EVENT-BASED VISUAL-INERTIAL ODOMETRY ON A FIXED-WING UNMANNED AERIAL VEHICLE

THESIS

Kaleb J Nelson, B.S.E.E., Captain, USAF

AFIT-ENG-MS-19-M-048

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT-ENG-MS-19-M-048

EVENT-BASED VISUAL-INERTIAL ODOMETRY ON A FIXED-WING UNMANNED AERIAL VEHICLE

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Electrical Engineering

Kaleb J Nelson, B.S.E.E., B.S.E.E.

Captain, USAF

March 21, 2019

AFIT-ENG-MS-19-M-048

EVENT-BASED VISUAL-INERTIAL ODOMETRY ON A FIXED-WING UNMANNED AERIAL VEHICLE

THESIS

Kaleb J Nelson, B.S.E.E., B.S.E.E.
Captain, USAF

Committee Membership:

Robert C Leishman, Ph.D
Chair

Scott L Nykl, Ph.D
Member

Scott R Graham, Ph.D
Member

Abstract

Event-based cameras are a new type of visual sensor that operate under a unique paradigm. These cameras provide asynchronous data on the log-level changes in light intensity for individual pixels, independent of other pixels' measurements. Through the hardware-level approach to change detection, these cameras can achieve microsecond fidelity, millisecond latency, ultra-wide dynamic range, and all with very low power requirements. The advantages provided by event-based cameras make them excellent candidates for visual odometry (VO) for unmanned aerial vehicle (UAV) navigation.

This document presents the research and implementation of an event-based visual-inertial odometry (EVIO) pipeline, which estimates a vehicle's 6-degrees-of-freedom (DOF) motion and pose utilizing an affixed event-based camera with an integrated Micro-Electro-Mechanical Systems (MEMS) inertial measurement unit (IMU). The front-end of the EVIO pipeline uses the current motion estimate of the pipeline to generate motion-compensated frames from the asynchronous event camera data. These frames are fed the back-end of the pipeline, which uses a Multi-State Constrained Kalman Filter (MSCKF) [1] implemented with Scorpion, a Bayesian state estimation framework developed by the Autonomy and Navigation Technology (ANT) Center at Air Force Institute of Technology (AFIT) [2]. This EVIO pipeline was tested on selections from the benchmark Event Camera Dataset [3]; and on a dataset collected, as part of this research, during the ANT Center's first flight test with an event-based camera.

Table of Contents

Page

List of Figures

Figure Page

List of Tables

EVENT-BASED VISUAL-INERTIAL ODOMETRY ON A FIXED-WING UNMANNED AERIAL VEHICLE

I. Introduction

1.1 Problem Background

Global Positioning System (GPS) signals and processing are critical to any modern navigation solution as, when fully functioning, no other single technology can currently achieve similar performance. However, this peak performance is dependent on receiving trustworthy, unobstructed signals from functioning satellites that remain in orbit. To mitigate the risk of dependence on GPS-only solutions in critical situations, the Autonomy and Navigation Technology (ANT) Center at Air Force Institute of Technology (AFIT), in partnership with Air Force Research Laboratory (AFRL), has invested in research into a variety of solutions that aim to work together to achieve similar performance as GPS-only solutions. One of these research avenues is visual odometry (VO), which utilizes visual sensors to estimate pose and motion over time.

1.1.1 Visual Odometry (VO)

VO solutions require cameras affixed to a vehicle to provide imagery that enables detection of unique landmarks in the camera's field of view. These landmarks need to be detected and tracked over multiple frames to estimate and/or correct the estimate of the motion of the vehicle relative to those landmarks. The performance of VO is limited by several aspects of the camera performance. A camera with a low sampling rate restricts visibility into movement occurring between frames, while a camera

with a high sampling rate requires more power for operation and for the back-end processing of increased amounts of data, even if the majority of the data is duplicative. Cameras can also suffer from blurring if the image capture time is too slow relative to the movement of the camera or scene. Lastly, due to the synchronous nature of their pixel's functionality, cameras are limited in their dynamic range; meaning they are restricted in the capability of adequately capturing very bright and very dim landmarks in the same image.

1.1.2 Event-Based Cameras

Event-based cameras are a new type of visual sensor that operate under a unique paradigm that helps address the shortfalls of classical cameras. These cameras provide asynchronous data on the log-level changes in light intensity for individual pixels, independent of other pixels' measurements. Through the hardware-level approach to change detection, these cameras can achieve microsecond fidelity, millisecond latency, and ultra-wide dynamic range, all with very low power requirements. Furthermore, the data rate is reflective of the changes in the scene. Rapid movement and/or highly textured scenes generate more data than slow movement and/or more uniform scenes.

Proven VO approaches, based on classical frame-based cameras, require adaption before incorporating the use of event-based cameras due to the unique operating paradigm of event-based cameras. Additionally, the relative technological immaturity, limited availability, and high-cost of the event-based cameras has hampered widespread investigation into event-based camera applications. However, new event-based camera companies are entering the market, improving the performance and availability of event-based cameras, potentially stimulating a broader adoption of the technology.

1.2 Research Objectives

The primary objective of this research is to estimate a vehicle's 6-degrees-of-freedom (DOF) motion and pose utilizing an affixed event-based camera with an integrated Micro-Electro-Mechanical Systems (MEMS) inertial measurement unit (IMU). The method consists of a front-end that generates frames from the event-based camera output and a back-end that uses an Extended Kalman Filter (EKF) to estimate the system states. This method is primarily inspired by Zhu's implementation of event-based visual-inertial odometry (EVIO) [4] which implemented Mourkis' Multi-State Constrained Kalman Filter (MSCKF) [1] for the back-end of the EVIO. This research, however, deviates from Zhu's approach by replacing the front-end that uses event-based feature tracking with probabilistic data association [5] with an implementation of Rebecq's motion compensation methodology [6], which uses the current motion estimation of the pipeline to generate motion-compensated frames from the asynchronous event-based camera data to feed to the back-end of the pipeline.

The ANT Center at AFIT has developed Scorpion, a plug-and-play Bayesian state estimation framework to facilitate rapid development of reusable Bayesian estimators [2]. The MSCKF implementation in this research utilizes Scorpion classes and functions, aiming to be a solid stepping stone for the addition of a robust implementation of the MSCKF to the Scorpion library as another tool for future navigation applications and research.

Another objective of this research is to collect event-based camera data during a flight test on a fixed-wing unmanned aerial vehicle (UAV). This is the first event-based camera flight test data collection for AFIT. Collecting an event-based camera dataset for EVIO research and analysis requires synchronized ground-truth, camera intrinsic calibration, and camera-IMU extrinsic calibration.

1.3 Document Overview

This document is organized as follows. Chapter I has laid out the problem this research aims to solve and discusses the objectives of the research and the software framework used in achieving those objectives. Chapter II provides an overview of relevant background information and literature references, including the foundation of VO and its applications, as well as the unique paradigm introduced with event-based cameras and recent state-of-the-art research into their use. Chapter III details the EVIO pipeline algorithm with a front-end generating motion-compensated frames feeding the back-end MSCKF. Chapter III also describes the methodology for calibrating event-based cameras and collecting data with event-based cameras on a flight test. Chapter IV presents the results of testing various stages of the pipeline with various parameters and the subsequent analysis, including tests on selections from the Event Camera Dataset published by the Robotics and Perception Group (RPG), Institute of Neuroinformatics (INI), University of Zürich (UZH) & Swiss Federal Institute of Technology (ETH) Zürich [3]; and on a dataset collected, as part of this research, on the ANT Center's first flight test with an event-based camera. Lastly, Chapter V discusses the conclusions drawn from the results, with emphasis on possible avenues of improvement to the methodology and alternative approaches that could better capitalize on the advantages of event-based cameras.

II. Background and Literature Review

This chapter will provide the foundational concepts and literature required for understanding subsequent chapters. It is organized as follows: Section 2.1 covers basics of visual odometry (VO), including the foundational methodology and a summary of recent developments. Section 2.2 describes the foundational paradigm and operating concept for event-based cameras and a summary of recent research with these sensors.

2.1 Visual Odometry (VO)

Visual odometry (VO), a term coined in 2004 by Nister [7], is the calculation of estimated egomotion of a platform (e.g. vehicle, robot, or human) relative to its environment using visual information from one or more cameras attached to the platform. Just as the Kalman Filter showcased its utility on the Apollo space launch missions in the 1960s [8], VO has proven useful in its implementation on the Mars rovers in the early 21st century [9, 10], which required tracking 6-degrees-of-freedom (DOF) motion on uneven and rocky terrain, an impossible feat for simple wheel odometry. Currently, VO is a useful supplement to other motion estimation sources like Global Positioning System (GPS) signals and inertial measurement units (IMUs) [11].

2.1.1 Camera Model

The foundation of VO requires modeling the geometry of the visual data. A simple and effective method used in the majority of VO methods is perspective projection with a pinhole camera model [11]. This model assumes that all light rays in the camera's field of view converge to a focal point and are projected onto a theoretical

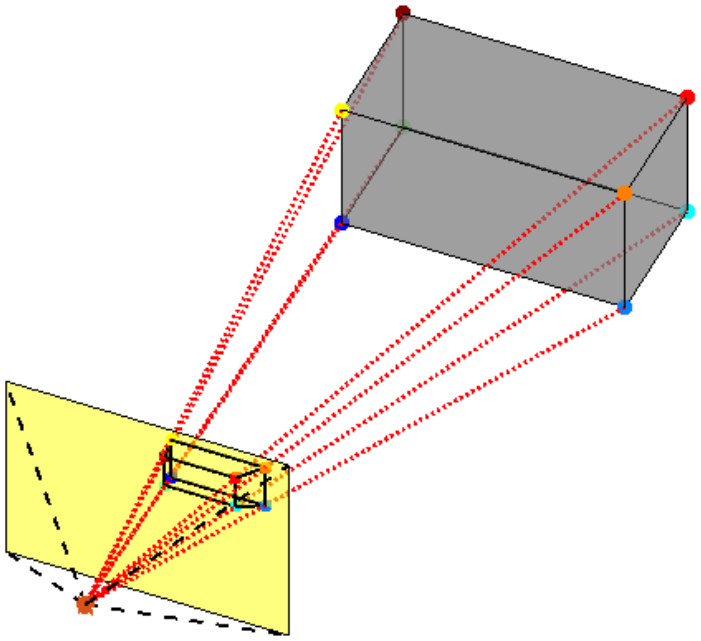

Figure 1: Perspective Pinhole Camera Model: objects in space are projected through a single focal point onto an image plane

two-dimensional plane, as shown in Figure 1. This translates three-dimensional world coordinates $[x, y, z]^T$ to pixel coordinates $[u, v]^T$ on the image plane via the perspective projection equation:

$$\frac{1}{z}\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{1}$$

where λ is the depth factor calculated as $1/z$, f_x and f_y are the focal lengths, and c_x and c_y are the coordinates of the center of the image plane [11].

When addressing distortion from the camera lens not resolved through a simple

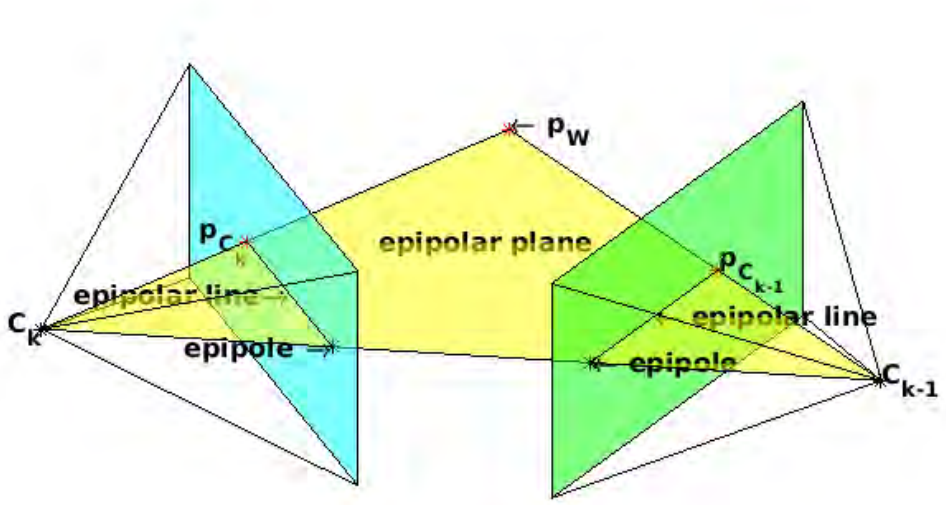

Figure 2: Epipolar geometry uses camera focal points and points in space seen in both images to establish an epipolar plane that defines the epipolar lines and epipoles that lie upon each image plane. This model provides the necessary geometric relations to define a matrix $\mathbf{T}_k^{k-1}$ to transform points between the (k) and $(k-1)$ frames of reference.

pinhole projection, the camera matrix K relates to the distorted pixel locations $[u_d, v_d]^T$ via

$$\lambda \begin{bmatrix} u_d \\ v_d \\ 1 \end{bmatrix} = K \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{2}$$

and the relation between distorted coordinates $[u_d, v_d]^T$ and undistorted coordinates $[u, v]^T$ is described by the polynomial model

$$\begin{bmatrix} u_d \\ v_d \end{bmatrix} = (1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \begin{bmatrix} u \\ v \end{bmatrix} + \begin{bmatrix} 2p_1 uv + p_2(r^2 + 2u^2) \\ p_1(r^2 + 2v^2) + 2p_2 uv \end{bmatrix} \tag{3}$$

, where $r^2 = u^2 + v^2$; k_1, k_2, and k_3 are the radial lens distortion coefficients; and p_1 and p_2 are the tangential distortion coefficients.

The distortion coefficients, focal lengths, and image plane center coordinates are calculated through the calibration method. Equations 1 and 3 are then used to translate measured pixel coordinates into homogeneous (i.e. depth-free) coordinates.

2.1.2 Epipolar Geometry

Epipolar geometry, as shown in Figure 2, is defined when a specific point in space is visible in two or more separate images. Epipolar geometry can be used to estimate the relative rigid body transformation between the camera coordinate frames at discrete time instants $k-1$ and k, defined as

$$\mathbf{T}_k^{k-1} = \begin{bmatrix} \mathbf{R}_k^{k-1} & t_k^{k-1} \\ 0 & 1 \end{bmatrix}, \tag{4}$$

where $\mathbf{R}_{k,k-1}$ is the direct cosine matrix (DCM) that rotates points in the k reference frame into the $k-1$ reference frame and where $t_{k,k-1}$ is the translation vector that describes the location of the focal point of C_k (i.e. origin of the k reference frame) in $k-1$ reference frame [11], so that

$$\begin{bmatrix} x_{k-1} \\ y_{k-1} \\ z_{k-1} \\ 1 \end{bmatrix} = \mathbf{T}_k^{k-1} \begin{bmatrix} x_k \\ y_k \\ z_k \\ 1 \end{bmatrix}, \tag{5}$$

or, similarly,

$$\begin{bmatrix} x_{k-1} \\ y_{k-1} \\ z_{k-1} \end{bmatrix} = \mathbf{R}_k^{k-1} \begin{bmatrix} x_k \\ y_k \\ z_k \end{bmatrix} + t_k^{k-1}. \tag{6}$$

2.1.3 Features

Unique identifying visual characteristics are required to identify points that can be matched in multiple images. Points with numerically identifiable image patterns are called features. There are many different types of features that can be extracted from images, generally categorized as corners or blobs. Corners identify the intersection of edges in the scene. Blobs describe an image pattern unique from the surrounding area with regard to its intensity and texture [12].

SIFT (scale invariant feature transform [13]) is a popular blob detector that blurs the image by convolving it with 2-D Gaussian kernels of various standard deviations, taking the differences between these blurred images and then identifying extrema [14]. Many other blob detectors follow a similar method, including SURF (speeded up robust features [15]), CENSURE (center surround extremas [16]), and KAZE (non-linear pyramid-based features [17]).

There are also a variety of corner feature detectors and descriptors. The FREAK (fast retina keypoint [18]) descriptor is used by the popular FAST (features from accelerated segment test [19]) and Harris (the Harris-Stephens algorithm [20]) detectors. The BRIEF (binary robust independent elementary features [21]) descriptor is used by the ORB (oriented FAST and rotated BRIEF [22]) detector. BRISK (binary robust invariant scalable keypoints [23]) is both a detector and descriptor that is a hybrid of both corner and blob feature detection and description.

