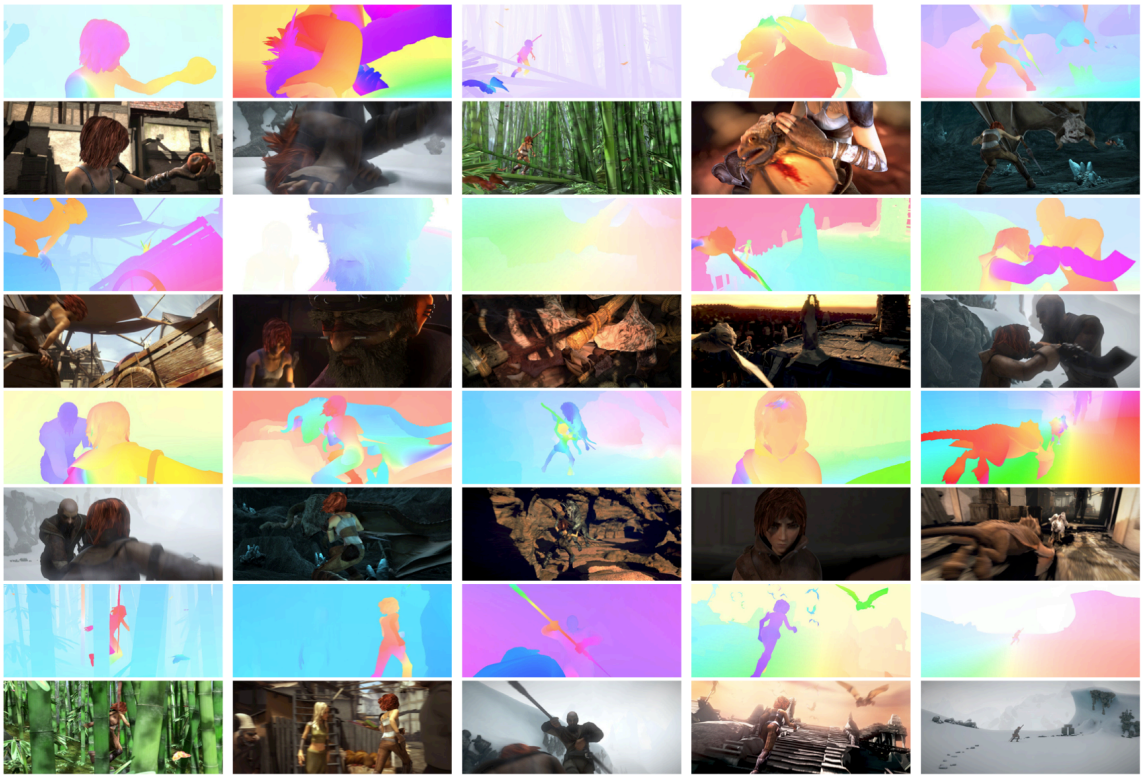


Figure 18. Sample of the MPI-Sintel dataset ground truth images and corresponding flow fields. Figure from [55]

The event data was generated through simulation utilizing V2E [20]. V2E converts video frames to realistic synthetic DVS events. The Super SloMo framework is used to interpolate the video frames since real events have a much lower latency than the original Sintel dataset videos. One requirement of using the Super SloMo framework is that the motion between

original frames not be too great otherwise bad artifacts will be introduced. Because of this the Sintel sequences with small motion between frames were selected to be used in this chapter and all others were discarded.

The metric used for optical flow evaluation is endpoint-error. This is calculated by computing the euclidean distance between the estimated optical flow vector and the groundtruth optical flow vector. The average of these comparisons throughout these frames is taken to represent the total error over the video. We also report the 1px, 3px, and 5px error. The 1px, 3px, and 5px value represents the mean number of endpoint error values less than 1px, 3px, and 5px respectively in magnitude. A larger value for the smaller pixel represents a better accuracy of the flow. The latency and computational complexity required to create the event frames is also taken into account and reported in the results section.

4.5 RESULTS

4.5.1 Latency and complexity

The latency of reconstructing a 240 x 180 image with 10000 events for each reconstruction method is given in Table 19. E2VID has the longest latency as it is the deepest network. Table 3 compares network memory and parameters among the three learning based reconstruction methods.

The latency of each method directly corresponds to the complexity of that method. The higher the complexity of the algorithm, the greater the latency. For our purposes, the simple reconstruction methods are the best options when only evaluating efficiency and not accuracy of the task.

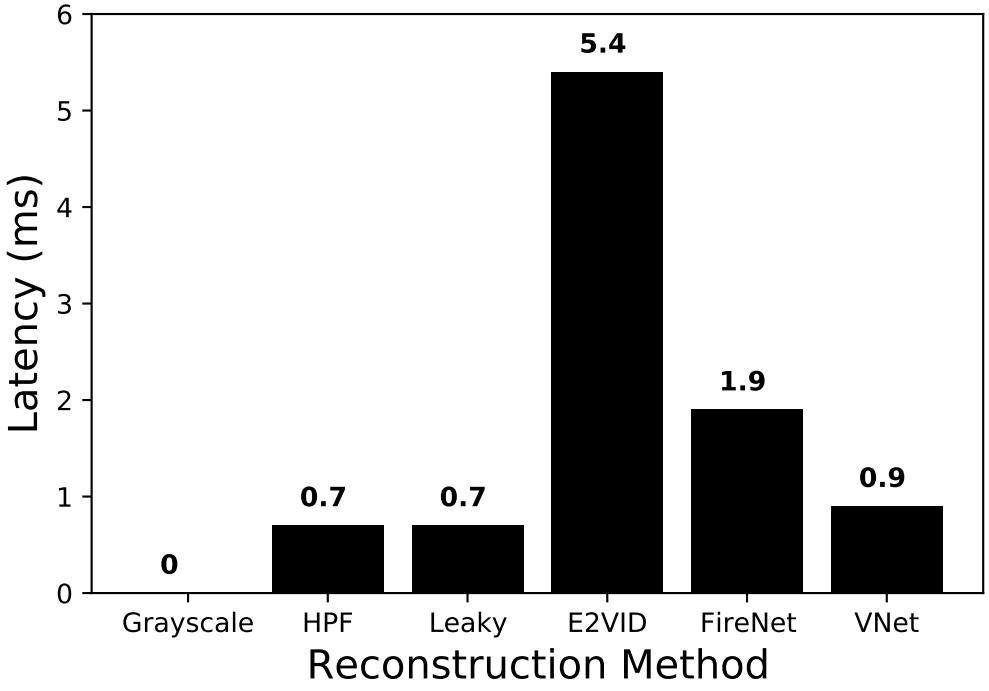


Figure 19. Representation vs inference time on a 240 x 180 resolution image. Results for E2VID and FireNet from [47].

Table 3. Representation vs network overview

	E2VID	FireNet	VNet
No. parameters	10700k	38k	577
Memory	43Mb	0.16Mb	2.31Kb
Downsampling	yes	no	no
Recurrent units	LSTM	GRU	no
Max. kernel size	5 x 5	3 x 3	3 x 3

4.5.2 Representation performance

The optical flow accuracy for the various event reconstruction methods is evaluated using the error metrics of RAFT. The RAFT model was trained on the FlyingThings and FlyingChairs datasets which are not event-based. The final results after evaluating RAFT with the recon-

structed image sequences is given in Table 4. E2VID performs the best besides the original non-event grayscale images, and on the other end leaky surface performs the worst. Figure 20 shows an example of each reconstruction and the corresponding visual of the ground-truth and learned optical flow result for that image.

From these results, we can see that the event reconstructions that have the most photometric consistency perform the best with RAFT. This was a non-obvious result because as demonstrated in Chapter 3, the most aesthetically pleasing image is not always the best performing in-terms of computer vision algorithms. With this observation in mind, among the learning based methods it makes sense that E2VID reconstructions result in the best RAFT error as E2VID also creates the most realistic (closest to grayscale) reconstruction. In the same way, FireNet has slightly worse quality reconstructions as compared to E2VID and results in a worse RAFT error.

Analyzing Figure 20 to compare the simple reconstruction methods, the resulting optical flow from the high-pass filter reconstruction has more detail than that of the leaky integrator as seen in the top left object and the girl's face. This correlates with the fact that the leaky reconstruction is more noisy as compared to the high-pass filter reconstruction. Although this is only a visual observation in relation to one image, it aligns with the analytic results where the high-pass filter method has a lower error than the leaky method.

The simple reconstruction methods, leaky integrator and high-pass filter, have a lower number of pixels with less than 1, 3, and 5 pixel error than the other methods. This shows that the simple methods are overall worse and there is not one part of the optical flow field that is much better than another part.

The VNet method results in an error similar to the simple reconstruction methods even though it is a learning based reconstruction. This seems to be a result of the very simple CNN architecture.

Figure 21 shows the tradeoff between latency and RAFT end-point error for the learning-based reconstruction methods. E2VID is the most accurate method, but also the slowest and most complicated neural network. FireNet provides a middle option with slightly worse error, but much faster inference and less parameters. Lastly VNet is the worst performing in terms of accuracy, but also the fastest and smallest network of the three. The choice of reconstruction method depends on the application of the optical flow results and whether the size/speed of the algorithm or accuracy of the results is most important.

Table 4. Event representation vs RAFT accuracy. RAFT trained with FlyingChairs and FlyingThings and evaluated on Sintel. Reporting end-point-error

Representation	EPE	1px	3px	5px
Grayscale	0.206	0.974	0.994	0.997
HPF	0.829	0.855	0.952	0.972
Leaky	1.265	0.742	0.903	0.937
E2VID	0.478	0.915	0.981	0.991
FireNet	0.591	0.892	0.976	0.988
VNet	0.947	0.782	0.951	0.977

4.5.3 Effects of event threshold

In order to study the robustness of these reconstruction techniques to real-world event camera parameters, an ablation study is performed of the event thresholds. Event cameras have a parameter which controls the sensitivity of generating events. There is a positive threshold which is the necessary change in log intensity to trigger a positive event as well as a negative threshold to trigger a negative event. Both the positive and negative event thresholds were varied while simulating the Sintel event data with V2E. Then the RAFT error was recorded

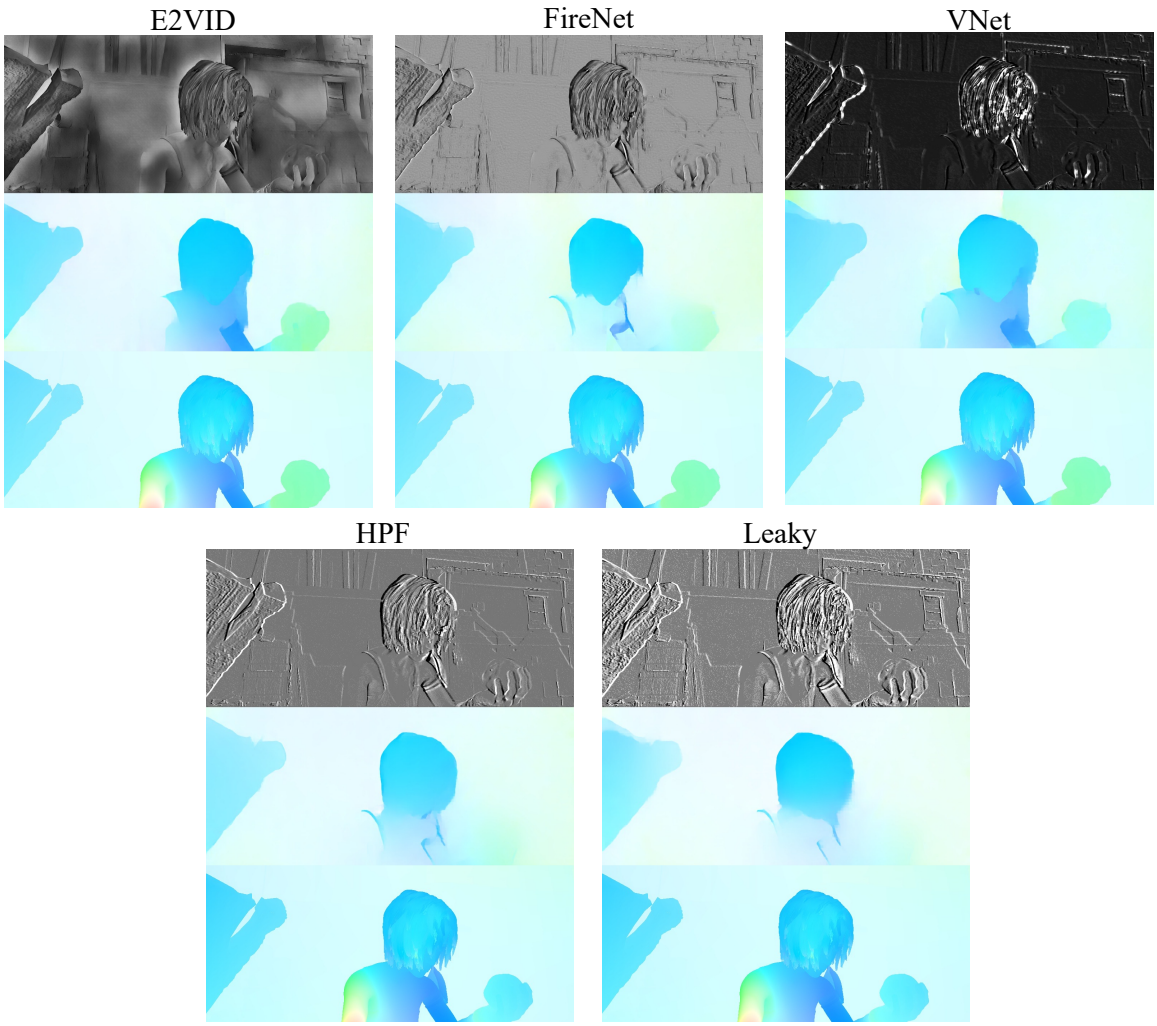


Figure 20. Event reconstruction optical flow results. Top: reconstructed event images, Middle: estimated optical flow, Bottom: ground-truth optical flow.

using this new dataset. The default V2E event threshold of 0.2 for both positive and negative events was used for the previous results in Table 4.

Table 5 reports the results of both increasing and decreasing the threshold from the default value. The visual results of the event reconstructions are shown in Figure 22. These results are harder to interpret the correlation between event threshold and optical flow error.

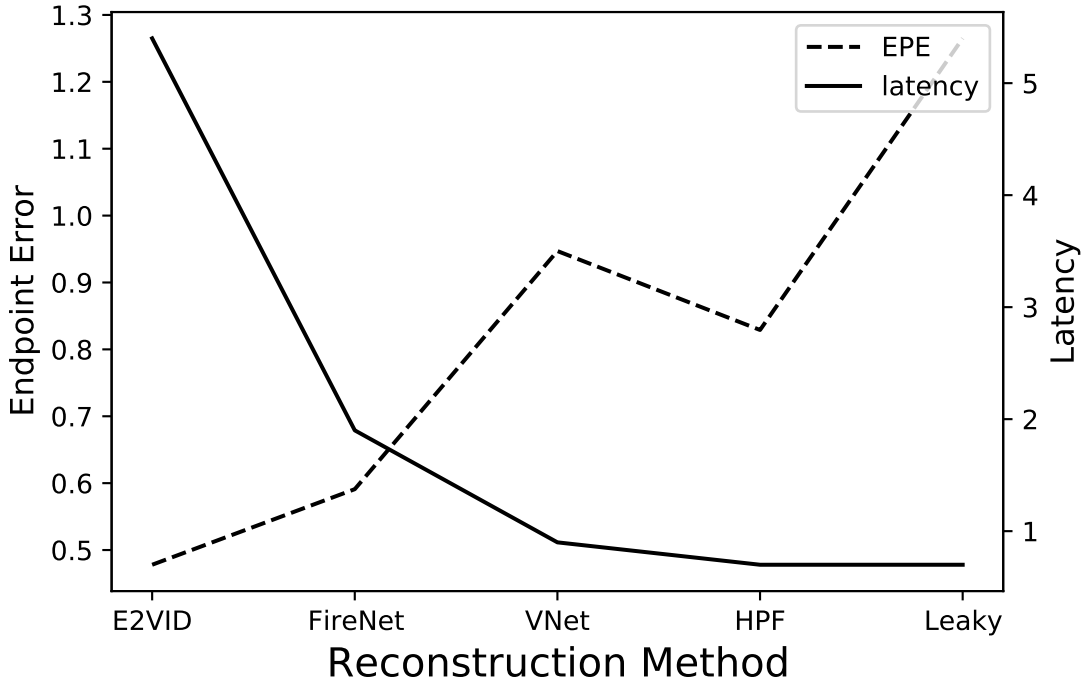


Figure 21. Event reconstruction methods latency vs optical flow error.