2.1.4 Motion Estimation

After features are identified in the $k-1$ and k images, the task is to match them together to extract the motion of the camera. The homogeneous coordinates of a matched feature j in each image, described as

$$\bar{\mathbf{p}}_j^k = \begin{bmatrix} u_j^k \\ v_j^k \\ 1 \end{bmatrix}, \tag{7}$$

are related through

$$(\bar{\mathbf{p}}_j^k)^T E_k^{k-1} \bar{\mathbf{p}}_j^{k-1} \tag{8}$$

where E_k^{k-1} is the essential matrix that, up to a multiplicative scalar, is equivalent to

$$E_k^{k-1} \simeq \lfloor t_k^{k-1} \times \rfloor \mathbf{R}_k^{k-1} \tag{9}$$

where, if $t_k^{k-1} = [t_x, t_y, t_z]^T$, then we have a skew symmetric matrix equivalent to a 3D cross product:

$$\lfloor t_k^{k-1} \times \rfloor = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix}. \tag{10}$$

Groups of matched features between two images are used to calculate an estimate of the essential matrix from which an estimate of the rotation matrix $\mathbf{R}_k^{k-1}$ and translation vector t_k^{k-1} can be extracted [11]. However, since the essential matrix is only a scaled representation of the relation between two poses, other sources and/or methods are required to obtain properly scaled transformations.

One method of obtaining this scaling information is to use a pair of statically linked cameras (i.e. stereo vision) with a known and calibrated relative transformation

between them (i.e. baseline). Stereo vision, however, turns into monocular vision when the distance to the features is significantly larger than the distance between the stereo cameras. Depth information can be obtained for monocular vision by incorporating measurements from other sensors, i.e. accelerometers, gyroscopes, magnetometer, GPS, etc., and using the vision information to improve the accuracy of the estimates from those sensors.

Applying the above procedure to a sequence of images results in successive transformations between the coordinate frames of incremental camera poses. These transformations combined with timing information result in relative 6-DOF motion estimation, i.e. odometry.

2.1.5 Related Research

Researchers from all over the world are using these underlying principles to explore many avenues of research. The VO tutorials by Scaramuzza and Fraundorfer [11, 14] gives a respected foundational summary of concepts and research in this arena. Progress continues to be made in these areas with improved monocular approaches and algorithms [24, 25, 26, 27, 28] and increased research into vision navigation of micro-aerial vehicles [29, 30, 31].

A plethora of research has been done on attacking VO and visual tracking related problems using deep learning (e.g. convolutional neural networks) [32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43].

Other research has furthered progress with visual-inertial odometry (VIO) [1, 44, 45, 46, 47], where the loss of scale in visual information is addressed by incorporating IMU information.

2.1.6 Roadblocks

The quality of VO in high-speed scenarios is constrained by the temporal resolution of "frame-based" cameras (i.e. "normal" cameras capturing intensity images) due to the missed information between images from a limited frame rate. Increasing the frame rate to capture more of this information not only requires more operational power, but also more processing time and/or power to manipulate increased amounts of data since relevant odometry information could be found anywhere within the data.

Furthermore, even high-quality frame-based cameras suffer from some level of motion blur in high-speed scenarios. It is impossible to instantaneously measure the light intensity. Frame-based cameras require some length of integration time, i.e. exposure time or shutter speed. The exposure time also impacts the ability of the camera to capture both high-illuminated and low-illuminated objects in the same scene, i.e. high-dynamic range (HDR). Some HDR scenes can be effectively captured with multiple images with different exposure times, however this is not feasible with a moving camera.

These issues with VO also restrict the performance of other machine vision applications, like object recognition/tracking or robotic controls.

2.2 Event-Based Cameras

The development of event-based cameras begins with the growth of neuromorphic engineering, which aims to develop integrated circuit technology inspired by how effectively and efficiently biological brains and associated sensory inputs are able to interpret the natural world. Event-based cameras, also known as dynamic vision sensors (DVSs), silicon eyes or neuromorphic cameras, resulted from efforts to emulate the asynchronous nature of animal neurons and synapses [48]. These unique cameras are effective in addressing the common performance roadblocks with VO and other

machine vision applications [49, 50, 51, 52].

2.2.1 Operating Principle

Event-based cameras are a novel approach to sensing visual information. Arrays of complementary metal-oxide semiconductor (CMOS) pixels in these sensors each independently and continuously sense light intensity (see Figure 3). A log-level change in the light intensity at a given pixel triggers an asynchronous output of an event, e, characterized by a timestamp, t, the pixel coordinates, (x, y), and the event polarity,

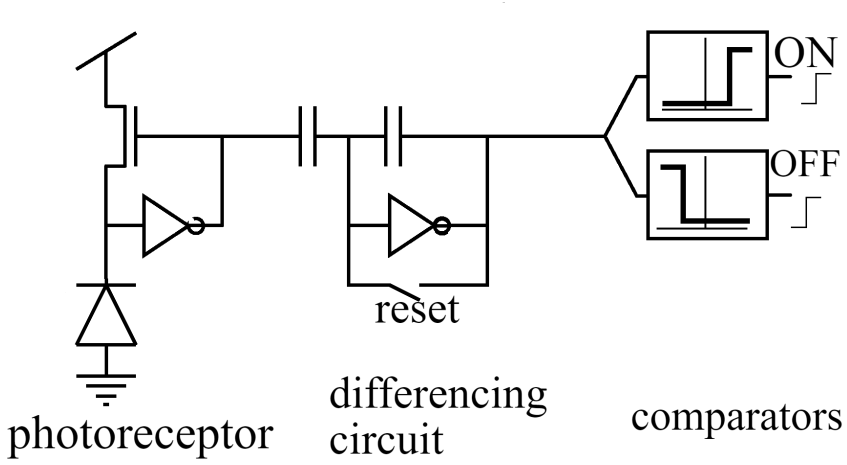

Figure 3: Abstract Schematic of a DVS pixel: the photoreceptor diode continuously senses the light intensity, the differencing circuit triggers at an selected threshold, and the comparators output (an ON or OFF signal) for a rise or fall in intensity [53]

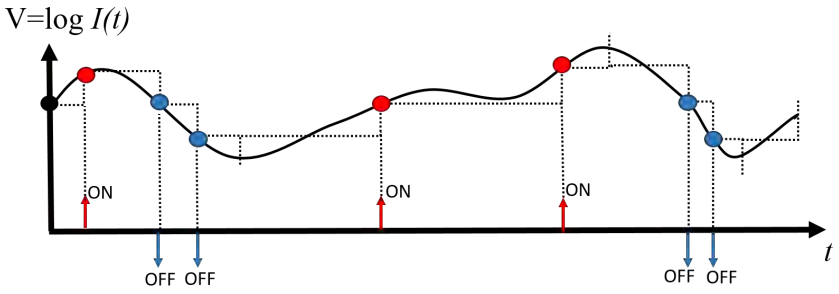

Figure 4: DVS Principle of Operation: each pixel measures the log-level intensity and outputs an ON or OFF event [53]

p (i.e. ON/OFF or rise/fall, see Figure 4) [53]. This is represented as a quadruplet of

$$e(t, x, y, p). \tag{11}$$

It can be noted that this type of information could certainly be calculated in post-processing from the output of any conventional digital camera. However, by utilizing a hardware approach to change detection, this sensor has the significant advantage of extremely high temporal resolution (microsecond precision with tens of microseconds reset) and latency on the order of a few milliseconds. The increased temporal resolution is shown in Figure 5. For a conventional camera to achieve similar temporal fidelity and latency, it would have to collect hundreds of thousands of frames per second and process the entirety of each frame, relative to the previous frame, within a millisecond.

Additionally, the independence of pixels in event-based cameras also enables a very high dynamic range of around 120 dB of light intensity. This means the sensor can capture both daylight illuminated items and the equivalent to full-moon-illuminated

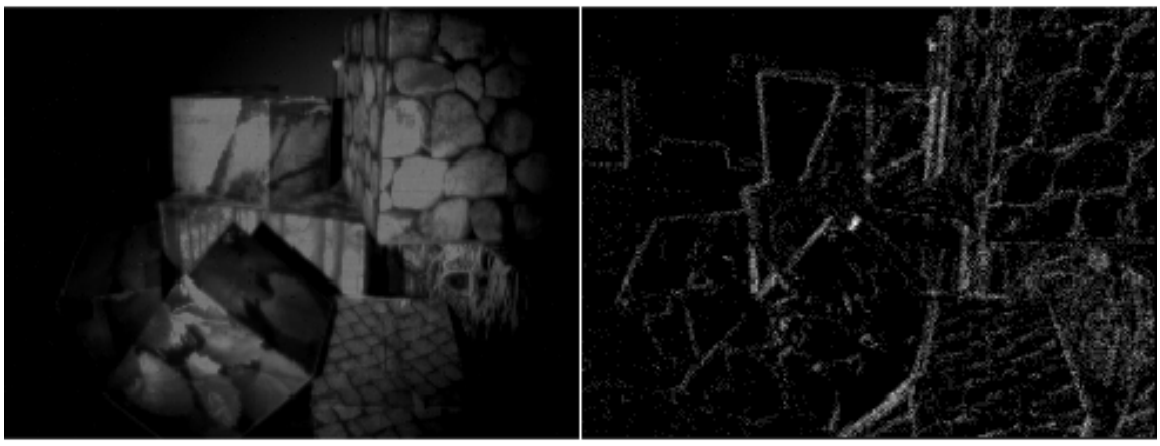

Figure 5: High Dynamic Range of DVS: the independent and asynchronous nature of event cameras on the right make it much more resilient dramatic difference in lighting in the same scene, where a classical camera is unable to capture the shadowed area

items simultaneously in its field of view. For example, a bright flashlight or even direct sunlight won't produce lens flare since neighboring pixels, not directly measuring the bright light, will be able to continue detecting changes in their portions of the scene with no impact on performance, as seen in Figure 5.

The output from a 128x128 resolution event-based camera has, on average, a very low bandwidth requirement on the order of hundreds of KB/s due to only collecting a relatively sparse portion of visual information in its field of view. Higher bandwidth requirements, on the order of 1-10 MB/s, can temporarily result from extremely rapid camera movement in close proximity to the environment or sudden changes in a scene like an explosion. Much lower bandwidth requirements, on the order of tens of KB/s, can be achieved during long-term overnight surveillance scenarios or space observations [54]. For comparison, a high-speed camera that matches an event-based camera's temporal resolution with 1MHz frame rate with only 1/16 the resolution of a 128x128 event-based camera requires a constant 1.5 GB/s. This is 100-1000 times more than the highest bandwidth requirement from an event-based camera [55].

The high temporal fidelity of the data as well as the impact of rapid and slow camera movement on the amount of events are illustrated in Figure 6.

2.2.2 Types of Event-based Cameras

The DVS and dynamic and active-pixel vision sensor (DAVIS) are event-based cameras developed at the Institute of Neuroinformatics (INI), University of Zürich (UZH) & Swiss Federal Institute of Technology (ETH) Zürich. They have a form factor similar to a web camera and utilize Universal Serial Bus (USB) for communication and power [53, 56]. The DVS produces event data from a 128x128 pixel array and operates at around 20 mW. The DAVIS 240C is an improved version that produces event data from a 240x180 pixel array while also outputting greyscale frames

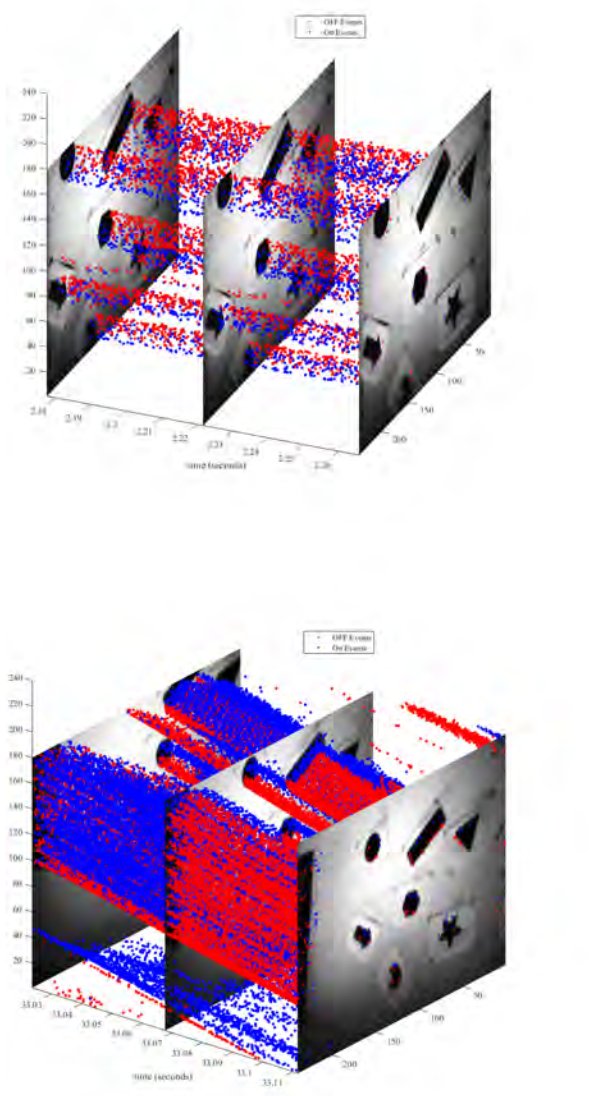

Figure 6: DVS Events vs. Images. DVS events are shown as scatter plots, with ON/OFF events representing rise/fall in intensity.

procured from integrated conventional camera pixels (i.e. active pixel sensor (APS)) as well as 6-DOF information from a built-in IMU. The DVS and DAVIS are still only available as research prototypes, sold through the company iniVation. The company Insightness sells a device similar to the DAVIS combined with evaluation kits used

for micro-aerial vehicle navigation and collision avoidance. Starting fall of 2018, the DVS was no longer available for purchase and an improved DAVIS 346 sensor with a 346x260 pixel array became available.

Another event-based camera was developed around the same time as the DVS. It is called the Asynchronous Time-based Image Sensor (ATIS) [57, 58]. It incorporates asynchronous intensity measurement circuitry at each pixel, which provides a slightly different operating paradigm. The ATIS provides not only a polarity (i.e. rise/fall) for each event but also an asynchronously sampled light intensity for that specific pixel. While this provides more relevant information and an opportunity to more succinctly encode intensity information, issues arise with its adaptive integration time which, for low light pixels, can take up to 2 seconds. This integration can be interrupted by a new event, making it effectively blind to quickly moving features in low light scenarios [48] [59].

Prophesee and Pixium Vision, both companies based out of Paris, France, have based their technology on the ATIS sensor. Prophesee has developed the commercially available ONBOARD reference system, with manufacturing quality assurance (QA) as their apparent targeted market. Pixium Vision has focused on the medical community with bionic vision restoration systems supporting age-related macular degeneration (AMD).

It must be noted that no event-based camera is mature enough to compete with the price point and image or video quality of conventional charge-coupled device (CCD) cameras. However, a DVS sensor does, at a low energy cost, provide visual information at a very high temporal fidelity that can be quickly processed independent from human interaction.

2.2.3 Current Research with Event-Based Cameras

The current research utilizing this sensor primarily capitalizes on its many advantages, i.e. the small form factor, low power requirements, high temporal resolution, high dynamic range, and relatively low data bandwidth. A variety of disciplines take advantage of its unique operating paradigm [60, 61], including support for the blind [62, 63], and space situational awareness (SSA) [54].

2.2.3.1 Visualization Methods

Since event-camera data is so inherently different from APS cameras, significant research has been done on various approaches to visualizing the information. This includes reconstructing scene intensity, utilizing the fact that DVS events are effectively samples of the log-level gradient of the light intensity [64, 65, 66, 67, 68, 69]. Others have built on the method of creating simulated images by a simple summation of events at each pixel over some variable time period by improving the clarity of those images through compensating for camera motion [70, 71]. Unique simulated images have been developed by incorporating the timing information of the events to affect the magnitude of value at each pixel [72, 73]. Further efforts have delved into developing 3-D reconstruction and depth maps using stereo DVS [74, 75, 76, 77, 78, 79, 80].

2.2.3.2 Filtering and Analysis Methods

Due to the nature of these sensors, there is also occasional background activity (BA) due to leaky currents or thermal noise in the pixel circuity. Several research efforts have proposed solutions to address this issue [81, 82, 83, 84]. Other research has introduced differing methods of analysis and filtering [82, 85, 86, 87].

2.2.3.3 Feature and Object Detection and Tracking

A foundational concept of almost all machine vision objectives is the ability to detect and associate features and objects in a scene. Due to the uniqueness of event-camera data, several efforts have focused on the detection of features particular to this asynchronous paradigm. There are a wide variety of approaches to this problem [88], with no definitive ideal solution. Some efforts focus on detecting lines [89] or corners [90, 91, 92] in the event data. Others incorporate the timing information into the feature detection and description [93, 94, 73]. Further research has been done generating optical flow from event-based cameras [95, 96, 97, 68, 98, 99, 100, 101] and identifying features within that optical flow [102]. Other research identifies features in the intensity images and uses the event-based camera data to track those features between frames [103].

Object identification/classification and tracking, which is tightly coupled with feature tracking, has also had significant improvements [104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114], though primarily with focus on a stationary camera and movement occurring in the field of view. Some of these efforts have incorporated machine learning [115, 116, 117], which helps address the significant paradigm shift required for working with this type of information.