Table 5. Ablation study of event-camera positive and negative threshold. Reporting end-point-error of RAFT

Representation	0.1 thresh.	0.2 thresh.	0.4 thresh.	0.6 thresh.	0.8 thresh.
HPF	1.243	0.829	1.022	1.356	1.647
Leaky	0.847	1.265	1.764	2.290	2.707
E2VID	0.876	0.478	0.594	0.848	1.170
FireNet	0.975	0.591	0.707	0.957	1.321
VNet	0.634	0.947	0.993	1.095	1.229

4.5.4 Effects of noise

The effect of noise from the event camera on the performance of RAFT was also evaluated. There are multiple sources of noise from an event camera, including leak events, shot noise, and limited bandwidth. In this section the V2E parameter “noisy” was used to add a combination

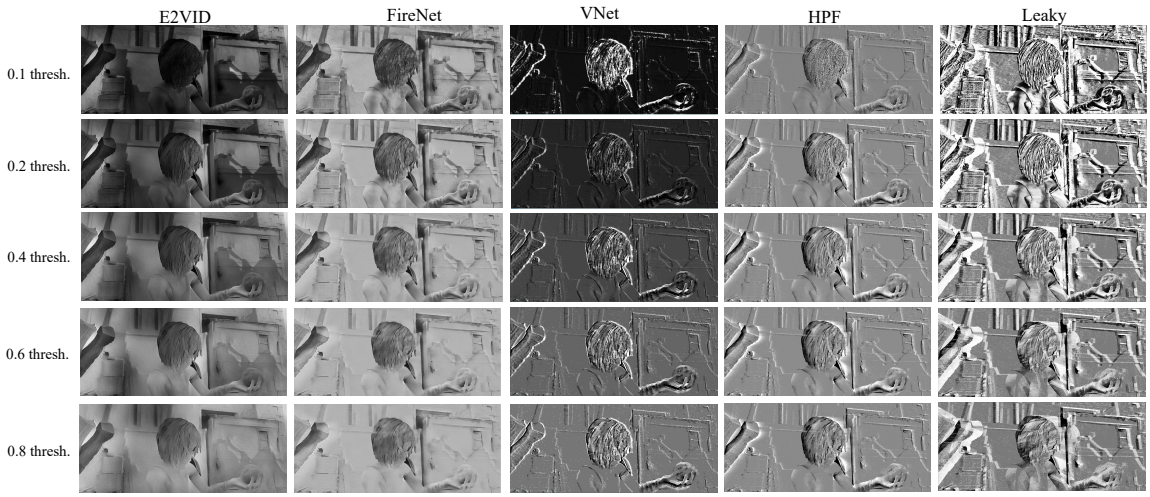


Figure 22. Event reconstructions with varying event thresholds.

of these sources of noise to the simulated events. Table 6 provides the quantitative results of evaluating RAFT with this new dataset and Figure 23 shows the event reconstructions.

All reconstruction methods worsened in performance relative to RAFT error as noise was increased as compared to the results in Table 20. From the observations made in the initial Representation Performance section, that the event cleanest reconstructions most similar to traditional images performed best, this result further confirms that view.

Table 6. Ablation study of event-camera noise

Representation	EPE	1px	3px	5px
HPF	0.829	0.855	0.952	0.972
HPF noisy	1.28	0.757	0.912	0.944
Leaky	1.265	0.742	0.903	0.937
Leaky noisy	1.458	0.709	0.890	0.931
E2VID	0.478	0.915	0.981	0.991
E2VID noisy	1.310	0.700	0.901	0.945
FireNet	0.591	0.892	0.976	0.988
FireNet noisy	1.679	0.650	0.869	0.913
VNet	0.947	0.782	0.951	0.977
VNet noisy	1.07	0.706	0.927	0.975

4.6 OPTIMIZING REPRESENTATION FOR VISION TASK

In order to try to further improve the results from event reconstructions, the learning-based methods of event reconstruction were optimized for the task. Each method was fine-tune trained with the RAFT loss. The RAFT model trained on the FlyingChairs and FlyingThings datasets was used and the Sintel dataset was used for fine-tune training of the reconstruction methods. The results are shown in Table 7 and Figure 24.

The results show that training the reconstruction network for the specific task we are performing improves the end-point error as compared to Table 20. This provides motivation for a future work of creating an event-based vision pipeline that learns the reconstruction method based on the vision task.

Table 7. Optimized event reconstructions for the task of RAFT. Representation vs RAFT accuracy. RAFT trained with FlyingChairs and FlyingThings and evaluated on Sintel (train). VNet trained with RAFT loss on Sintel

Representation	EPE	1px	3px	5px
VNet	0.664	0.857	0.967	0.985
E2VID	0.201	0.743	0.994	0.997
FireNet	0.565	0.882	0.977	0.990

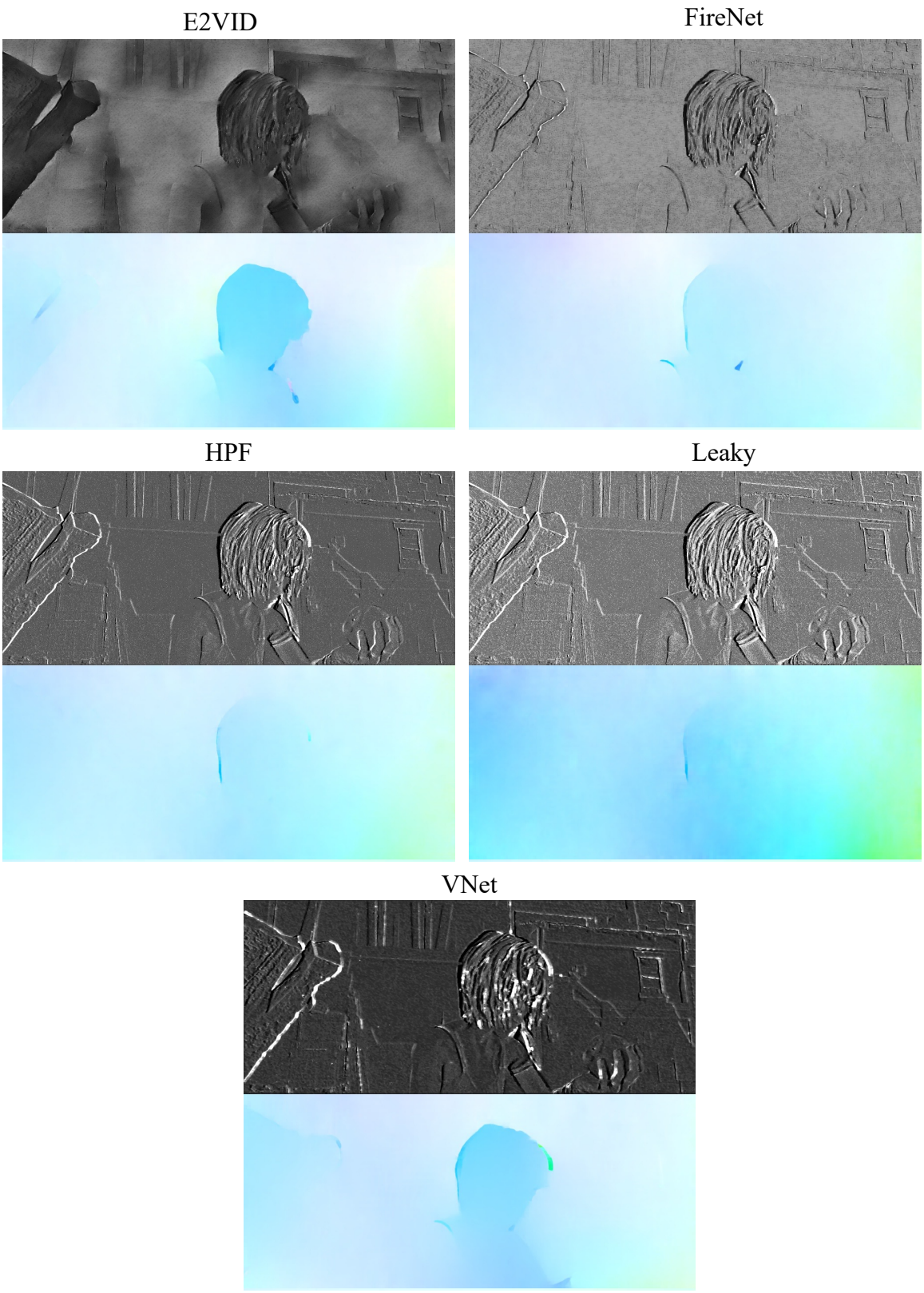


Figure 23. Event reconstructions and resulting optical flow with event camera noise

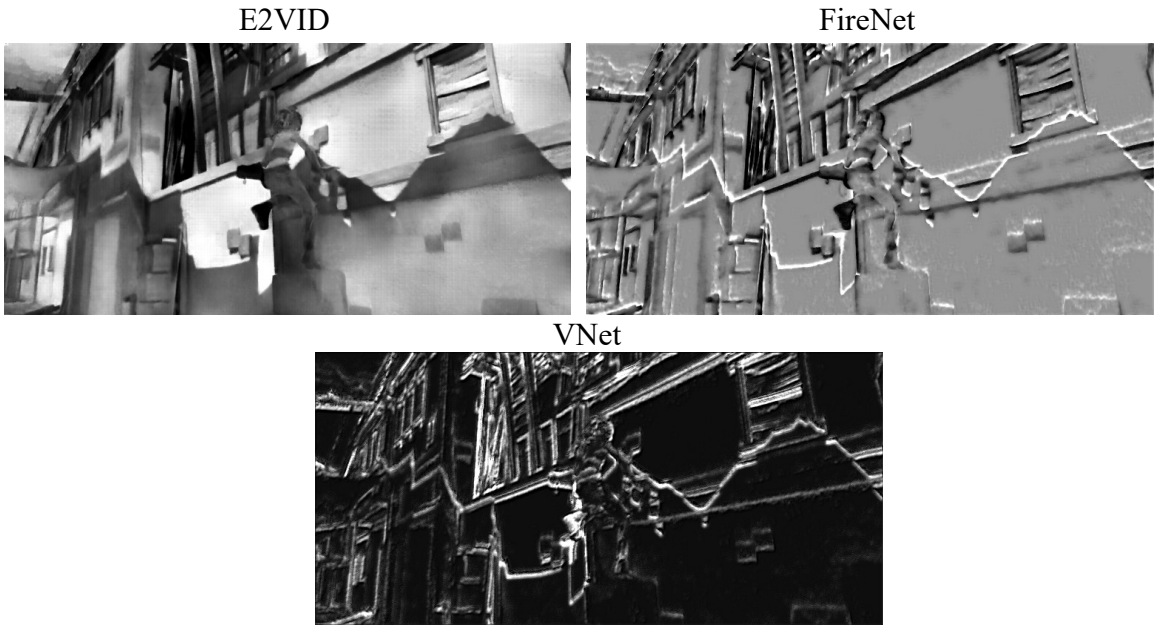


Figure 24. Results of training event reconstruction method for optical flow task.