Significant success with these processes has been showcased with gesture recognition [118, 119], including efforts using IMB's neuromorphic chip, TrueNorth [120], as well as micro particle tracking [121], star-tracking [122], and slot car racing [123].

2.2.3.4 Motion Estimation

Building off the foundation of estimating the position and motion of objects seen by a camera, other research has addressed estimating egomotion, or motion of the viewing body. Some efforts focused on improving angular velocity estimation

(i.e. no translational movement) with event-based camera data [65, 124, 125]. For full 6-DOF Event-based Visual Odometry (EVO), early research focused on map-based localization [126, 127] and/or utilizing depth information [128, 129]. Progress continued with research into a variety of methods only utilizing event-based camera data [130, 131, 132, 67, 133, 134, 135, 136]. More recent research has incorporated IMU information to accomplish event-based visual-inertial odometry (EVIO) [4, 76], with state-of-the-art Hybrid-EVIO utilizing both event-based camera data and intensity images [137], improving robustness in both stationary and rapid-movement scenarios.

2.2.3.5 Datasets

The limited availability of event-based cameras, due to their current high cost, is an obstacle to widespread investigations into applications for these unconventional sensors. To alleviate this issue, several research organizations produced event-camera datasets, supporting performance evaluation of available algorithms. The Robotics and Perception Group (RPG) at INI has made available a dataset with a couple dozen scenarios collected with a DAVIS camera. They include indoor and outdoor environments viewing simple shapes, varied surfaces, or just a simple office. The datasets include both simple rotational or translational movement as well as more erratic rapid movements. All collections include events, camera calibration information, IMU, and the frame images from the integrated CCD camera. For the indoor collections in controlled environments, they also include ground truth data from a motion capture system [3].

A more recent dataset, the Multi-Vehicle Event Camera Dataset, is provided by the General Robotics, Automation, Sensing and Perception (GRASP) Laboratory at the University of Pennsylvania. It includes stereo event-camera data and intensity images fused with LIDAR, IMU, motion capture, and GPS, making pose and depth

available as ground truth baselines [138]. There is also the Poker-DVS dataset focused on identification of playing cards [139], the RoShamBo (Rock-Paper-Scissors game) DVS Dataset for human gesture recognition [140], as well as simulators [3] and emulators/converters [141, 142] to more easily create custom datasets.

III. Methodology

Preamble

The methodology described in this chapter is primarily inspired by Zhu's use of the Multi-State Constrained Kalman Filter (MSCKF) [1] as the back-end of an event-based visual-inertial odometry (EVIO) pipeline [4]. Section 3.4 contains a detailed description of the MSCKF. The front-end used in this work diverges from Zhu's EVIO implementation by incorporating Rebecq's motion-compensated event images [76], with the implementation described in Section 3.3. The MSCKF was implemented utilizing MATLAB and the Autonomy and Navigation Technology (ANT) Center's Scorpion library described in Section 3.2. These methods were tested on select datasets from Robotics and Perception Group (RPG)'s Event-Camera Dataset, mentioned previously in Section 2.2.3, as well as on a dataset collected with a dynamic and active-pixel vision sensor (DAVIS) 240C event camera on a flight test utilizing a fixed wing unmanned aerial vehicle (UAV) conducted on the airfield at Camp Atterbury, Indiana. Additional details on the format of RPG's Event-Camera Dataset are described in Section 3.5.1. Details on the format and the methodology used to collect the Camp Atterbury dataset, including the calibration methods used, are described in Section 3.5.2.

3.1 Notation

To clarify the notation used throughout this methodology description, $\mathbf{p}_k^B$ is the position of point k in the frame B, $\mathbf{p}_A^B$ is the position of the origin of frame A in the frame of B, and $\mathbf{R}_A^B$ is the direct cosine matrix (DCM) that rotates points from frame A to frame B so that

$$\mathbf{p}_k^B = \mathbf{R}_A^B(\mathbf{p}_k^A) + \mathbf{p}_A^B. \tag{12}$$

Quaternions are in the Jet Propulsion Laboratory (JPL) format [143], which is

$$\begin{bmatrix} q_w & q_x & q_y & q_z \end{bmatrix}^T \equiv q_w + q_x\mathbf{i} + q_y\mathbf{j} + q_z\mathbf{k} \tag{13}$$

where

$$\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1. \tag{14}$$

$\mathbf{0}_{M\times N}$ is a null, or zero, matrix with M rows and N columns. $\mathbf{I}_N$ is an identity matrix of dimension N.

For a given position $\mathbf{p}$ in frame A, i.e.

$$\mathbf{p}^A = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \tag{15}$$

then the skew symmetric matrix is defined as

$$\lfloor \mathbf{p}^A \times \rfloor = \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix} \tag{16}$$

3.2 Scorpion Library

The Scorpion library is a Bayesian state estimation framework developed by the ANT Center at Air Force Institute of Technology (AFIT) [2] to more easily develop sensor modules and system descriptions used to estimate system states through

stochastic estimation. This framework can be used on multiple developmental environments including Python, Kotlin, Java or within MATLAB. The MATLAB implementation is used in this work. Scorpion provides a clear structure for developing custom state blocks, sensors and measurements in addition to having "off-the-shelf" modular components available for commonly used system states like position, velocity, orientation and system biases as well as commonly used sensors like cameras, Global Positioning System (GPS), and rangefinders.

3.3 Motion-Compensated Event Frames

Visual odometry (VO) requires identifying, locating, and matching features in the camera's field of view. In order to capitalize on the many feature identification and matching methods discussed in Section 2.1.3, this work combined batches of events into high-contrast greyscale images. The simplest approach to generating images from events is through event integration, which accumulates events at each pixel location over some time window or for some contiguous batch of events. The polarity characteristic of each event (i.e. whether it was a rise or fall in light intensity) as well as the specific timing of the event provided by the camera is discarded. This generates an image that highlights changes in light intensity (i.e. intensity gradient). However, these images do not provide an instantaneous sample of the light intensity gradient, as the nature of event-based cameras is capturing the change in light intensity over time. This creates some issues with generating quality visualization depending on the speed of movement of the camera. If too short of a time window and/or too few events are used, which can easily occur with a slowly moving camera, an integrated image can be too sparse and effectively noisy, resulting in insufficient information about the scene, as illustrated in the top right image of Figure 7. If the time window is too long or too many events are selected for the speed of the camera and the richness of the

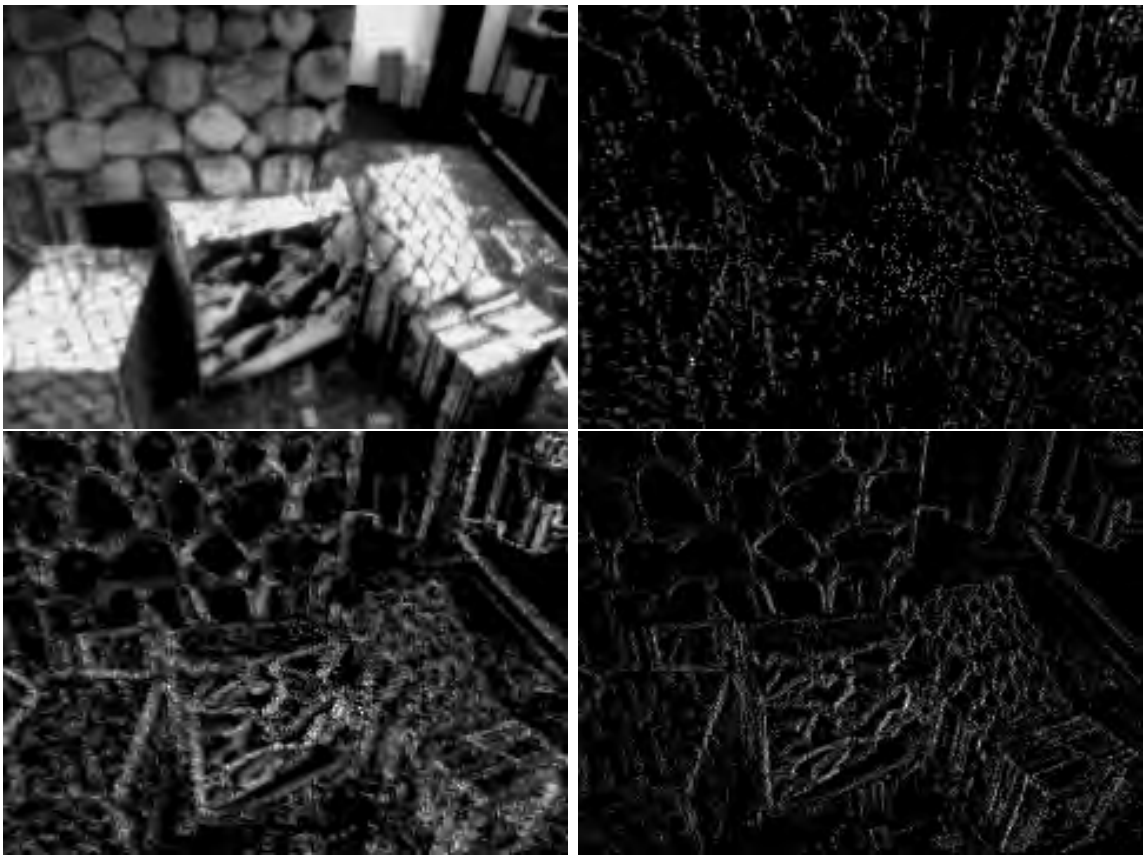

Figure 7: Integrated vs. Motion-Compensated Event Frames. Top left: intensity image. Top right: integrated event frame with 5,000 events. Bottom left integrated event frame with 50,000 events. Bottom right: motion-compensated event frame with 50,000 events

scene, then the integrated image has significant motion blur, as shown in the bottom left frame of Figure 7.

A motion-compensated event image uses an alternative method, described in [76] and inspired by [124], which takes into account the specific event time and the estimated pose to generate event images that compensate for camera movement, resulting in images with stronger contrasting edges.

A motion-compensated event frame is constructed using the INS measurements and timestamps, through mechanization [144], and the timestamps of the individual events. The process is to estimate the actual physical location of each event in the

world frame and then to project the locations back onto a single camera image frame that occurs (in time) either before or after the event occurred, as if all the events had been "taken" at the same time. A start time is selected, which corresponds to some camera pose from the mechanization and is designated as the targeted reference frame. Utilizing the interpolated camera pose at each event's time (obtained through inertial navigation system (INS) mechanization), each event's location in the navigation frame is estimated using a best guess for depth. The guess for depth can come from averaging and/or interpolating the depth of tracked features or simply using some other external measurement like a barometer. The event location in the navigation frame is then transformed into the targeted reference frame and, through camera intrinsics, projected onto a pixel location. The events at each pixel are accumulated then scaled by the maximum number of events at any pixel, resulting in a sparse greyscale image, shown in the bottom right frame of Figure 7. This method is fully described in Algorithm 1.

Improvements in speed can be obtained by precomputing a look-up table for the undistorted homogeneous coordinates for each pixel location and by vectorizing the frame-of-reference and camera intrinsic transformations. Additionally, due to the events' microsecond fidelity, further efficiency can be obtained by rounding event timestamps to some nearest Δt (e.g. 10 µs or 100 µs or some other value, defined as a system parameter) and reusing interpolated reference frame transforms for identical timestamps. This helps since the interpolation can be computationally expensive. However, it can also result in less accurate estimations with a large Δt.

Algorithm 1 Motion-Compensated Event Image

function MCIMAGE($e, ins, filt, N$)
  ▷ N events in e, at pixel locations $[e.x, e.y]$ at time $e.t$ ▷ INS ins and Kalman Filter $filt$
  $\mathbf{R}_W^{C_0}, \mathbf{p}_{C_0}^W \leftarrow ins.\text{GETPOSE}(e_0.t)$ ▷ Target pose for motion-compensated events
  for $k \leftarrow 1, N$ **do** ▷ Loop over all events
    $\begin{bmatrix} x_H \\ y_H \end{bmatrix} \leftarrow \text{UNDISTHOMOGPOS}(e_k.x, e_k.y)$ ▷ Pixel location to position
    $z \leftarrow filt.depth(e_k.t)$ ▷ Expected depth of features
    $\mathbf{p}_k^{C_k} \leftarrow \begin{bmatrix} x_H \\ y_H \\ 1 \end{bmatrix} z$ ▷ Estimated 3-D event position
    $\mathbf{R}_W^{C_k}, \mathbf{p}_{C_k}^W \leftarrow ins.getPose(e_k.t)$ ▷ Camera pose at time $e_k.t$
    $\mathbf{R}_{C_k}^W \leftarrow (\mathbf{R}_W^{C_k})^T$
    $\mathbf{p}_k^W \leftarrow \mathbf{p}_{C_k}^W + \mathbf{R}_{C_k}^W(\mathbf{p}_k^{C_k})$ ▷ Transform into navigation frame
    $\begin{bmatrix} x_k^{C_0} \\ y_k^{C_0} \\ z_k^{C_0} \end{bmatrix} \leftarrow \mathbf{R}_W^{C_0}(\mathbf{p}_k^W - \mathbf{p}_{C_0}^W)$ ▷ Transform into target frame
    $\begin{bmatrix} x_{Hk}^{C_0} \\ y_{Hk}^{C_0} \\ 1 \end{bmatrix} \leftarrow \frac{1}{z_k^{C_0}} \begin{bmatrix} x_k^{C0} \\ y_k^{C_0} \\ z_k^{C_0} \end{bmatrix}$ ▷ Homogeneous coordinates
    $\begin{bmatrix} x_k^i \\ y_k^i \\ 1 \end{bmatrix} \leftarrow K \begin{bmatrix} x_{Hk}^{C0} \\ y_{Hk}^{C_0} \\ 1 \end{bmatrix}$ ▷ Position to undistorted image pixel location
    $image(x_k^i, y_k^i) \leftarrow image(x_k^i, y_k^i) + 1$ ▷ Accumulate for image
  end for
  $image \leftarrow \frac{image}{\text{MAX}(image)}$
  return $image$ ▷ The motion-compensated image
end function

3.4 Multi-State Constraint Kalman Filter

The MSCKF was introduced by Mourikis [1] and implemented with event-based cameras in Zhu's EVIO [4]. The MSCKF implemented in this research is primarily inspired by the description in [1], though with several adjustments made to improve integration with AFIT's ANT Center's Scorpion library (see Section 3.2). The essence of the MSCKF is to update vehicle states from features tracked across multiple frames.

3.4.1 Primary States

The primary propagated states are

$$X_{IMU} = \begin{bmatrix} \mathbf{p}_I^G \\ \mathbf{v}_I^G \\ \bar{q}_G^I \\ \mathbf{b}_a \\ \mathbf{b}_g \end{bmatrix} \tag{17}$$

where $\mathbf{p}_I^G$ is the position of the inertial measurement unit (IMU), in meters, relative to the north-east-down (NED) navigation reference frame, $\{G\}$; $\mathbf{v}_I^G$ is the velocity of the IMU, in meters per second, relative to the navigation reference frame; $\bar{q}_G^I$ is the IMU orientation relative to the navigation reference frame described as a quaternion that rotates vectors from the navigation frame to the IMU reference frame $\{I\}$; and $\mathbf{b}_a$ and $\mathbf{b}_g$ are the accelerometer biases and gyroscope biases, respectively.