Chapter 5

A CASE STUDY OF THE CHALLENGES IN ADAPTING ORB-SLAM2 FOR AN EVENT FEATURE-BASED SLAM

Continuing with the theme of energy-efficient visual navigation, in this chapter we evaluate performing a feature-based simultaneous localization and mapping (SLAM) system, ORB-SLAM2, with event-based cameras. Initial experiments of using event reconstructions for this task proves to be insufficient to acquire reliable results. Due to this observation, we perform a study of the initialization stage of the algorithm. In addition, we developed a convolutional neural network (CNN) for learning ORB feature descriptors for events. The results from this network demonstrate additional challenges to the problem of an event feature-based SLAM, which are further discussed in this chapter.

5.1 MOTIVATION

The main event-based visual odometry (VO) and simultaneous localization and mapping (SLAM) techniques use geometrical and intensity based approaches. Two methods stand out for monocular pure event-based VO for 6-DoF motions in natural 3D scenes. The method in [22] uses three interleaved Baysian filters which estimate image intensity, depth and camera pose. A geometric approach is used in [40] based on a technique of semi-dense mapping and an image alignment tracker. This method does not recover the absolute intensity image of events. These current approaches focus on geometrical solutions for monocular event only vision and there is a lack of work on feature-based methods.

There are also multiple methods that utilize event cameras alongside other sensor types.

In this area there is more work on feature-based methods. For example, the method in [54] combines events, images, and IMU to perform SLAM in high-speed and HDR scenarios. This method tracks corner features with the Lucas-Kanade tracker through both frames and event frames [48] and then uses these for triangulation of landmarks.

In this chapter we aim to bridge the gap with a feature-based SLAM approach that only uses event cameras and no other sensors. While current approaches focus on creating new SLAM systems for event vision, we believe it is useful to adapt traditional methods that have proven to be reliable and accurate for event vision. ORB-SLAM2, which is a feature-based algorithm, is one of the most reliable, complete, and accurate methods for monocular SLAM available [38]. In order to take advantage of this system, in this chapter we investigate incorporating event-based vision with feature-based monocular ORB-SLAM2.

We work towards a modular approach of changing the input representation, but using an established back-end SLAM system, ORB-SLAM2. Adaption of a traditional feature-based SLAM improves the ability to incorporate event-based vision in legacy systems as well.

5.2 DATASET AND METRICS

The TUM RGB-D benchmark dataset [49] was used in the studies of this chapter. This dataset provides sequences of RGB images along with ground truth trajectory obtained with an external motion capture system. The images have a resolution of 640x480. The corresponding event data was generated through simulation utilizing V2E [[20]]. V2E converts video frames to realistic synthetic DVS events. The Super SloMo framework was used to interpolate the video frames since real events have a lower latency than the RGB videos. We utilize 7 sequences from this dataset for our experiments.

We also utilize the Event-Camera Dataset in the feature-matching evaluation portion of

this chapter [35]. This dataset consists of real event data from a DAVIS240C including the events, images, and IMU measurements. Both the images and events have a 240 x 180 spatial resolution.

Our evaluation error metric for ORB-SLAM2 is absolute trajectory error (ATE) defined as the difference between points of the true and the estimated trajectory [49]. The true and estimated poses are matched via timestamps and then aligned using a similarity transform [19], as the scale of monocular SLAM is unknown. Then ATE is calculated as a root mean squared error.

5.3 CHALLENGES

5.3.1 Event Reconstructions with ORB-SLAM2

We first evaluated using event intensity reconstructions for event ORB-SLAM2 similar to the work done in Chapter 4 of this thesis. The event reconstruction techniques evaluated are the learning-based network E2VID and the simple reconstruction methods high-pass filter and leaky integrator [42, 46]. The results of these tests on the TUM RGB-D dataset are given in Table 8.

Almost all of the sequences failed to initialize with ORB-SLAM2. E2VID was the only reconstruction type that began tracking and mapping for three out of the seven sequences evaluated. For those three sequences however, the initialization of ORB-SLAM2 was repeated multiple times due to a bad initialization and the tracking was often lost. The ATE values indicate that there was poor accuracy for the portion of the tracking that was completed.

This result demonstrates that unlike the results from Chapter 4 with optical flow, simply

Table 8. ORB-SLAM2 event reconstruction results (Absolute Trajectory error RMSE in cm). The sequences marked with X did not initialize. Sequences with * only track approximately half of the video.

Sequence	grayscale	E2VID	hpf	leaky
fr1_xyz	0.90	X	X	X
fr2_xyz	0.29	X	X	X
fr1_desk	1.74	X	X	X
fr2_desk	0.84	3.65*	X	X
fr3_long_office	1.9	17.7*	X	X
fr3_sit_halfsph	1.34	X	X	X
fr3_walk_halfsph	1.74	21.7*	X	X

reconstructing the event images into intensity form does not work well for this feature-based SLAM task.

5.3.2 ORB-SLAM2 Initialization

Due to the difficulty ORB-SLAM2 showed in initializing with event reconstructions, we further evaluated the initialization process of the system. We discovered that the feature matching step of the ORB-SLAM2 system prevented initialization of the mapping and tracking threads. ORB-SLAM2 first extracts features from the keyframe and the current frame. If a minimum number of features are not found, then features are extracted from new frames until the threshold is met. After the minimum number of features have been found in these frames (1000 features is the default), matching between the features is completed with a Brute-Force matcher. A maximum threshold of 50 for the distance between features is set to designate the features as a match. The last parameter that must be met for initialization is at least 100 matches must be found between frames.

The performance of ORB-SLAM2 with E2VID event reconstructions was evaluated while varying the ORB-SLAM2 initialization parameters. Only E2VID results are provided here because the other reconstruction methods were still unable to initialize with these changes.

Table 9 reports the results of increasing the number of features that are extracted from each frame. Increasing the number of features by 500 resulted in ORB-SLAM2 initializing for all sequences. However, after initialization the system has frequent re-initialization and lost tracks. For the few results that the ATE is low, only a very small portion of the trajectory was completed.

Table 9. ORB-SLAM2 event reconstruction results (Absolute Trajectory error RMSE in cm) with varying number of features extracted from each frame. The sequences marked with X did not initialize. Sequences with * only track approximately half of the video or less.

Sequence	1000 features	1500 features	2000 features	3000 features
fr1_xyz	X	19.4*	18.5*	3.74*
fr2_xyz	X	2.23*	1.45*	24.4*
fr1_desk	X	7.98*	6.56*	19.9*
fr2_desk	3.65*	3.36*	3.58*	3.29*
fr3_long_office	17.7*	165*	39.9*	148*
fr3_sit_halfsph	X	36.8*	4.35*	4.5*
fr3_walk_halfsph	21.7*	38.9*	38.8*	38.1*

We also tested varying the maximum feature matching distance and minimum number of feature matches thresholds. These changes resulted in the same outcome as changing the number of initial features found in each frame. Specifically, the system was more likely to initialize, but had bad tracking accuracy and consistency. These results demonstrate that simply lowering the requirements of ORB-SLAM2 initialization does not make the system succeed with event reconstructions.

5.3.3 ORB Feature Matching

In this section, we further evaluate ORB feature matching to aid in determining why event representations do not perform well with ORB-SLAM2. This is done with Python OpenCV packages with no connection to ORB-SLAM2. The only difference between the OpenCV and

ORB-SLAM2 implementation of ORB is ORB-SLAM2 enforces a grid to have the keypoints evenly distributed over the whole image. Since this restriction decreases the number of features found, in this section this modification is unimportant.

TUM RGB-D Sequence: Figure 26 shows the result of performing brute-force feature matching between extracted ORB features. Only a subset of all feature matches found are visualized. More matches are found in the traditional images than the event representations. Straight lines between features usually mean the matches are good. Examining these figures, we observe that both event representations perform poorly and generate mostly incorrect feature matches. This seems to be caused by an inconsistency of edges and corners between frames. Because of this inconsistency, the descriptors for a feature at the same location are different which causes high matching distances.

A quantitative evaluation of this qualitative observation was also performed. The number of feature matches and feature matching distance between ten pairs of sample images from the freiburg1_desk sequence are given. For the E2VID event images, ORB resulted in an average of 162 matches with a mean matching distance of 45. For the one bin event voxel images, ORB resulted in an average of 90 matches with a mean matching distance of 43. For the traditional images, ORB resulted in an average of 425 matches and a mean matching hamming distance of 36. The average number of matches with a matching distance below the ORB-SLAM2 threshold of 50 was 109 for E2VID and 345 for the traditional images. Figure 25 shows the distribution of matching distances under 50 for two example frames.

Due to the poor reconstruction quality of this dataset with the E2VID method, we further evaluated the event simulation process. We discovered that the V2E method, which converts the traditional video sequence to events, produced bad artifacts in the conversion process. The Super SloMo framework that interpolates between the frames, introduced warping artifacts due to large pixel displacements between the original sequential frames [20].

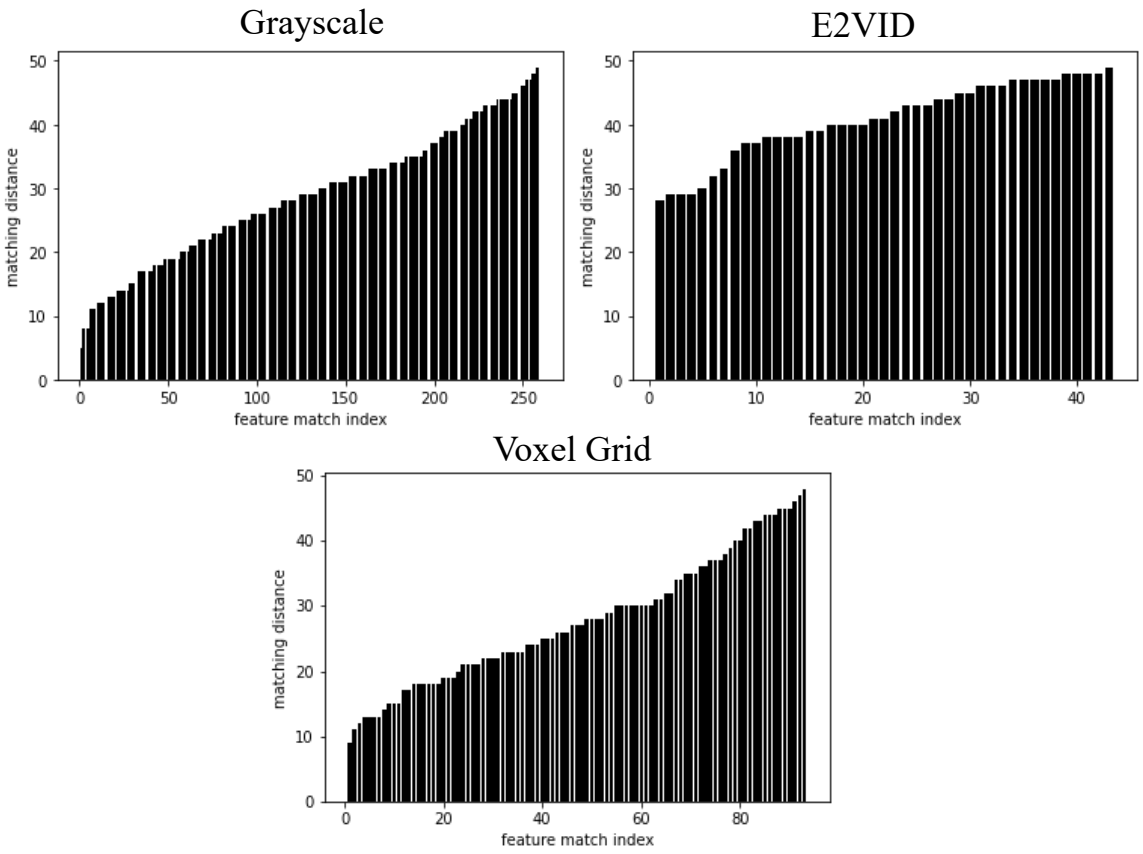


Figure 25. ORB feature matching with the simulated event sequence. Top: ORB feature matching distance for traditional images. Bottom: ORB feature matching distance for E2VID event images

DAVIS240C Sequence: This was a surprising result that E2VID reconstructions, which are reported to be of a high image quality, do not perform well with feature matching. To test if the problem is due to the poor quality of the simulated event data, we also evaluate with the real event dataset used in the E2VID paper for visual-inertial odometry [42].

The qualitative results are shown in Figure 29. More correct matches are found with the E2VID reconstructions as compared to the previous sequence. The one bin voxel grid, however, resulted in a very minimal number of matches where most are obviously incorrect. The number of events generated with this sequence as compared to the frieburg sequence is much fewer.

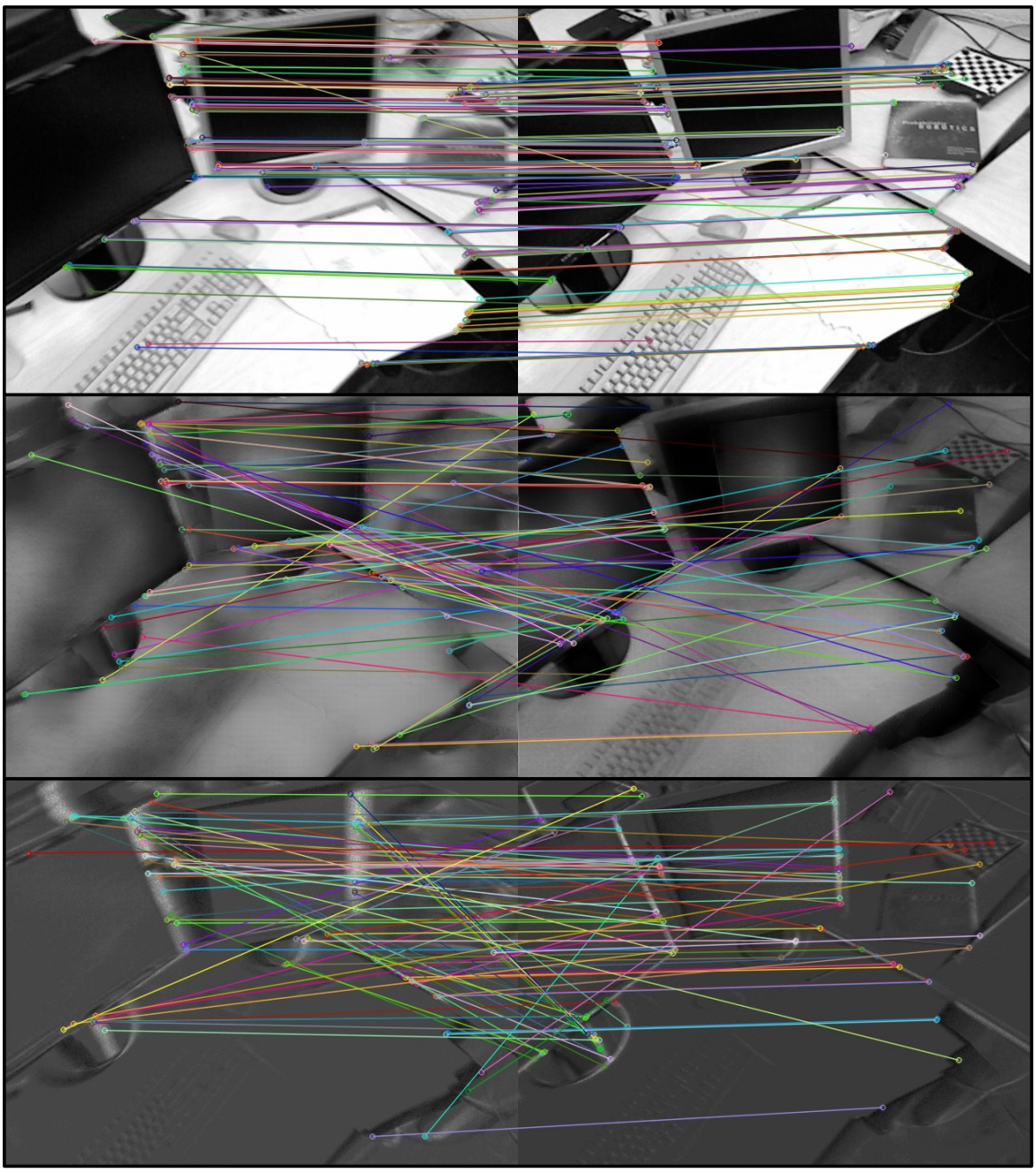


Figure 26. Brute-force matching of ORB features on the simulated sequence: Top is traditional grayscale images, Middle is E2VID reconstructed event images, Bottom is one bin voxel grid event representation

This is a difference between the event trigger thresholds of the DAVIS camera and the simulated data.