The primary error states are ordered to match the "off-the-shelf" Pinson-15 state block available in the Scorpion library, which is a 15-state representation of the error model of an INS in a localized NED frame [144] organized as

$$\tilde{X}_{IMU} = \begin{bmatrix} \tilde{\mathbf{p}}_I^G \\ \tilde{\mathbf{v}}_I^G \\ \delta\boldsymbol{\theta} \\ \tilde{\mathbf{b}}_a \\ \tilde{\mathbf{b}}_g \end{bmatrix}, \tag{18}$$

where $\tilde{\mathbf{p}}_I^G$, $\tilde{\mathbf{v}}_I^G$, $\tilde{\mathbf{b}}_a$, and $\tilde{\mathbf{b}}_g$ are the errors for position, velocity, accelerometer bias, and gyroscope bias. $\delta\theta$ is the tilt error associated with the error quaternion, $\delta\bar{q}$, which

corrects the estimated rotation, $\hat{\bar{q}}$, to the true rotation $\bar{q}$, i.e. $\bar{q} = \delta\bar{q} \otimes \hat{\bar{q}}$, where $\otimes$ is quaternion multiplication [1]. For small values of $\delta\theta$, a decent approximation of $\delta\bar{q}$ can be efficiently estimated as

$$\delta\bar{q} \simeq \begin{bmatrix} 1 \\ \frac{1}{2}\delta\theta \end{bmatrix}. \tag{19}$$

Normalizing the magnitude of $\delta\bar{q}$ (so that $q_w^2 + q_x^2 + q_y^2 + q_z^2 = 1$) resulting from the $\delta\theta$ approximation in (19) ensures there are no unintended magnitude changes during quaternion correction. Further accuracy in calculating $\delta\bar{q}$ can be obtained at a loss of efficiency by fully converting the tilt errors, represented in radians as $(\theta_x, \theta_y, \theta_z)$ for rotations around the X, Y, and Z axes respectively for roll, pitch, and yaw, into the quaternions [145] via

$$\begin{bmatrix} q_w \\ q_x \\ q_y \\ q_z \end{bmatrix} = \begin{bmatrix} \cos(\theta_z)\cos(\theta_y)\cos(\theta_x) + \sin(\theta_z)\sin(\theta_y)\sin(\theta_x) \\ \cos(\theta_z)\cos(\theta_y)\sin(\theta_x) - \sin(\theta_z)\sin(\theta_y)\cos(\theta_x) \\ \sin(\theta_z)\cos(\theta_y)\sin(\theta_x) + \cos(\theta_z)\sin(\theta_y)\cos(\theta_x) \\ \sin(\theta_z)\cos(\theta_y)\cos(\theta_x) - \cos(\theta_z)\sin(\theta_y)\sin(\theta_x) \end{bmatrix}. \tag{20}$$

3.4.2 State Augmentation

For each new image, the current camera pose (orientation $\bar{q}_G^{C_i}$ and position $\mathbf{p}_{C_i}^{\tilde{W}}$) is calculated using the IMU pose estimate (from the INS mechanization output with corrections from the extended Kalman Filter (EKF)'s current error states) by

$$\bar{q}_G^{C_i} = \bar{q}_I^C \otimes \bar{q}_G^{I_i} \tag{21}$$

and

$$\mathbf{p}_{C_i}^W = \mathbf{p}_{I_i}^G + \mathbf{R}_{I_i}^G(\mathbf{p}_C^I), \tag{22}$$

where $\bar{q}_I^C$ is the quaternion representing the rotation from the IMU reference frame to the camera reference frame, and $\mathbf{p}_C^I$ is the position of the camera in the IMU reference frame.

$\bar{q}_G^{I_i}$ is the quaternion representing the rotation from the navigation frame to the IMU frame with a DCM of $\mathbf{R}_G^{I_i}$, with $\mathbf{R}_{I_i}^G = (\mathbf{R}_G^{I_i})^T$ as the DCM for the rotation from the IMU frame to the navigation frame.

Each of these poses are augmented onto the IMU state so that the system state vector is

$$X_k = \begin{bmatrix} X_{IMU_k} \\ \bar{q}_G^{C_1} \\ \mathbf{p}_{C_1}^W \\ \bar{q}_G^{C_2} \\ \mathbf{p}_{C_2}^W \\ \vdots \\ \bar{q}_G^{C_{i-1}} \\ \mathbf{p}_{C_{i-1}}^W \\ \bar{q}_G^{C_i} \\ \mathbf{p}_{C_i}^W \end{bmatrix}. \tag{23}$$

The associated error state, $\delta\theta_{C_i}$ and $\tilde{\mathbf{p}}_{C_i}^W$, for this pose is augmented onto the

existing states

$$\tilde{X}_k = \begin{bmatrix} \tilde{X}_{IMU_k} \\ \delta\theta_{C_1} \\ \tilde{\mathbf{p}}_{C_1}^W \\ \delta\theta_{C_2} \\ \tilde{\mathbf{p}}_{C_2}^W \\ \vdots \\ \delta\theta_{C_{i-1}} \\ \tilde{\mathbf{p}}_{C_{i-1}}^W \\ \delta\theta_{C_i} \\ \tilde{\mathbf{p}}_{C_i}^W \end{bmatrix} \tag{24}$$

The covariance matrix is also augmented through

$$\mathbf{P}_{k|k} \leftarrow \begin{bmatrix} \mathbf{I}_{6N+15} \\ \mathbf{J} \end{bmatrix} \mathbf{P}_{k|k} \begin{bmatrix} \mathbf{I}_{6N+15} \\ \mathbf{J} \end{bmatrix}^T \tag{25}$$

where

$$\mathbf{J} = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{R}_I^C & \mathbf{0}_{3\times6} & \mathbf{0}_{3\times6N} \\ \mathbf{I}_3 & \mathbf{0}_{3\times3} & \lfloor \mathbf{R}_{C_i}^W \mathbf{p}_C^I \times \rfloor & \mathbf{0}_{3\times6} & \mathbf{0}_{3\times6N} \end{bmatrix}. \tag{26}$$

Due to the structure of the Scorpion `SensorEKF` class used to track the IMU states, these error states and their associated covariances are saved as separate variables–not added as additional state blocks–for future filter propagation and updates.

3.4.3 Detect and Match Features

Features are detected in each image utilizing detectors available in MATLAB, including SURF [15] or KAZE [17] blob detectors or FAST [19] or Harris [20] corner

detectors. The image is separated into individual regions and features are detected in each region. If the MSCKF, due to memory constraints, allows a maximum of N features to be detected in each image and the image is broken into a 4 by 4 grid of regions, then the strongest $N/16$ features in each of the 16 regions would be used for matching. This ensures that if weak features need to be discarded, then the remaining features are generally still spread out over the entire image, providing a wider field of information to use in motion estimation.

These new features are first matched to tracked features, which are features detected in a sufficient number of images to be included in the EKF update described in Section 3.4.6. If new features do not match any tracked features, they are matched to candidate features, which are recently detected features with too few samples to be included in the EKF update. New features not matched to existing tracked or candidate features are added to the candidate features list, retaining the features descriptor for future matching. Candidate features not seen in some set number of images (e.g. 2 or 3) are discarded. Pose and features relations are tracked as explained in Section 3.4.7.

3.4.4 Propagating States

During propagation, the camera pose states, $\bar{q}_G^{C_i}$ and $\mathbf{p}_{C_i}^W$, with their associated error states, $\delta\theta_{C_i}$ and $\tilde{\mathbf{p}}_{C_i}^W$, and covariances, remain unchanged.

The primary states, described in Equation 17, are propagated with an instantiation of an INS mechanization object, using the accelerometer and gyroscope measurements from the IMU. This INS mechanization takes as input ΔVs and $\Delta\theta$s, i.e. discrete changes in velocity and Euler angles, which are the rotation about each three-dimensional axis. These come from scaling the IMU measurements by the applicable Δt. The INS mechanization calculations are accomplished in a local-level NED frame

that is initialized with a latitude-longitude-altitude (LLA) that enables the INS object to internally account for gravity, removing the need to separately account for gravity when passing on IMU measurements.

The primary error states, $\tilde{X}_{IMU_k}$, and the associated covariance matrix, $\mathbf{P}_{II_{k|k}}$, at initial time t_k are propagated to $\tilde{X}_{IMU_{k+1}}$ and $\mathbf{P}_{II_{k+1|k}}$ at a new time t_{k+1} using the "off-the-shelf" measurement model provided by the Scorpion Pinson-15 error state block, which simply requires the current solution and force from the INS mechanization object and the new time to propagate to, t_{k+1}.

The overall system covariance, $\mathbf{P}_{k|k}$, is then

$$\mathbf{P}_{k|k} = \begin{bmatrix} \mathbf{P}_{II_{k|k}} & \mathbf{P}_{IC_{k|k}} \\ \mathbf{P}_{IC_{k|k}}^T & \mathbf{P}_{CC_{k|k}} \end{bmatrix}, \tag{27}$$

where $\mathbf{P}_{CC_{k|k}}$ is the covariance matrix for the camera pose error states and $\mathbf{P}_{IC_{k|k}}$ is the correlation between the primary error states and the camera pose error states.

Using the state transition matrix, $\mathbf{\Phi}(t_{k+1}, t_k)$, generated from the Pinson-15 state block established in the Scorpion `SensorEKF` filter object, the overall system covariance is propagated as

$$\mathbf{P}_{k+1|k} = \begin{bmatrix} \mathbf{P}_{II_{k+1|k}} & \mathbf{\Phi}(t_{k+1}, t_k)\mathbf{P}_{IC_{k|k}} \\ \mathbf{P}_{IC_{k|k}}^T\mathbf{\Phi}(t_{k+1}, t_k)^T & \mathbf{P}_{CC_{k|k}} \end{bmatrix}. \tag{28}$$

Note that $\mathbf{P}_{CC_{k|k}}$ does not change during propagation, and, since

$$(\mathbf{P}_{IC_{k|k}}^T\mathbf{\Phi}(t_{k+1}, t_k)^T)^T = \mathbf{\Phi}(t_{k+1}, t_k)\mathbf{P}_{IC_{k|k}}, \tag{29}$$

only $\mathbf{\Phi}(t_{k+1}, t_k)\mathbf{P}_{IC_{k|k}}$ requires calculation outside of the Scorpion `SensorEKF` filter object during propagation.

3.4.5 Least Squares Estimate (LSE)

The 3D location of each feature, expressed in the NED navigation frame, must be known before that feature can be used in the measurement update process in Section 3.4.6. We estimate the 3D position of the feature using a least-squares estimation approach, which utilizes Montiel's intersection approach [146] with an inverse-depth parametrization [147], as briefly described as follows.

The 3D position of the jth feature, expressed in the C_ith camera frame, where it is observed, is

$$\mathbf{p}_j^{C_i} = \mathbf{R}_{C_n}^{C_i}(\mathbf{p}_j^{C_n}) + \mathbf{p}_{C_n}^{C_i} \tag{30}$$

where C_n is the camera pose at the earliest observation of the feature with $\mathbf{R}_{C_n}^{C_i}$ and $\mathbf{p}_{C_n}^{C_i}$ as the rotation and translation between C_n and C_i. The position of feature j in the C_nth frame is defined as

$$\mathbf{p}_j^{C_n} = \begin{bmatrix} X_j^{C_n} \\ Y_j^{C_n} \\ Z_j^{C_n} \end{bmatrix} = Z_j^{C_n} \begin{bmatrix} \left(\frac{X_j^{C_n}}{Z_j^{C_n}}\right) \\ \left(\frac{Y_j^{C_n}}{Z_j^{C_n}}\right) \\ \left(\frac{1}{Z_j^{C_n}}\right) \end{bmatrix} = Z_j^{C_n} \begin{bmatrix} \alpha_j \\ \beta_j \\ \rho_j \end{bmatrix}. \tag{31}$$

Equation 30 is then rewritten as

$$\mathbf{p}_j^{C_i} = Z_j^{C_n} \left(\mathbf{R}_{C_n}^{C_i} \begin{bmatrix} \left(\frac{X_j^{C_n}}{Z_j^{C_n}}\right) \\ \left(\frac{Y_j^{C_n}}{Z_j^{C_n}}\right) \\ 1 \end{bmatrix} + \frac{1}{Z_j^{C_n}} \mathbf{p}_{C_n}^{C_i} \right) \tag{32}$$

$$= Z_j^{C_n} \left(\mathbf{R}_{C_n}^{C_i} \begin{bmatrix} \alpha_j \\ \beta_j \\ 1 \end{bmatrix} + \rho_j \mathbf{p}_{C_n}^{C_i} \right) \tag{33}$$

$$= Z_j^{C_n} \begin{bmatrix} h_{i1}(\alpha_j, \beta_j, \rho_j) \\ h_{i2}(\alpha_j, \beta_j, \rho_j) \\ h_{i3}(\alpha_j, \beta_j, \rho_j) \end{bmatrix} \tag{34}$$

where

$$h_{i1}(\alpha_j, \beta_j, \rho_j) = r_{i,1,1}\alpha_j + r_{i,1,2}\beta_j + r_{i,1,3} + \rho_j X_i \tag{35}$$

$$h_{i2}(\alpha_j, \beta_j, \rho_j) = r_{i,2,1}\alpha_j + r_{i,2,2}\beta_j + r_{i,2,3} + \rho_j Y_i \tag{36}$$

$$h_{i3}(\alpha_j, \beta_j, \rho_j) = r_{i,3,1}\alpha_j + r_{i,3,2}\beta_j + r_{i,3,3} + \rho_j Z_i, \tag{37}$$

using

$$\mathbf{R}_{C_n}^{C_i} = \begin{bmatrix} r_{i,1,1} & r_{i,1,2} & r_{i,1,3} \\ r_{i,2,1} & r_{i,2,2} & r_{i,2,3} \\ r_{i,3,1} & r_{i,3,2} & r_{i,3,3} \end{bmatrix} \tag{38}$$

and

$$\mathbf{p}_{C_i}^{C_n} = \begin{bmatrix} X_i \\ Y_i \\ Z_i \end{bmatrix}. \tag{39}$$

The feature's normalized location measurement from each frame, meaning the (x, y) portion of the homogeneous location of feature j in the C_ith reference frame, not image (pixel) coordinates, is also rewritten as

$$\mathbf{z}_{ij} = \frac{1}{h_{i3}(\alpha_j, \beta_j, \rho_j)} \begin{bmatrix} h_{i1}(\alpha_j, \beta_j, \rho_j) \\ h_{i2}(\alpha_j, \beta_j, \rho_j) \end{bmatrix} = \begin{bmatrix} g_{ij1}(\alpha_j, \beta_j, \rho_j) \\ g_{ij2}(\alpha_j, \beta_j, \rho_j) \end{bmatrix} \tag{40}$$

where

$$g_{ij1}(\alpha_j, \beta_j, \rho_j) = \frac{h_{i1}(\alpha_j, \beta_j, \rho_j)}{h_{i3}(\alpha_j, \beta_j, \rho_j)} \tag{41}$$

and

$$g_{ij2}(\alpha_j, \beta_j, \rho_j) = \frac{h_{i2}(\alpha_j, \beta_j, \rho_j)}{h_{i3}(\alpha_j, \beta_j, \rho_j)}. \tag{42}$$

These descriptions are then used in a least-squares minimization, with the error as

$$e_i = \hat{\mathbf{z}}_{ij} - \mathbf{z}_{ij} \tag{43}$$

where $\mathbf{z}_{ij}$ is the feature's measured normalized location and $\hat{\mathbf{z}}_{ij}$ is the normalized location result from Equation 40 after iteratively optimizing the estimated quantities $(\hat{\alpha_j}, \hat{\beta}_j, \hat{\rho}_j)$ using

$$\begin{bmatrix} \Delta\alpha_j \\ \Delta\beta_j \\ \Delta\rho_j \end{bmatrix} = H^{-1}b \tag{44}$$

where

$$H = \sum_{i=n}^{Mj} (J_i(\alpha_j, \beta_j, \rho_j))^{-1} \Omega J_i(\alpha_j, \beta_j, \rho_j), \tag{45}$$

$$b^T = \sum_{i=n}^{Mj} e_i \Omega J_i(\alpha_j, \beta_j, \rho_j). \tag{46}$$

The information matrix Ω is

$$\Omega = \begin{bmatrix} \sigma^2_{\mathtt{im}} & 0 \\ 0 & \sigma^2_{\mathtt{im}} \end{bmatrix}^{-1}, \tag{47}$$

and the Jacobian is

$$J_i(\alpha_j, \beta_j, \rho_j) = \begin{bmatrix} \frac{\partial g_{ij1}}{\partial \alpha_j} & \frac{\partial g_{ij1}}{\partial \beta_j} & \frac{\partial g_{ij1}}{\partial \rho_j} \\ \frac{\partial g_{ij2}}{\partial \alpha_j} & \frac{\partial g_{ij2}}{\partial \beta_j} & \frac{\partial g_{ij2}}{\partial \rho_j} \end{bmatrix} \tag{48}$$

where

$$\frac{\partial g_{ij1}}{\partial \alpha_j} = \frac{r_{i,1,1}}{r_{i,3,3} + \alpha_j r_{i,3,1} + \beta_j r_{i,3,2} + Z_i \rho_j} - \frac{r_{i,3,1}(r_{i,1,3} + \alpha_j r_{i,1,1} + \beta_j r_{i,1,2} + X_i \rho_j)}{(r_{i,3,3} + \alpha_j r_{i,3,1} + \beta_j r_{i,3,2} + Z_i \rho_j)^2} \tag{49}$$

$$\frac{\partial g_{ij1}}{\partial \beta_j} = \frac{r_{i,1,2}}{r_{i,3,3} + \alpha_j r_{i,3,1} + \beta_j r_{i,3,2} + Z_i \rho_j} - \frac{r_{i,3,2}(r_{i,1,3} + \alpha_j r_{i,1,1} + \beta_j r_{i,1,2} + X_i \rho_j)}{(r_{i,3,3} + \alpha_j r_{i,3,1} + \beta_j r_{i,3,2} + Z_i \rho_j)^2} \quad (50)$$

$$\frac{\partial g_{ij1}}{\partial \rho_j} = \frac{X_i}{r_{i,3,3} + \alpha_j r_{i,3,1} + \beta_j r_{i,3,2} + Z_i \rho_j} - \frac{Z_i(r_{i,1,3} + \alpha_j r_{i,1,1} + \beta_j r_{i,1,2} + X_i \rho_j)}{(r_{i,3,3} + \alpha_j r_{i,3,1} + \beta_j r_{i,3,2} + Z_i \rho_j)^2} \quad (51)$$

$$\frac{\partial g_{ij2}}{\partial \alpha_j} = \frac{r_{i,2,1}}{r_{i,3,3} + \alpha_j r_{i,3,1} + \beta_j r_{i,3,2} + Z_i \rho_j} - \frac{r_{i,3,1}(r_{i,2,3} + \alpha_j r_{i,2,1} + \beta_j r_{i,2,2} + Y_i \rho_j)}{(r_{i,3,3} + \alpha_j r_{i,3,1} + \beta_j r_{i,3,2} + Z_i \rho_j)^2} \quad (52)$$

$$\frac{\partial g_{ij2}}{\partial \beta_j} = \frac{r_{i,2,2}}{r_{i,3,3} + \alpha_j r_{i,3,1} + \beta_j r_{i,3,2} + Z_i \rho_j} - \frac{r_{i,3,2}(r_{i,2,3} + \alpha_j r_{i,2,1} + \beta_j r_{i,2,2} + Y_i \rho_j)}{(r_{i,3,3} + \alpha_j r_{i,3,1} + \beta_j r_{i,3,2} + Z_i \rho_j)^2} \quad (53)$$

$$\frac{\partial g_{ij2}}{\partial \rho_j} = \frac{Y_i}{r_{i,3,3} + \alpha_j r_{i,3,1} + \beta_j r_{i,3,2} + Z_i \rho_j} - \frac{Z_i(r_{i,2,3} + \alpha_j r_{i,2,1} + \beta_j r_{i,2,2} + Y_i \rho_j)}{(r_{i,3,3} + \alpha_j r_{i,3,1} + \beta_j r_{i,3,2} + Z_i \rho_j)^2} \quad (54)$$

Each iteration generates a new values for $\Delta\alpha_j$, $\Delta\beta_j$, and $\Delta\rho_j$ to be applied to $\hat{\alpha}_j$, $\hat{\beta}_j$, and $\hat{\rho}_j$ on the next iteration. Once $\Delta\alpha_j$, $\Delta\beta_j$, and $\Delta\rho_j$ fall below some defined threshold, $\hat{\alpha}_j$, $\hat{\beta}_j$, and $\hat{\rho}_j$ are sufficiently optimized to generate and least-squares estimate (LSE) of the the position of the feature in the navigation frame as

$$\hat{\mathbf{p}}_j^G = \frac{1}{\hat{\rho}_j} \mathbf{R}_{C_n}^G \begin{bmatrix} \hat{\alpha}_j \\ \hat{\beta}_j \\ 1 \end{bmatrix} + \mathbf{p}_{C_n}^G \quad (55)$$

to be used in the update state described in Section 3.4.6.

3.4.6 Updating States

3.4.6.1 Update Trigger

An update to the system error states, which aims to improve the accuracy of the pose error states, occurs in one of two cases:

- When one or more features are no longer detected after a certain number of frames (set as a tunable parameter), these features are used in an EKF update then subsequently discarded. After this type of update, each pose is checked to

see if there are any other remaining features in those poses. If there are none, then that pose and the corresponding error states are discarded.

- When the number of retained poses reaches some set maximum, $N_{\max}$. In this case, select poses (and associated information) are discarded after utilizing their measurement information. This methodology followed the practice in Mourikis' MSCKF in discarding one third of all poses, starting with the second oldest pose and removing every third pose.

3.4.6.2 Measurement Model

For each feature j to be used in the update and for each pose C_i where that feature is detected, the LSE position in the navigation frame from Section 3.4.5, $\hat{\mathbf{p}}_{f_j}^G$, is used to generate the LSE location in each camera frame

$$\hat{\mathbf{p}}_j^{C_i} = \mathbf{R}_G^{C_i}(\hat{\mathbf{p}}_{f_j}^G) = \begin{bmatrix} \hat{X}_j^{C_i} \\ \hat{Y}_j^{C_i} \\ \hat{Z}_j^{C_i} \end{bmatrix}. \tag{56}$$

The normalized coordinates of the LSE location in each camera frame is

$$\hat{\mathbf{z}}_{ij} = \begin{bmatrix} \frac{\hat{X}_j^{C_i}}{\hat{Z}_j^{C_i}} \\ \frac{\hat{Y}_j^{C_i}}{\hat{Z}_j^{C_i}} \end{bmatrix}. \tag{57}$$

The measurement residual is then computed through

$$\mathbf{r}_{ij} = \mathbf{z}_{ij} - \hat{\mathbf{z}}_{ij} \tag{58}$$

where $\hat{\mathbf{z}}$ and $\mathbf{z}$ are the LSE and measured normalized coordinates of the feature in the given pose's reference frame described in Section 3.4.5.

This measurement residual is approximated [1] as

$$\mathbf{r}_{ij} \simeq \mathbf{H}_{\mathbf{X}_{ij}} \hat{\mathbf{X}} + \mathbf{H}_{f_{ij}} \tilde{\mathbf{p}}_j^G + \mathbf{n}_{ij} \tag{59}$$

where $\mathbf{n}_{ij}$ is the zero-mean, white Gaussian measurement noise; the Jacobians are

$$\mathbf{H}_{f_{ij}} = \mathbf{J}_{ij} \mathbf{R}_G^{C_i}, \tag{60}$$

and

$$\mathbf{H}_{\mathbf{X}_{ij}} = \begin{bmatrix} \mathbf{0}_{2\times 15} & \mathbf{0}_{2\times 6} & \dots & \mathbf{0}_{2\times 6} & \mathbf{J}_{ij} \lfloor \hat{\mathbf{p}}_j^{C_i} \times \rfloor & -\mathbf{H}_{f_{ij}} & \mathbf{0}_{2\times 6} & \dots & \mathbf{0}_{2\times 6} \end{bmatrix} \tag{61}$$

with the Jacobian matrix as

$$\mathbf{J}_{ij} = \frac{1}{\hat{Z}_j^{C_i}} \begin{bmatrix} 1 & 0 & -\frac{\hat{X}_j^{C_i}}{\hat{Z}_j^{C_i}} \\ 0 & 1 & -\frac{\hat{Y}_j^{C_i}}{\hat{Z}_j^{C_i}} \end{bmatrix}. \tag{62}$$

To note,

$$\begin{bmatrix} \mathbf{J}_{ij} \lfloor \hat{\mathbf{p}}_j^{C_i} \times \rfloor & -\mathbf{H}_{f_{ij}} \end{bmatrix} \tag{63}$$

is the Jacobian with respect to pose i.

Vertically concatenating the residuals $\mathbf{r}_{ij}$ into $\mathbf{r}_j$ and the Jacobians $\mathbf{H}_{\mathbf{X}_{ij}}$ and $\mathbf{H}_{f_{ij}}$ into $\mathbf{H}_{\mathbf{X}_j}$ and $\mathbf{H}_{f_j}$ results in

$$\mathbf{r}_j \simeq \mathbf{H}_{\mathbf{X}_j} \tilde{\mathbf{X}}_j + \mathbf{H}_{f_j} \tilde{\mathbf{p}}_j^G + \mathbf{n}_j \tag{64}$$

where $\tilde{\mathbf{X}}_j$ are the relevant error states, $\mathbf{n}_j$ has a covariance matrix of $\mathbf{R}_j = \sigma_{\mathtt{im}}^2 \mathbf{I}_{2M_j}$ with M_j as the number of poses that contain the jth feature.

In order to utilize this relationship in the EKF update, a new residual is required

to avoid the issue of how $\tilde{\mathbf{p}}_j^G$ is correlated with $\tilde{\mathbf{X}}_j$ due to the state estimate, $\mathbf{X}$, being used to calculate the LSE feature position $\hat{\mathbf{p}}_j^{C_i}$. This new residual, $\mathbf{r}_{0j}$, is defined as

$$\mathbf{r}_{0j} = \mathbf{H}_{0j}\tilde{\mathbf{X}}_j + \mathbf{n}_{0j} \tag{65}$$

where

$$\mathbf{H}_{0j} = \mathbf{A}^T H_{X_j} \tag{66}$$

and

$$\mathbf{r}_{0j} = \mathbf{A}^T r_j \tag{67}$$

where $\mathbf{A}$ is the U portion of a singular value decomposition of $\mathbf{H}_{f_j}$, so that $\mathbf{r}_{0j}$ is formed by projecting r_j on the left nullspace of $\mathbf{H}_{f_j}$.

3.4.6.3 Update Equations

Further concatenating $\mathbf{r}_{0j}$ and $\mathbf{H}_{0j}$ from each feature in the update into $\mathbf{r}_0$ and $\mathbf{H}_0$ results in

$$\mathbf{r}_0 = \mathbf{H}_0\tilde{\mathbf{X}} + \mathbf{n}_0 \tag{68}$$

Since $\mathbf{H}_0$ can be exceptionally large, computational complexity is reduced by taking the QR decomposition of $\mathbf{H}_0$, denoting it as

$$\mathbf{H}_0 = \begin{bmatrix} \mathbf{Q}_1 & \mathbf{Q}_2 \end{bmatrix} \begin{bmatrix} \mathbf{T}_H \\ \mathbf{0} \end{bmatrix} \tag{69}$$

where $\mathbf{Q}_1$ and $\mathbf{Q}_2$ are unitary matrices whose columns form bases for the range and nullspace of $\mathbf{H}_0$ and $\mathbf{T}_H$ is an upper triangular matrix.

This enables Equation 68 to be rewritten as

$$\mathbf{r}_0 = \begin{bmatrix} \mathbf{Q}_1 & \mathbf{Q}_2 \end{bmatrix} \begin{bmatrix} \mathbf{T}_H \\ \mathbf{0} \end{bmatrix} \tilde{\mathbf{X}} + \mathbf{n}_0 \tag{70}$$

which can become

$$\begin{bmatrix} \mathbf{Q}_1^T \mathbf{r}_0 \\ \mathbf{Q}_2^T \mathbf{r}_0 \end{bmatrix} = \begin{bmatrix} \mathbf{T}_H \\ \mathbf{0} \end{bmatrix} \tilde{\mathbf{X}} + \begin{bmatrix} \mathbf{Q}_1^T \mathbf{n}_0 \\ \mathbf{Q}_2^T \mathbf{n}_0 \end{bmatrix} \tag{71}$$

Since $\mathbf{Q}_2^T \mathbf{r}_0$ is just noise ($\mathbf{Q}_2^T \mathbf{n}_0$), a new residual is defined using this decomposition as

$$\mathbf{r}_n = \mathbf{Q}_1 \mathbf{r}_0 = \mathbf{T}_H \tilde{\mathbf{X}} + \mathbf{Q}_1^T \mathbf{n}_0 \tag{72}$$

where $\mathbf{Q}_1^T \mathbf{n}_0$ is a noise vector with covariance $\mathbf{R}_n = \sigma_{\mathtt{im}}^2 \mathbf{I}_r$, where r is the number of columns in $\mathbf{Q}_1$.

The EKF update can then be completed with a Kalman gain

$$\mathbf{K} = \mathbf{P}\mathbf{T}_H^T (\mathbf{T}_H \mathbf{P} \mathbf{T}_H^T + \mathbf{R}_n)^{-1}, \tag{73}$$

a state correction

$$\Delta \mathbf{X} = \mathbf{K}\mathbf{r}_n, \tag{74}$$

and an updated state covariance matrix

$$\mathbf{P}_{k+1|k+1} = (\mathbf{I}_{6N+15} - \mathbf{K}\mathbf{T}_H)\mathbf{P}_{k+1|k}(\mathbf{I}_{6N+15} - \mathbf{K}\mathbf{T}_H)^T + \mathbf{K}\mathbf{R}_n\mathbf{K}^T. \tag{75}$$

3.4.7 Bookkeeping

Due to the fact that features or poses are not permanently tracked and that features and poses can be somewhat arbitrarily related, a clear and efficient method

of tracking which features correspond to which poses is required. An $N_{\max}$ x $M_{\max}$ logical relation index matrix is established, due to the efficiency of MATLAB logical indexing; $N_{\max}$ is the maximum number of retained poses, and $M_{\max}$ is the maximum number of retained features. Additionally, at any point, only the NxM upper left submatrix is used, where N is the current number of retained poses, and M is the current number of retained features. $N_{\max}$, $M_{\max}$, N, and M are also used to define the sizes and relevant portions of the variables storing feature descriptors, poses, and pose errors.

When a pose or feature is removed because it no longer provides relevant information, the retained features/pose and related logical indices are reordered so all are maintained consecutively in the first N or M elements of a relevant list (e.g. the first M feature descriptors) or in the first NxM submatrix of a relevant matrix (i.e. the logical relation matrix).

3.5 Datasets

Multiple datasets collected under a variety of scenarios were utilized to evaluate the performance of the VO pipeline. The RPG Event-Camera Dataset [3] discussed in Section 3.5.1 is a public dataset utilized by many recent research efforts [135, 66, 124, 4, 125, 76, 137]. The Camp Atterbury dataset discussed in Section 3.5.2 is a contribution of this work and is the pioneering use of Event-Based Cameras for AFIT's ANT Center and is the first known instance of a flight test on-board a fixed-wing UAV.

3.5.1 RPG UZH Event Camera Dataset

The Robotics and Perception Group (RPG) at University of Zürich (UZH) has been at the forefront of event-based camera research. As discussed in Section 2.2.3,

they released the Event Camera Dataset [3]. The calibration parameters are provided as a text file with $(f_x, f_y, c_x, c_y, k_1, k_2, p_1, p_2, k_3)$ which provides the camera matrix parameters and distortion coefficients explained in Section 2.1.1. The DAVIS output is available either as text files for events, IMU measurements, and image timestamps with a folder of ".png" images or as a rosbag binary file compatible with Robot Operating System (ROS), a multi-purpose robotic software framework. The company iniVation has also provided a toolbox on GitHub [148] for MATLAB that can convert rosbag binary files to an custom "address-event data" (.aedat) binary file with tools to easily accomplish various tasks, including loading custom selections of the data into a MATLAB `struct`, reorienting the images or event data, normalizing timestamps off the first timestamp, and various visualizing/plotting methods.

3.5.2 Camp Atterbury Flight Test

An additional contribution of this thesis is an event-based camera dataset captured with a DAVIS attached to the belly of a 14-foot fixed wing Aeroworks 100CC Carbon CUB UAV. Other sensors also recording on the same flights were 1280x960 color camera running at around 30 frames per second, a Piksi multi-GNSS module, a magnetometer, and a Pixhawk autopilot flight controller. These sensors enabled high-fidelity ground truth data for the flight as well as generating multiple datasets to support future research efforts at the ANT Center.

The other sensor drivers on board the UAV all used Lightweight Communications and Marshalling (LCM) messaging protocols, where messages are passed through a publish/subscribe paradigm optimized for real-time systems with high-bandwidth, low latency requirements. Due to the prototype nature of the event-based camera, this was the first known instance of using LCM messaging with event-camera data. Adjustments were made to the popular dynamic vision sensor (DVS) ROS package

developed by RPG [133, 53] to enable publishing event-camera data in LCM. Future efforts could directly utilize `libcaer`, a minimal C library to access, configure and get data from the DAVIS, to broadcast event-camera LCM messages with possibly lower latency.

3.5.2.1 Calibration

The DAVIS camera and its integrated IMU were calibrated using the Kalibr visual-inertial calibration toolbox [149, 150, 151, 152, 153].

The camera intrinsic calibration took advantage of the integrated greyscale frame-based camera output. Images were taken of a checkerboard pattern, like the one shown in Figure 8.

The Kalibr tools were used to automatically find the corners in each image. These corners were used to generate estimates of the camera calibration parameters discussed in Section 2.1.1. These parameters were then optimized across all images with sufficient number of identified corners.

The camera matrix and distortion parameters were then used to by Kalibr's camera-IMU calibration tool to estimate the camera-IMU extrinsic calibration, which is the transformation matrix between the camera's reference frame and the IMU reference frame.

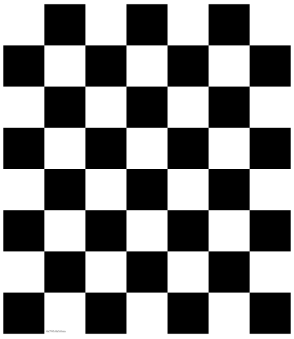

Figure 8: Calibration Checkerboard: the easily identifiable corners with multiple straight lines in this pattern are used to estimate camera matrix and distortion coefficient parameters

IV. Results and Analysis

4.1 Preamble

Using the methodology described in Chapter III, the event frame front-end and Multi-State Constrained Kalman Filter (MSCKF) back-end were developed and tested separately prior to merging into one cohesive event-based visual-inertial odometry (EVIO) pipeline. The motion-compensated frames from various datasets using ground truth motion information are shown in Section 4.2. The results of the MSCKF using virtual feature points and standard greyscale frames are shown in Section 4.7.