For a quantitative evaluation, ten frames from the dynamic_translation sequence are used. The event data reconstructed with E2VID resulted in an average of 210 matches with a mean matching distance of 37. The corresponding traditional grayscale images resulted in an average of 231 matches with a mean matching distance of 28. We do not report the averages from the voxel grid images as the features are visually incorrectly matched. Figure 27 shows the distribution of matching distances under 50 for two example frames. Although this dataset performs better feature matching with the event E2VID reconstructions, the grayscale and event data it does not work with ORB-SLAM2. This seems to be due to the low spatial resolution of the frames.

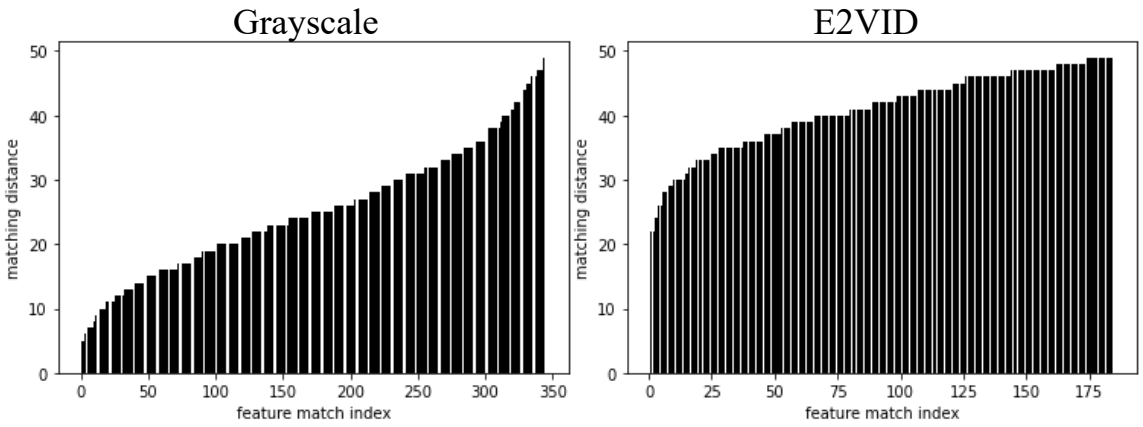


Figure 27. ORB feature matching with the real event sequence. Top: ORB feature matching distance for traditional images. Bottom: ORB feature matching distance for E2VID event images

5.3.4 Learning Descriptors for Event ORB-SLAM2

We demonstrated in the previous section the difficulty of performing feature matching with event data in the form of traditional grayscale images as one bin voxel grids. In this section we present the challenge of performing ORB-SLAM2 with non-reconstructed events.

Approach: The feature descriptors for the event data are output from a neural network which are then used as the input to ORB-SLAM2. Our approach for learning event features is shown in Figure 28. Ground-truth feature keypoints and descriptors were obtained by running ORB-SLAM2 on the traditional grayscale images from the TUM RGB-D freiburg1_desk sequence. The input to the neural network is a 31 x 31 event voxel grid patch centered around a keypoint from the groundtruth data.

The convolutional neural network we use is a variation of the AlexNet with a decreased number of layers and varying number of input channels for the voxel grid [24]. The architecture is given in Table 10.

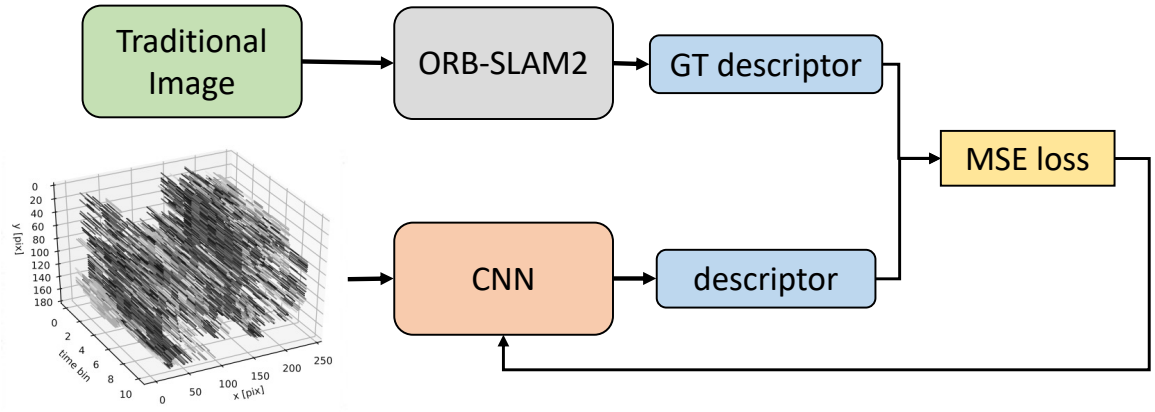


Figure 28. Event ORB feature descriptor learning pipeline

Results: Many variations of the AlexNet-like architecture were evaluated. The initial training resulted in averaged outputs (same output descriptor for different voxel grid input patches). Including dropout layers improved these results, however the test loss never decreased enough. We also tested with adding batch normalization layers as well as varying the learning rates, batch sizes, number of voxel grid bins, and number of minimum nonzero pixels for each voxel grid input.

However, this method resulted in an inability to learn descriptors well enough for consistent

Table 10. Event feature descriptor learning architecture.

Layer	Num. Filters	Filter Size	Activation Function	Output Size
Input	-	-	-	num. bins x 31 x 31
Convolution	16	11	ReLU	16 x 7 x 7
Convolution	48	3	ReLU	48 x 7 x 7
Convolution	96	3	ReLU	96 x 7 x 7
Convolution	64	3	ReLU	64 x 7 x 7
Convolution	16	3	ReLU	16 x 7 x 7
AdaptiveAvgPool	-	-	-	16 x 6 x 6
Linear	-	-	ReLU	256 x 1 x 1
Linear	-	-	ReLU	128 x 1 x 1
Linear	-	-	ReLU	32 x 1 x 1

matching in the ORB-SLAM2 system. Our finding from testing noise in the feature descriptors demonstrated that the error must be very small for the descriptors to match well. This method of learning descriptors was not able to reach that small error suitable to work well. A MSE error from the network of 0.03 corresponds to a descriptor matching distance of 50 which is the maximum threshold of ORB-SLAM2. The lowest the test loss decreased to with our network was 0.09.

Noise in descriptors While performing the learning of event descriptors, we evaluated the resilience of feature matching in the ORB-SLAM2 system to noise. Random noise was added to each of the 32 8 byte values of the descriptors. It was discovered that adding noise of even +-1 resulted in the system not finding enough matches and not initializing. Decreasing the match distance threshold increased the probability that enough matches would be found, but tracking was more likely to be lost in the middle of the sequence.

5.4 DISCUSSION

The initial results in this chapter demonstrate the fact that ORB-SLAM2 with event voxel grids, E2VID, HPF, or Leaky Integrator event reconstruction methods does not consistently generate accurate trajectories and maps. Through the challenges evaluated in this chapter, we have provided many insights into why these event representations did not work in hopes of aiding future work on this topic. The main factor influencing poor results is the quality of the event data, like spatial resolution, and the consistency of events through time. The ORB feature matching evaluations show that the real event data, which seems to be less prone to blur and ghosting, performs better than the simulated data that does have these effects. One path to attempt performing ORB-SLAM2 with event reconstructions might be to generate a high resolution, high quality event dataset with trajectory ground truth.

The other pathway of performing ORB-SLAM2 with non-reconstructed event data through voxel grids also provides useful insights for future work. The failure of our CNN to learn event feature descriptors demonstrates that the problem needs a more advanced solution. It is planned as future work to evaluate a different input event representation that preserves features through time, such as an event time surface [25].