The back-end MSCKF needs viable features fed from the front-end frames. The results of testing various types of feature detection are shown in Section 4.3.

To ensure the initial event frames are properly motion-compensated until a filter update, the accelerometer and gyroscope biases for each dataset were manually estimated, as shown in Section 4.6.

While calibration parameters were provided with the Robotics and Perception Group (RPG) Event Camera Dataset [3], the Camp Atterbury dataset required independent calibration. The results of the camera calibration using the Kalibr toolbox are shown in Section 4.5.

The results of final pipeline, including both motion-compensated event frames and the MSCKF, are shown in Section 4.7.

4.2 Motion-Compensated Event Frames

The RPG's Shapes dataset [3] is a simple scenario to easily verify and troubleshoot compensating for motion. The sample greyscale image in Figure 9 shows how this dataset only has a handful of high-contrast features with distinct edges. The impact of motion compensation on the quality of the event frames is shown in Figure 10. With

Figure 9: Greyscale Image Sample from Shapes dataset: simple black images on a white background

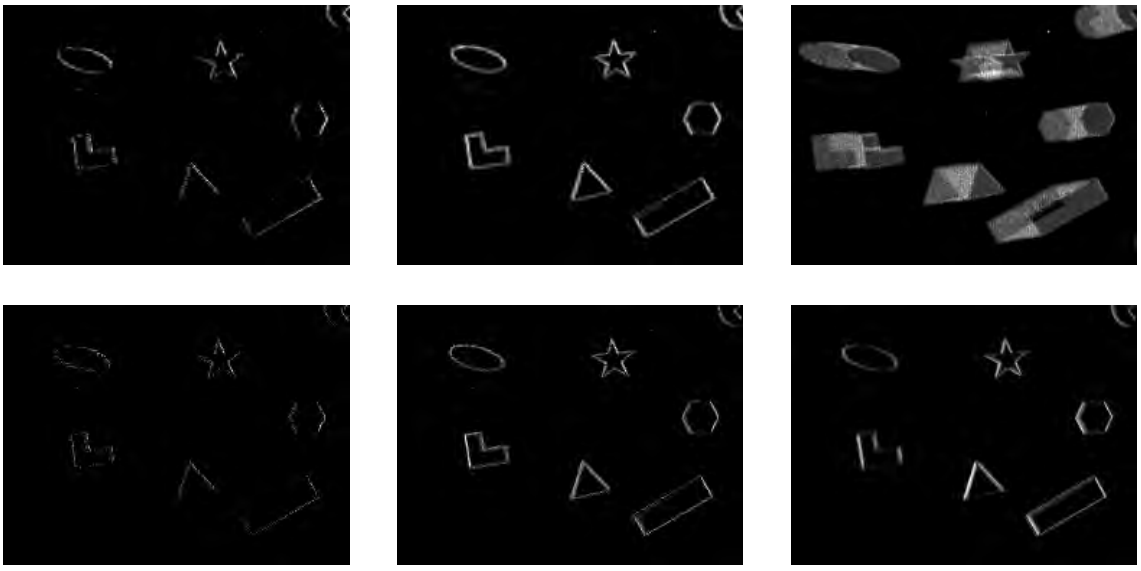

Figure 10: Shapes dataset: Motion-Compensated With Ground Truth. The top three images are created with only simple accumulation with 1,000, 5,000, or 50,0000 events. The bottom three accumulate the same events except with motion compensation using ground truth data

only 1,000 events, both integrated and motion-compensated event frames are quite sparse with ill-defined edges. Due to the minimal movement over time for this frame, 5,000 events only shows moderate improvement when using motion compensation, with only slight blurring occurring in the simple integrated frame. Using 50,000 events shows dramatic blurring in the simply-integrated frame, while the motion-

Figure 11: Greyscale Image Sample from Boxes dataset: pseudo-randomly placed highly textured boxes of variety sizes

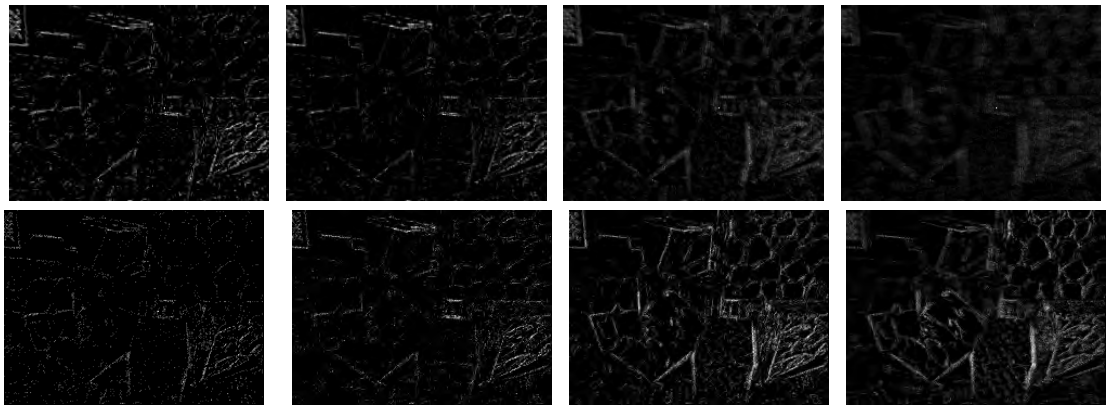

Figure 12: Boxes dataset event-frames motion-compensated with ground truth. The top four images are created with only simple accumulation with 5,000, 20,000, 50,0000, or 100,000 events, from left to right. The bottom four accumulate the same events except with motion compensation using ground truth data

compensated frame provides very distinct outlines of every shape.

The RPG's Boxes dataset [3], with a sample greyscale images shown in Figure 11, is much more varied in texture and depth than the Shapes dataset. This variety causes a much higher rate of events from an event-based camera compared to the Shapes dataset. Figure 12 shows that much many more events are necessary to cause the blurring issues in the simply-integrated event frames and that the motion-compensated frames ensure well-defined edges for all features.

The varied depths of all the features in the camera's field of view in the Boxes dataset did prevent the entire motion-compensated event frame from having perfectly defined edges. The difference is most noticeable when comparing the texture on the top right box in the foreground to the checkerboard in the background on the top left. The quality difference occurs because the current algorithm only uses a constant average depth to initialize three-dimensional positions of tracked and candidate features in each pose.

The Camp Atterbury dataset is from a 14-foot-wingspan fixed-wing UAV flying at around 20 m/s at an altitude of about 250 m over an airfield surrounded by a handful of buildings and fields with a few roads. The sparseness of the scene resulted in a rate of events comparable to the Shapes dataset. The initial location for features in the Camp Atterbury dataset used the altitude measurement to estimate depth, avoiding the need to initialize feature positions with a manually estimated depth. A motion-compensated image does correct some of the blurring seen in the plain integrated image, Figure 14, but still fails to achieve the clarity obtained with a greyscale image in Figure 13.

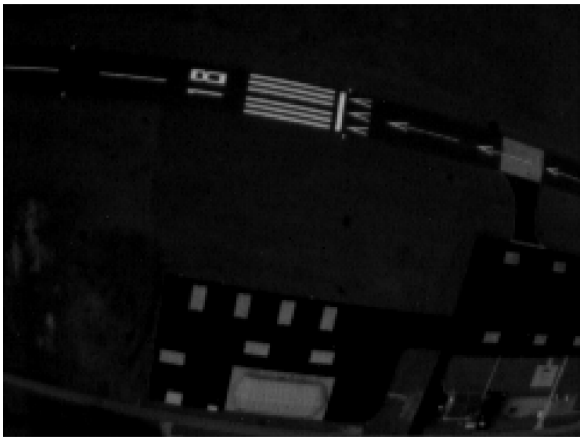

Figure 13: Greyscale Image Sample from Camp Atterbury dataset: downward facing camera aboard a 14-foot wingspan fixed-wing UAV flying at 250 meters above the ground in this image

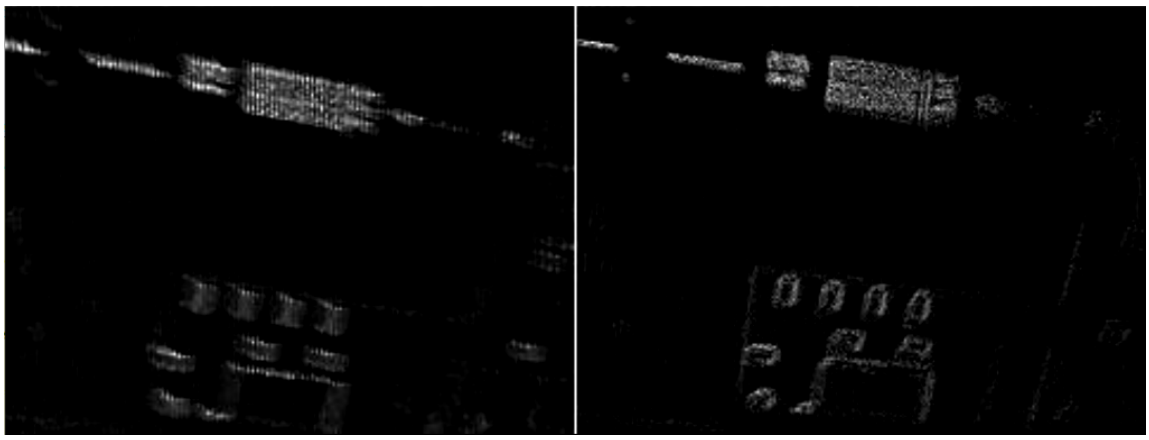

Figure 14: A Camp Atterbury dataset integrated frame, on the left, compared to a motion-compensated event frame, on the right. Both frames begin at around 18 minutes into the final flight in the dataset and use 30,000 events that occur over 481 milliseconds

This poor performance, when compared to the sharp images obtained with the Shapes dataset, is primarily because the ground truth data from the PIKSI Multi-Global Navigation Satellite System (GNSS) only provided high-fidelity position information; the ground truth orientation had much less accuracy, preventing the creation of highly-contrasted motion-compensated images.

4.3 Feature Detection

The KAZE [17] and SURF [15] blob feature detection methods as well as the FAST [19] and Harris [20] corner feature detection methods, as discussed in Section 3.4.3, were tested on motion-compensated event frame samples from the Shapes and Boxes datasets from the RPG Event Camera Dataset [3] and these are shown in Figures 15 to 22.

The SURF methods, shown in Figures 15 and 16, detected the fewest features, with only around 10-20 features detected in each frame in the Shapes dataset and 100-200 features detected in each frame in the Boxes dataset. More important than the number of features detected on each image are the numbers of features that can

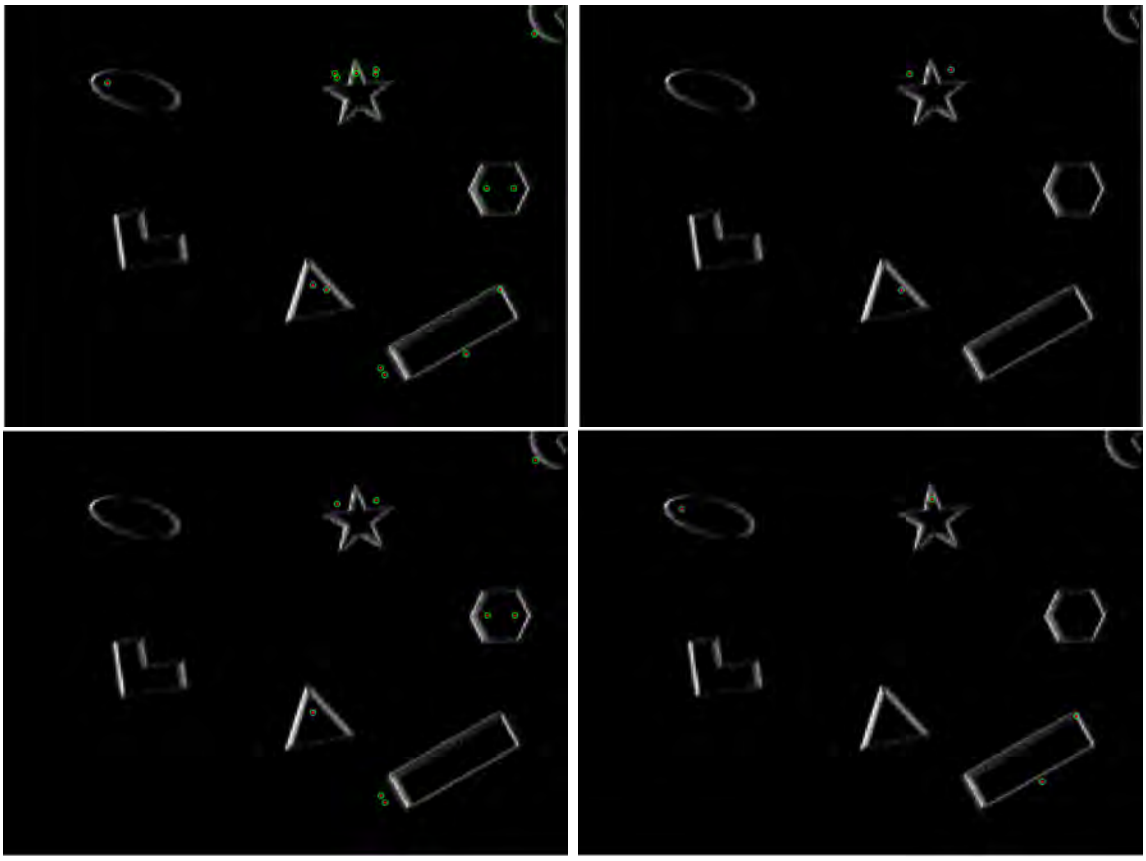

Figure 15: Shapes dataset SURF feature detection using 30,000 events starting at 0.195 seconds into the dataset. Top left: 15 detected SURF features. Top right: 3 new features matched to tracked features. Bottom left: 8 new features matched to candidate features. Bottom right: 4 new candidate features.

be consistently matched and tracked from image to image. For the SURF Features, around half of the newly detected features matched to recently detected features, i.e. candidate features, but only around 5-10% matched to consistent features, i.e. tracked features, which is insufficient to generate quality feature tracks required for reliable visual odometry (VO).

The Harris feature detection method, with results shown in Figures 17 and 18, was able to detect around twice as many features than the SURF detection method, but Harris feature matching had even worse results than SURF feature matching, with only 5-20% of detected features matched to candidates and only 1-2%, if any,

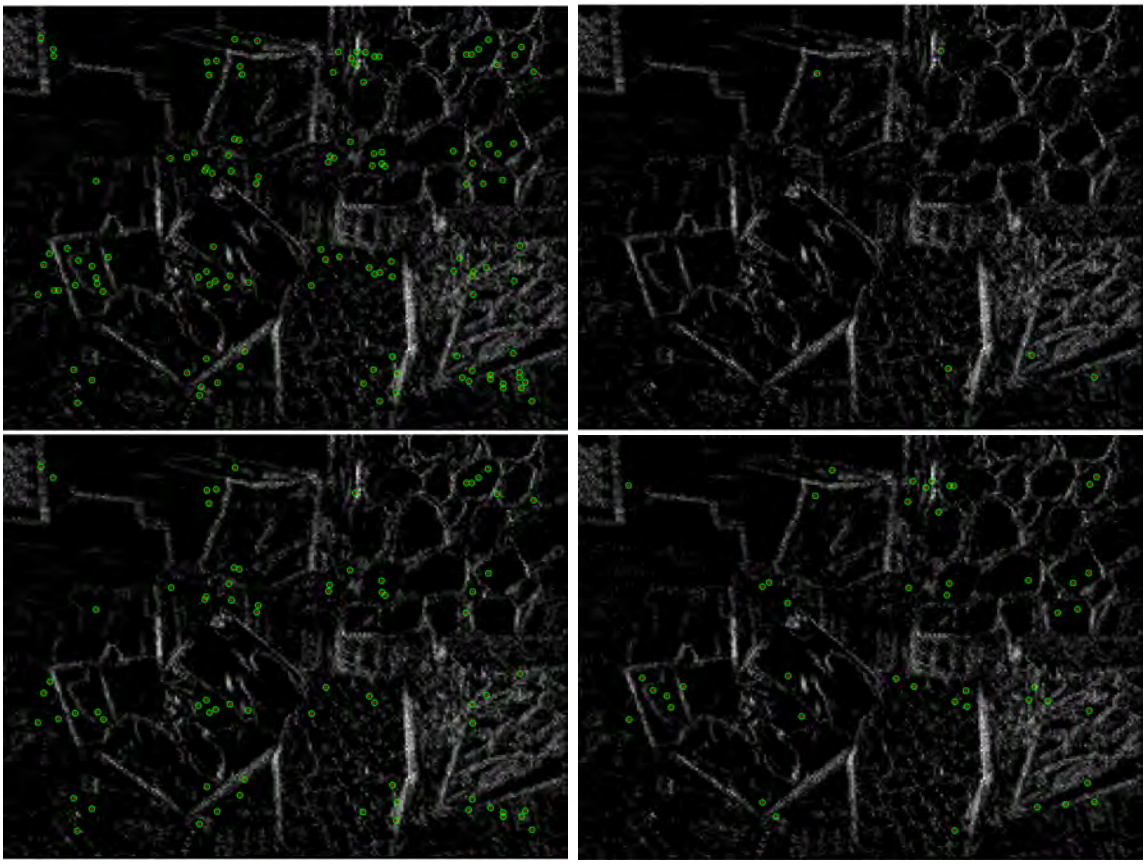

Figure 16: Boxes dataset SURF feature detection using 50,000 events starting at 0.197 seconds into the dataset. Top left: 125 detected SURF features. Top right: 5 new features matched to tracked features. Bottom left: 73 new features matched to candidate features. Bottom right: 47 new candidate features.

matched to tracked features.

FAST corner feature detection was the method used by Zhu's [4] and Rebecq's EVIO implementations. The results from testing with FAST corner feature detection, shown in Figures 19 and 20, illustrate how this method was able to detect many more features, with around 100 features detected in the Shapes dataset frames and around 500-600 in the Boxes dataset frames. However, the matching rate was still only around 20-30% for candidate features and 1-2% for tracked features. Low rates of matching do not generate long feature tracks, which are essential for quality measurement updates to the MSCKF.

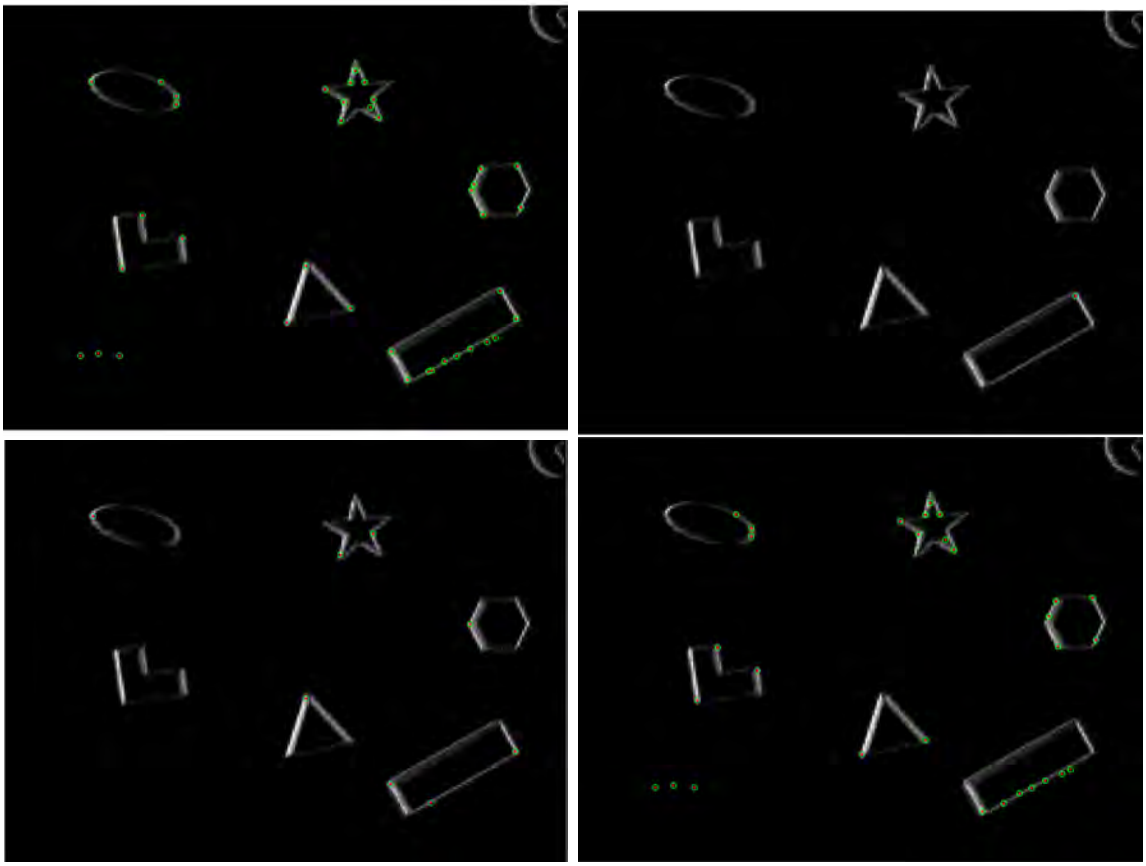

Figure 17: Shapes dataset Harris feature detection using 30,000 events starting at 0.195 seconds into the dataset. Top left: 39 detected Harris features. Top right: 1 new feature matched to tracked features. Bottom left: 8 new features matched to candidate features. Bottom right: 30 new candidate features.

Lastly, the KAZE feature detection, with results shown in Figures 21 and 22, was employed with much better performance, detecting slightly more features than the FAST method, with around 100-200 detected on the Shapes dataset frames and around 600-800 detected on the Boxes dataset frames, but with incredibly better performance on matching over time, with 30-40% matched to candidate features and 20-30% matched to tracked features.

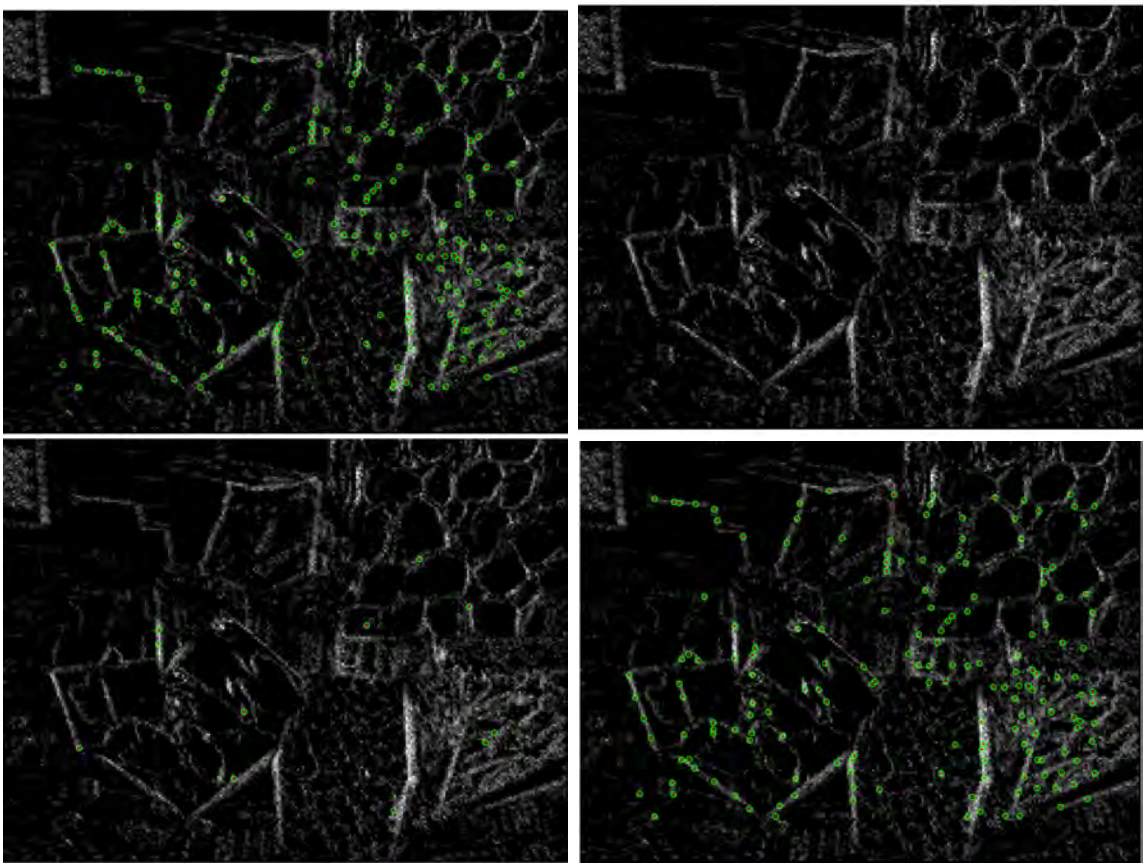

Figure 18: Boxes dataset Harris feature detection using 50,000 events starting at 0.246 seconds into the dataset (no tracked features were matched earlier). Top left: 187 detected Harris features. Top right: 1 new feature matched to tracked features. Bottom left: 11 new features matched to candidate features. Bottom right: 175 new candidate features.

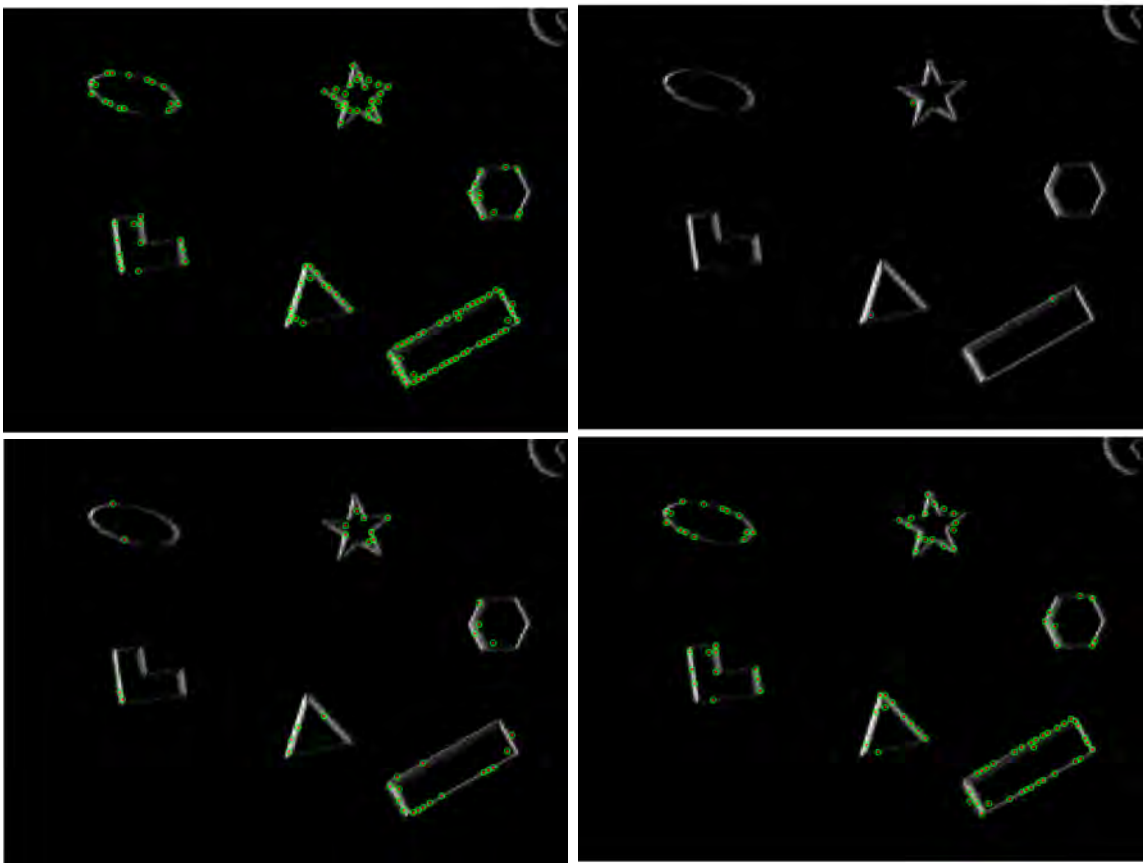

Figure 19: Shapes dataset FAST feature detection using 30,000 events starting at 0.245 seconds into the dataset (no tracked features were matched earlier). Top left: 132 detected FAST features. Top right: 3 new features matched to tracked features. Bottom left: 36 new features matched to candidate features. Bottom right: 93 new candidate features.

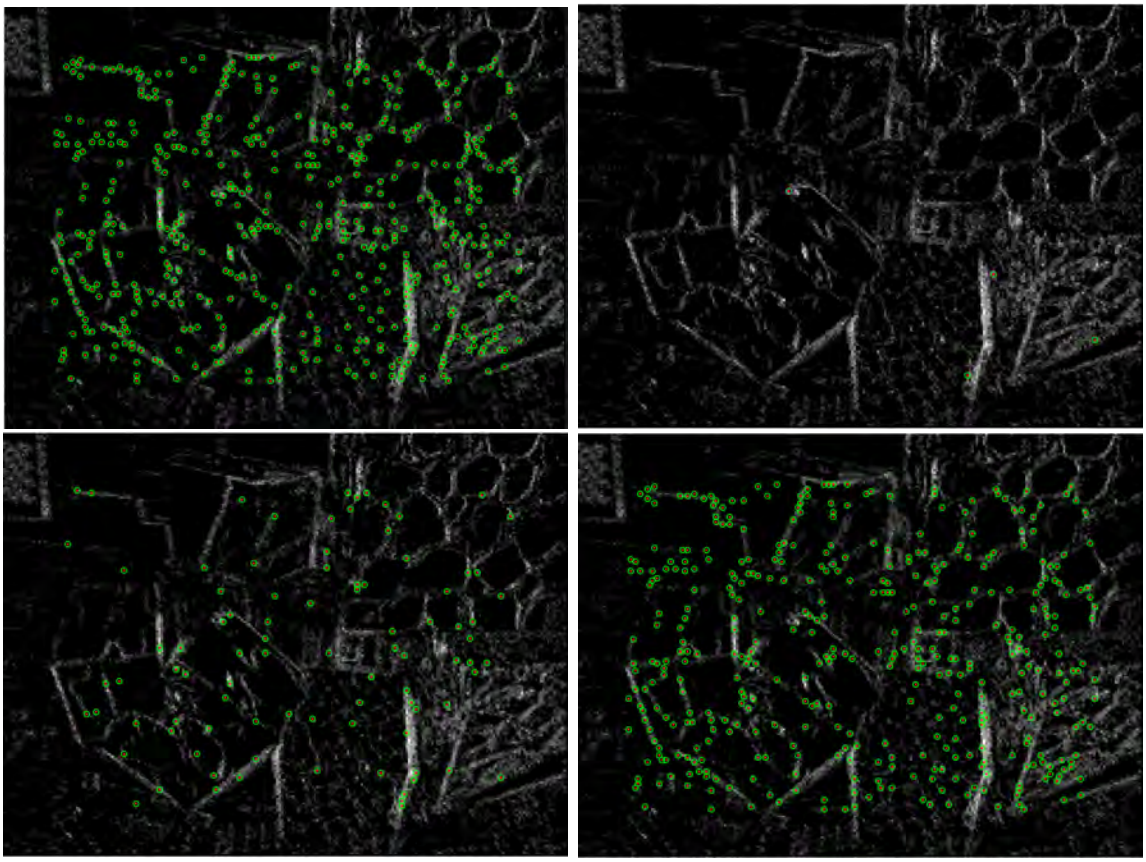

Figure 20: Boxes dataset FAST feature detection using 50,000 events starting at 0.296 seconds into the dataset (no tracked features were matched earlier). Top left: 514 detected FAST features. Top right: 4 new features matched to tracked features. Bottom left: 91 new features matched to candidate features. Bottom right: 419 new candidate features.