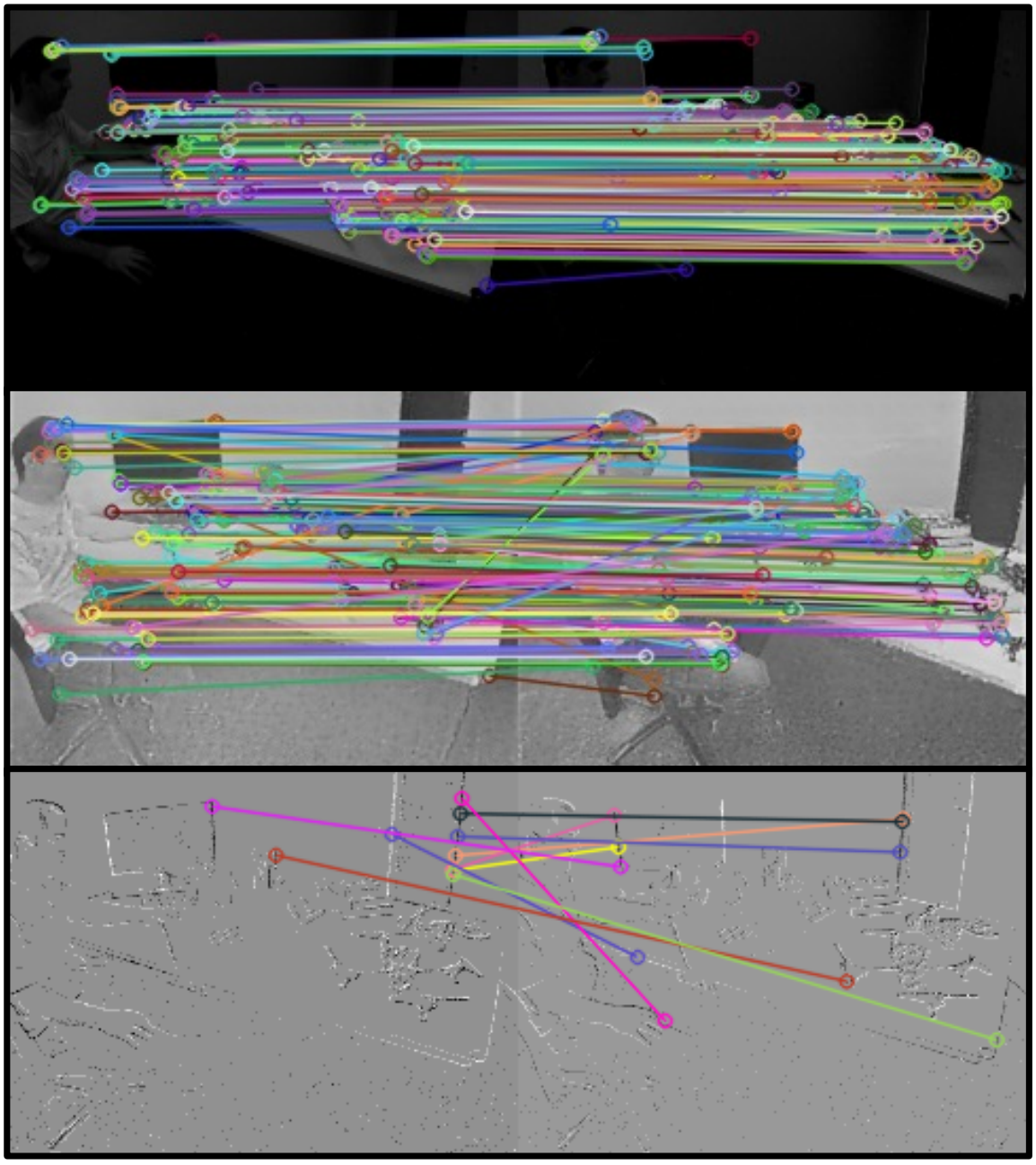


Figure 29. Brute-force matching of ORB features on the real sequence: Top is traditional grayscale images, Middle is E2VID reconstructed event images, Bottom is one bin voxel grid event representation

Chapter 6

CONCLUSION

In this thesis we presented methods and tradeoffs for energy-efficient visual navigation. In-particular we concentrated on accuracy/efficiency tradeoffs of using varying bit level sensor quantization and event camera sensors for the tasks of simultaneous localization and mapping (SLAM) and optical flow. We first presented results from performing varying levels of linear and logarithmic quantization of RAW images with the ORB-SLAM2 system. It was shown that logarithmic quantization outperforms linear quantization with both direct and indirect SLAM systems. The gradient-based quantization scheme we introduced also showed improved energy-efficiency while maintaining performance.

Additionally, we presented a study of event-based vision reconstruction techniques with a learning-based optical flow, RAFT. The quality of the intensity reconstructions directly correlated with the resulting error of the optical flow results. We also showed that learning or fine-tuning the event reconstruction with RAFT loss improves the results.

Lastly, we provide a case study of performing feature-based visual SLAM (ORB-SLAM) with event-based vision. The difficulty of matching traditional features, which is necessary for ORB-SLAM with events is demonstrated. Due to the inconsistency between frames of event representations, the descriptors are unable to be matched correctly.

6.1 LIMITATIONS AND FUTURE DIRECTIONS

A limitation of Chapter 3 is that the results of linear, logarithmic and gradient quantization only worked well for ORB-SLAM and not LSD-SLAM. This is due to the differences between

feature-based and direct-based SLAM algorithms. In future work, changes could be evaluated for adapting these methods for direct-based SLAM. For example, initial tests showed that performing the process of demosaicing, which is part of the image sensor pipeline (ISP), further improved results.

The event-based vision results provided in this thesis are limited to reconstructing the continuous event data representation into discrete frames. We specifically evaluated this format of event data to provide a modular approach separating the front-end representation from the back-end vision task. The studies performed in Chapters 4 and 5 provide background for a possible future work of implementing a front-end event reconstruction method adaptable to various vision tasks.

REFERENCES

[1] Tim Bailey and Hugh Durrant-Whyte. “Simultaneous localization and mapping (SLAM): Part II”. In: *IEEE Robotics & Automation Magazine* 13.3 (2006), pp. 108–117.

[2] Nate Barnett, Dave Costernaro, and Ingrid Rohmund. “Direct and Indirect Impacts of Robots on Future Electricity Load”. In: (2017).

[3] Ryad Benosman et al. “Asynchronous Frameless Event-Based Optical Flow”. In: *Neural Netw.* 27 (Mar. 2012), pp. 32–37. DOI: 10.1016/j.neunet.2011.11.001. URL: https://doi.org/10.1016/j.neunet.2011.11.001.

[4] Tobias Brosch, Stephan Tschechne, and Heiko Neumann. “On event-based optical flow detection”. In: *Frontiers in Neuroscience* 9 (2015).

[5] Mark Buckler, Suren Jayasuriya, and Adrian Sampson. “Reconfiguring the imaging pipeline for computer vision”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 975–984.

[6] D. J. Butler et al. “A naturalistic open source movie for optical flow evaluation”. In: *European Conf. on Computer Vision (ECCV)*. Ed. by A. Fitzgibbon et al. (Eds.) Part IV, LNCS 7577. Springer-Verlag, Oct. 2012, pp. 611–625.

[7] Michael Calonder et al. “BRIEF: binary robust independent elementary features”. In: *Proceedings of the 11th European conference on Computer vision: Part IV*. ECCV’10. Heraklion, Crete, Greece: Springer-Verlag, 2010, pp. 778–792. URL: http://dl.acm.org/citation.cfm?id=1888089.1888148.

[8] Youngcheol Chae et al. “A 2.1 M Pixels, 120 Frame/s CMOS Image Sensor With Column-Parallel $\Delta\Sigma$ ADC Architecture”. In: *IEEE Journal of Solid-State Circuits* 46.1 (2010), pp. 236–247.

[9] Huaijin G Chen et al. “ASP vision: Optically computing the first layer of convolutional neural networks using angle sensitive pixels”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 903–912.

[10] Olivia Christie, Joshua Rego, and Suren Jayasuriya. “Analyzing Sensor Quantization Of Raw Images For Visual Slam”. In: *2020 IEEE International Conference on Image Processing (ICIP)*. 2020, pp. 246–250. DOI: 10.1109/ICIP40778.2020.9191352.

[11] Matthew A Clapp, Viktor Gruev, and Ralph Etienne-Cummings. “Focal-plane analog image processing”. In: *CMOS imagers*. Springer, 2004, pp. 141–202.

[12] Hugh Durrant-Whyte and Tim Bailey. “Simultaneous localization and mapping: part I”. In: *IEEE Robotics & Automation Magazine* 13.2 (2006), pp. 99–110.

[13] Felix Endres et al. “3-D Mapping With an RGB-D Camera”. In: 30.1 (2014), pp. 177–187.

[14] Jakob Engel, Thomas Schöps, and Daniel Cremers. “LSD-SLAM: Large-scale direct monocular SLAM”. In: *European Conference on Computer Vision*. Springer. 2014, pp. 834–849.

[15] Weikang Fang et al. “FPGA-based ORB Feature Extraction for Real-Time Visual SLAM”. In: *Proceedings of the IEEE International Conference on Field Programmable Technology*. 2017.

[16] Philipp Fischer et al. *FlowNet: Learning Optical Flow with Convolutional Networks*. 2015. DOI: 10.48550/ARXIV.1504.06852. URL: https://arxiv.org/abs/1504.06852.

[17] Guillermo Gallego et al. “Event-based Vision: A Survey”. In: *CoRR* abs/1904.08405 (2019). arXiv: 1904.08405. URL: http://arxiv.org/abs/1904.08405.

[18] Mathias Gehrig et al. “Dense Optical Flow from Event Cameras”. In: *CoRR* abs/2108.10552 (2021). arXiv: 2108.10552. URL: https://arxiv.org/abs/2108.10552.

[19] Berthold Horn. “Closed-form solution of absolute orientation using unit quaternions”. In: *Journal of the Optical Society of America A* 4.2 (1987), pp. 629–642.

[20] Yuhuang Hu, Shih-Chii Liu, and Tobi Delbruck. *v2e: From Video Frames to Realistic DVS Events*. 2021. arXiv: 2006.07722 [cs.CV].

[21] Max Jaderberg et al. “Spatial Transformer Networks”. In: *NIPS*. 2015.

[22] Hanme Kim, Stefan Leutenegger, and Andrew J. Davison. “Real-Time 3D Reconstruction and 6-DoF Tracking with an Event Camera”. In: *ECCV*. 2016.

[23] Georg Klein and David Murray. “Parallel Tracking and Mapping for Small AR Workspaces”. In: *Proceedings of the 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*. 2007.

[24] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. “ImageNet Classification with Deep Convolutional Neural Networks”. In: *Commun. ACM* 60.6 (May 2017), pp. 84–90. DOI: 10.1145/3065386. URL: https://doi.org/10.1145/3065386.

[25] Xavier Lagorce et al. “HOTS: A Hierarchy of Event-Based Time-Surfaces for Pattern Recognition”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.7 (2017), pp. 1346–1359. DOI: 10.1109/TPAMI.2016.2574707.

[26] Chankyu Lee et al. “Spike-FlowNet: Event-based Optical Flow Estimation with Energy-Efficient Hybrid Neural Networks”. In: (Mar. 2020).

[27] Xu Lei et al. “A Novel FastSLAM Framework Based on 2D Lidar for Autonomous Mobile Robot”. In: *Electronics* 9 (Apr. 2020), p. 695. DOI: 10.3390/electronics9040695.

[28] Ruihao Li, Sen Wang, and Dongbing Gu. “DeepSLAM: A Robust Monocular SLAM System with Unsupervised Deep Learning”. In: *IEEE Transactions on Industrial Electronics* (2020), pp. 1–1.

[29] Patrick Lichtsteiner, Christoph Posch, and Tobi Delbruck. “A 128$\times$128 120 dB 15μ s Latency Asynchronous Temporal Contrast Vision Sensor”. In: *IEEE Journal of Solid-State Circuits* 43.2 (2008), pp. 566–576.

[30] Runze Liu et al. “ESLAM: An energy-efficient accelerator for real-time ORB-SLAM on FPGA platform”. In: *Proceedings of the 56th Annual Design Automation Conference 2019*. 2019, pp. 1–6.

[31] Zhenhong Liu et al. “Ultra-low-power image signal processor for smart camera applications”. In: *Electronics Letters* 51.22 (2015), pp. 1778–1780.

[32] Bruce D. Lucas and Takeo Kanade. “An Iterative Image Registration Technique with an Application to Stereo Vision”. In: *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*. IJCAI’81. Vancouver, BC, Canada: Morgan Kaufmann Publishers Inc., 1981, pp. 674–679.

[33] Misha A. Mahowald and Carver Mead. “The Silicon Retina”. In: *Scientific American* 264.5 (1991), pp. 76–83. URL: http://www.jstor.org/stable/24936904.

[34] Yongguo Mei et al. “A Case study of mobile robot’s energy consumption and conservation techniques”. In: *Proceedings of 12th International Conference on Advanced Robotics*. July 2005.

[35] Elias Mueggler et al. “The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and SLAM”. In: *The International Journal of Robotics Research* 36.2 (Feb. 2017), pp. 142–149. DOI: 10.1177/0278364917691115. URL: https://doi.org/10.1177%2F0278364917691115.

[36] Peter Muller and Andreas Savakis. “Flowdometry: An Optical Flow and Deep Learning Based Approach to Visual Odometry”. In: *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2017, pp. 624–631. DOI: 10.1109/WACV.2017.75.

[37] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. “ORB-SLAM: a versatile and accurate monocular SLAM system”. In: *IEEE Transactions on Robotics* 31.5 (2015), pp. 1147–1163.

[38] Raul Mur-Artal and Juan D Tardós. “Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras”. In: *IEEE Transactions on Robotics* 33.5 (2017), pp. 1255–1262.

[39] L. Pan, M. Liu, and R. Hartley. “Single Image Optical Flow Estimation With an Event Camera”. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, June 2020, pp. 1669–1678. DOI: 10.1109/CVPR42600.2020.00174. URL: https://doi.ieeecomputersociety.org/10.1109/CVPR42600.2020.00174.

[40] Henri Rebecq et al. “EVO: A Geometric Approach to Event-Based 6-DOF Parallel Tracking and Mapping in Real Time”. In: *IEEE Robotics and Automation Letters* 2.2 (2017), pp. 593–600. DOI: 10.1109/LRA.2016.2645143.

[41] Henri Rebecq et al. “High Speed and High Dynamic Range Video with an Event Camera”. In: *IEEE Trans. Pattern Anal. Mach. Intell. (T-PAMI)* (2019). URL: http://rpg.ifi.uzh.ch/docs/TPAMI19_Rebecq.pdf.

[42] Henri Rebecq et al. “High Speed and High Dynamic Range Video with an Event Camera”. In: *CoRR* abs/1906.07165 (2019). arXiv: 1906.07165. URL: http://arxiv.org/abs/1906.07165.

[43] Whitman A Richards. “Lightness scale from image intensity distributions”. In: *Applied Optics* 21.14 (1982), pp. 2569–2582.

[44] E. Rosten, R. Porter, and T. Drummond. “Faster and Better: A Machine Learning Approach to Corner Detection”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.1 (Jan. 2010), pp. 105–119. DOI: 10.1109/tpami.2008.275. URL: https://doi.org/10.1109%2Ftpami.2008.275.

[45] Ethan Rublee et al. “ORB: An efficient alternative to SIFT or SURF”. In: *2011 International Conference on Computer Vision*. 2011, pp. 2564–2571. DOI: 10.1109/ICCV.2011.6126544.

[46] Cedric Scheerlinck, Nick Barnes, and Robert Mahony. “Continuous-time Intensity Estimation Using Event Cameras”. In: *Asian Conf. Comput. Vis. (ACCV)*. Dec. 2018, pp. 308–324. DOI: 10.1007/978-3-030-20873-8_20.

[47] Cedric Scheerlinck et al. “Fast Image Reconstruction with an Event Camera”. In: *IEEE Winter Conf. Appl. Comput. Vis. (WACV)*. 2020, pp. 156–163. DOI: 10.1109/WACV45572.2020.9093366.

[48] Jianbo Shi and Tomasi. “Good features to track”. In: *1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*. 1994, pp. 593–600. DOI: 10.1109/CVPR.1994.323794.

[49] J. Sturm et al. “A Benchmark for the Evaluation of RGB-D SLAM Systems”. In: *Proc. of the International Conference on Intelligent Robot Systems (IROS)*. Oct. 2012.

[50] Zachary Teed and Jia Deng. “RAFT: Recurrent All-Pairs Field Transforms for Optical Flow”. In: *CoRR* abs/2003.12039 (2020). arXiv: 2003.12039. URL: https://arxiv.org/abs/2003.12039.

[51] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics*. Cambridge, Mass.: MIT Press, 2005. URL: http://www.amazon.de/gp/product/0262201623/102-8479661-9831324?v=glance&n=283155&n=507846&s=books&v=glance.

[52] Carlo Tomasi and Takeo Kanade. *Detection and Tracking of Point Features*. Tech. rep. International Journal of Computer Vision, 1991.

[53] Rudy J Van de Plassche. *CMOS integrated analog-to-digital and digital-to-analog converters*. Vol. 742. Springer Science & Business Media, 2013.

[54] Antoni Rosinol Vidal et al. “Hybrid, Frame and Event based Visual Inertial Odometry for Robust, Autonomous Navigation of Quadrotors”. In: *CoRR* abs/1709.06310 (2017). arXiv: 1709.06310. URL: http://arxiv.org/abs/1709.06310.

[55] J. Wulff et al. “Lessons and insights from creating a synthetic optical flow benchmark”. In: *ECCV Workshop on Unsolved Problems in Optical Flow and Stereo Estimation*. Ed. by A. Fusiello et al. (Eds.) Part II, LNCS 7584. Springer-Verlag, Oct. 2012, pp. 168–177.

[56] Guang-Zhong Yang et al. “The grand challenges of Science Robotics”. In: *Science Robotics* 3.14 (2018), eaar7650. DOI: 10.1126/scirobotics.aar7650. eprint: https://www.science.org/doi/pdf/10.1126/scirobotics.aar7650. URL: https://www.science.org/doi/abs/10.1126/scirobotics.aar7650.

[57] Alex Zhu et al. “EV-FlowNet: Self-Supervised Optical Flow Estimation for Event-based Cameras”. In: June 2018. DOI: 10.15607/RSS.2018.XIV.062.

[58] Alex Zhu et al. “Unsupervised Event-based Learning of Optical Flow, Depth, and Egomotion”. In: (Dec. 2018).

[59] Alex Zihao Zhu et al. “Unsupervised Event-based Learning of Optical Flow, Depth, and Egomotion”. In: *CoRR* abs/1812.08156 (2018). arXiv: 1812.08156. URL: http://arxiv.org/abs/1812.08156.