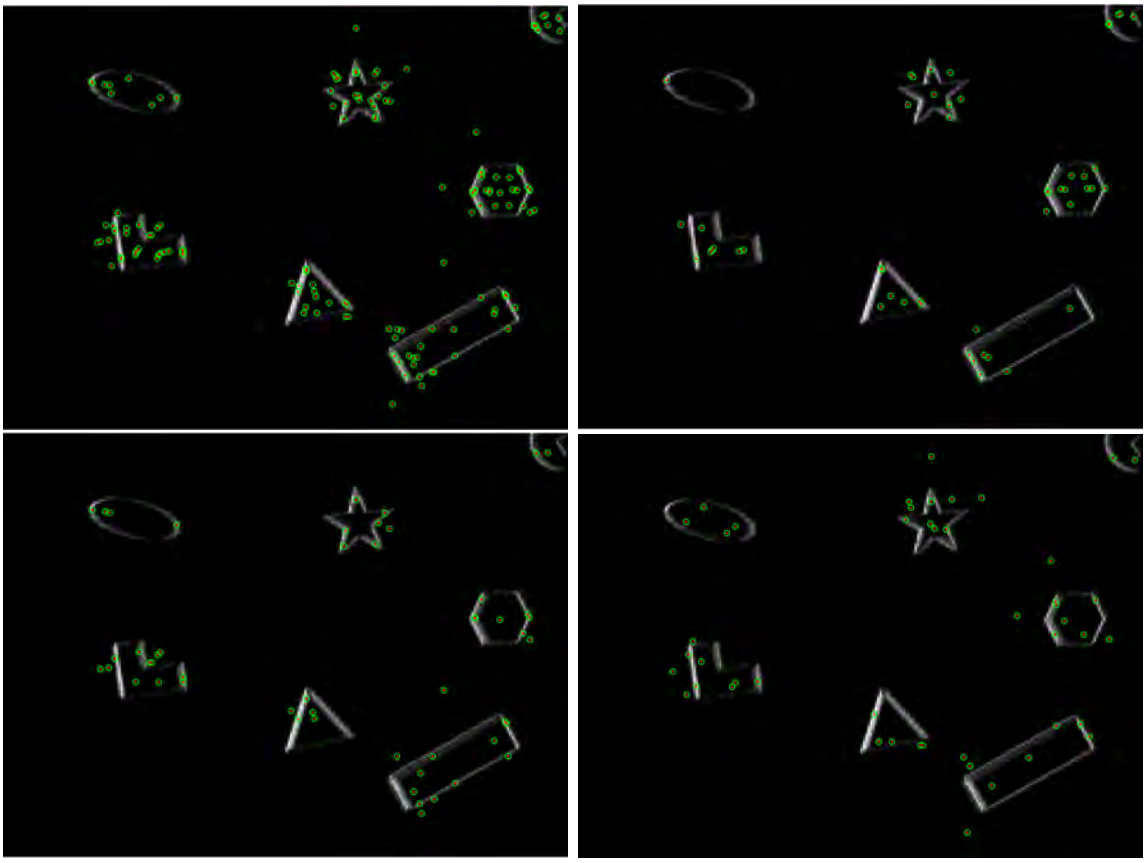

Figure 21: Shapes dataset KAZE feature detection using 30,000 events starting at 0.195 seconds into the dataset. Top left: 149 detected KAZE features. Top right: 50 new features matched to tracked features. Bottom left: 52 new features matched to candidate features. Bottom right: 47 new candidate features.

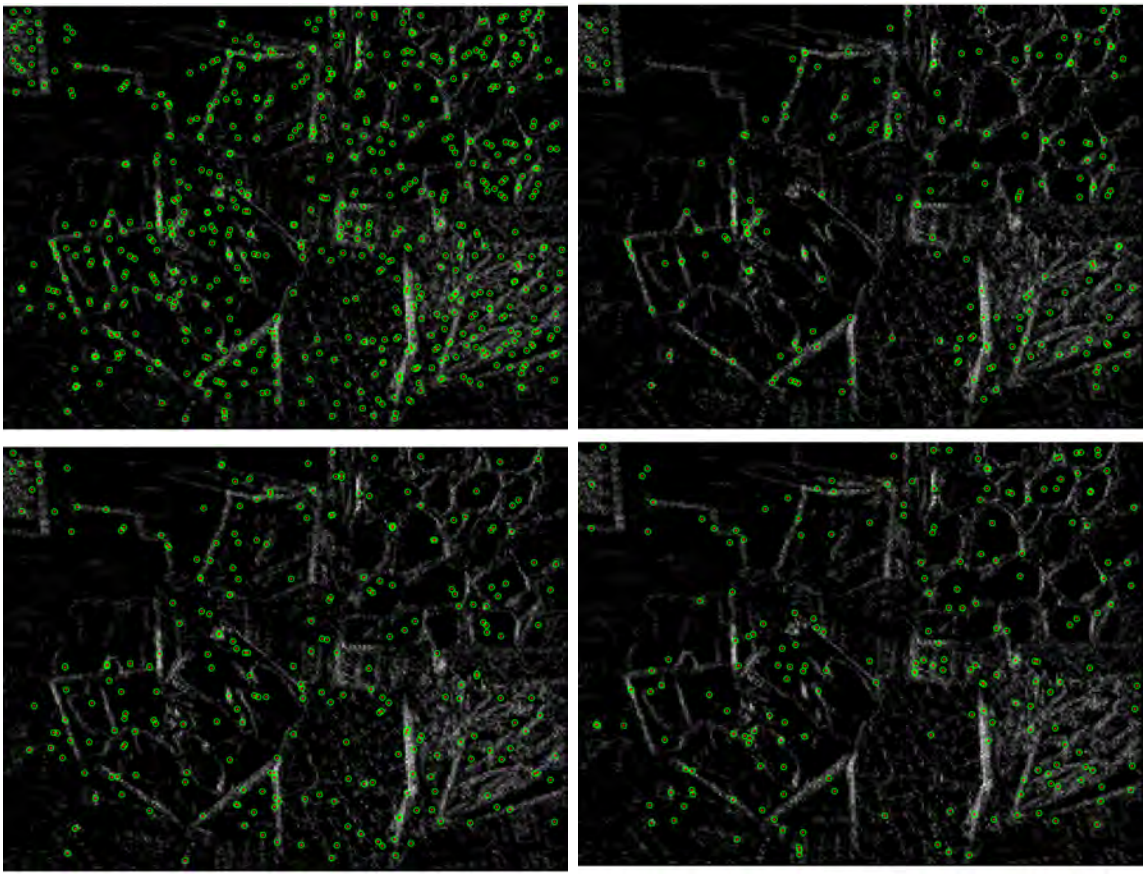

Figure 22: Shapes dataset KAZE feature detection using 50,000 events starting at 0.197 seconds into the dataset. Top left: 651 detected KAZE features. Top right: 194 new features matched to tracked features. Bottom left: 254 new features matched to candidate features. Bottom right: 203 new candidate features.

4.4 Feature Tracking

To troubleshoot the feature tracking matching and bookkeeping system discussed in Sections 3.4.3 and 3.4.7, artificial features were generated and fed to the MSCKF. The results are shown in Figure 23. This illustrates how a third of poses are discarded

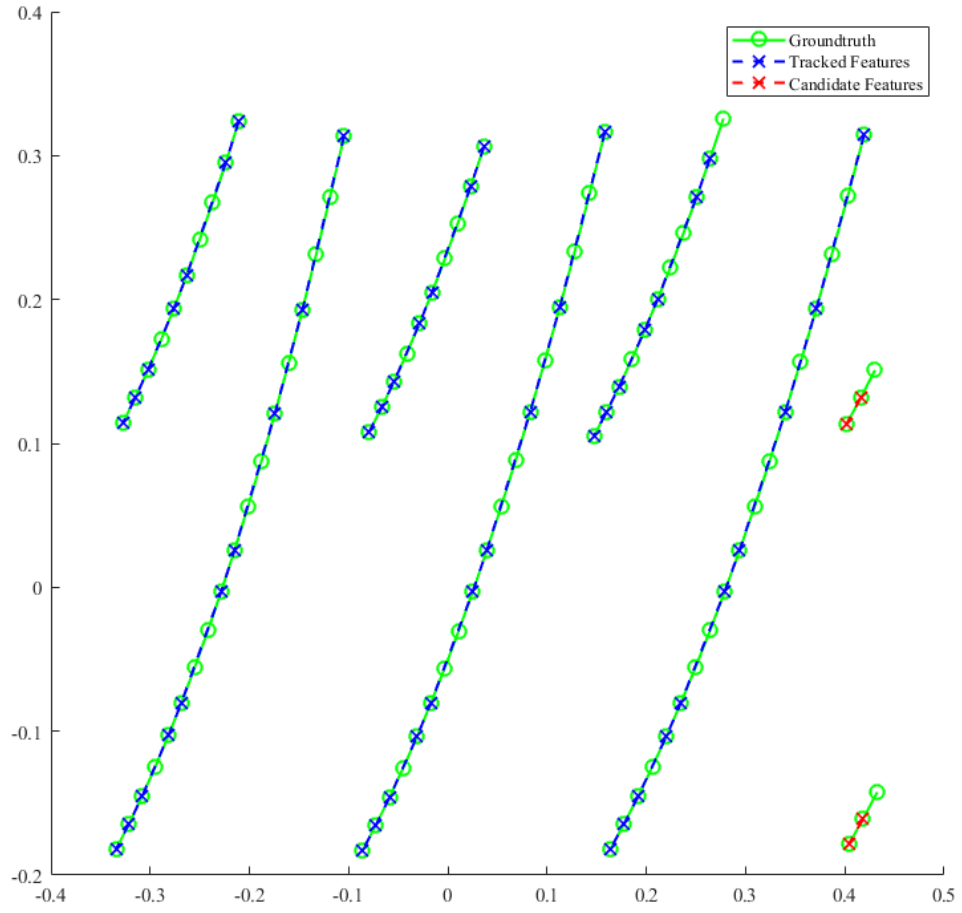

Figure 23: Feature tracks from simulated feature detection: simulated feature detection locations with simplified descriptors were fed to the MSCKF for each pose in a simulated trajectory. The ground truth detections not included in tracked features (i.e. the circles without pink Xs between circles with pink Xs in a single feature track) is due to removal of a third of poses when the maximum number of retained poses is reached.

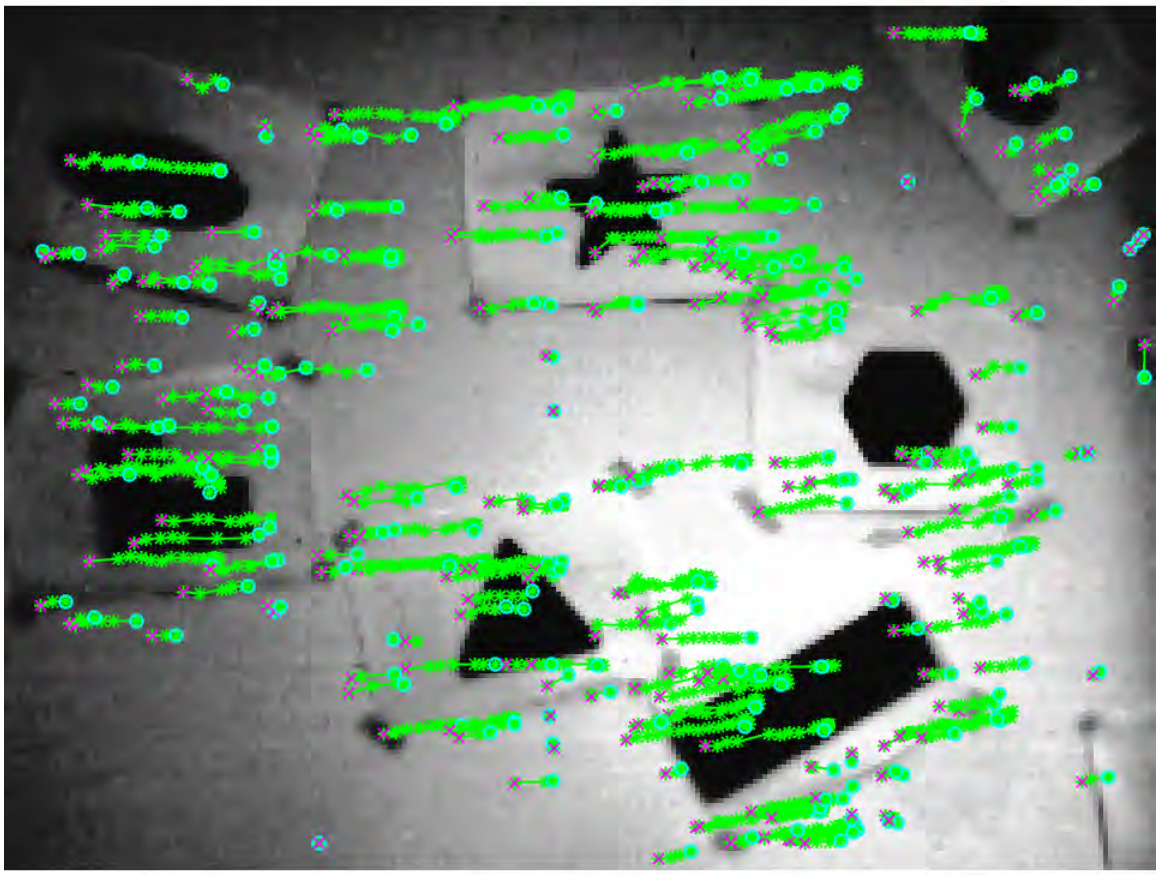

Figure 24: Feature tracks for greyscale images from Shapes dataset: the pink x is the first detection in a given track, with asterisks for each subsequent detection until the circle at the final detection. The image displayed is the last in the sequence and many of the features marked with circles were found on this image. The several dozen feature tracks indicate movement in the positive-X direction, though with some outliers.

when a set limit is reached, as well as how candidate features are properly initialized, separately accounted for, and then transitioned into tracked features to be used in state updates.

The MSCKF feature tracking methodology was then tested using greyscale images as well as on motion-compensated event frames from Shapes, Boxes, and Camp Atterbury datasets. Both greyscale images and motion-compensated frames features were detected using the KAZE feature detector, as that appeared to have the greatest

Figure 25: Feature tracks for greyscale images from Boxes dataset: the pink x is the first detection in a given track, with asterisks for each subsequent detection until the circle at the final detection. The image displayed is the last in the sequence and many of the features marked with circles were found on this image. The several dozen feature tracks indicate some negative yaw rotation with some possible movement in the negative-X direction in the camera's frame of reference, though with some outliers.

number of matched features over time in Section 4.3. Most of the feature tracks generated from the greyscale images, as shown in Figures 24 to 26, as well as the for the feature tracks generated from motion-compensated event frames, as shown in Figures 27 to 29, are consistent in indicating the same general movement of the scene.

The Shapes dataset with motion-compensated event frames in Figure 27 showed a significant clustering of feature tracks around the black shapes, which limits the amount of samples in significant portions of the scene. The Boxes dataset with

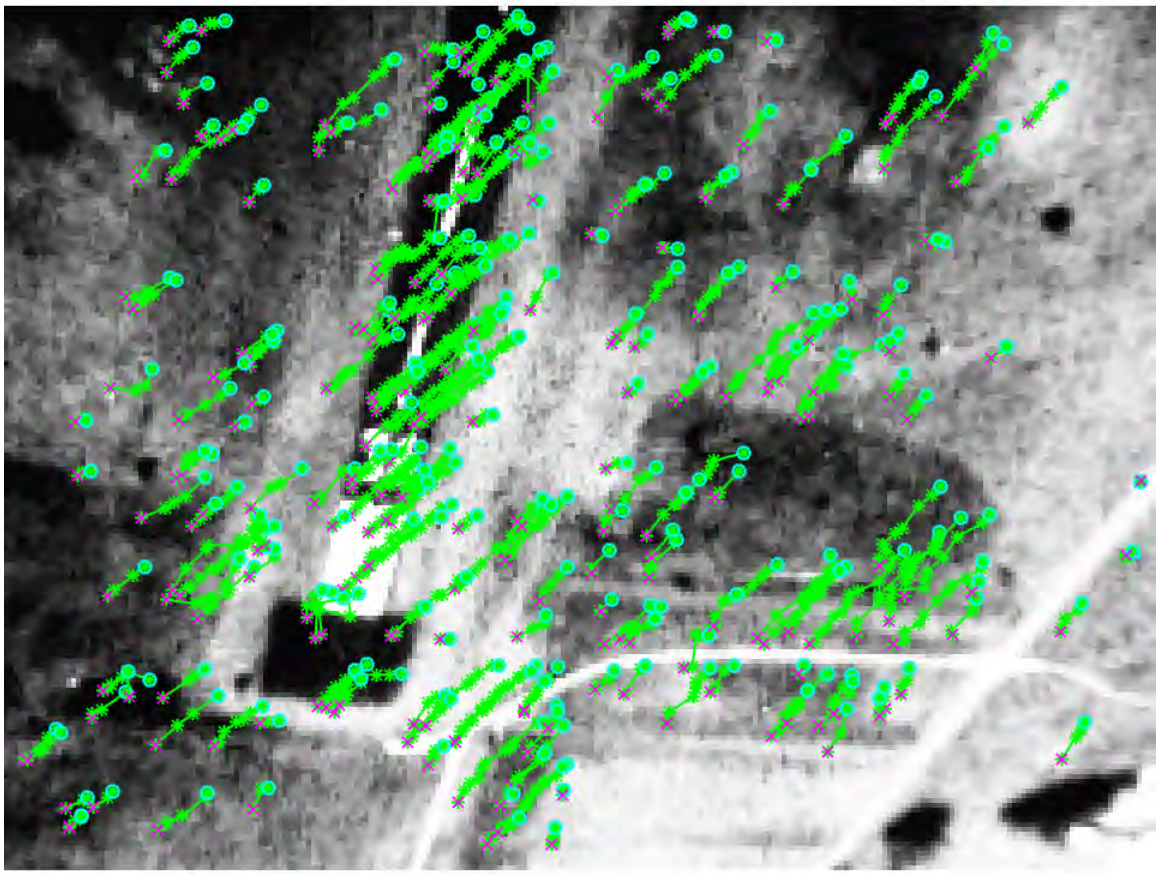

Figure 26: Feature tracks for greyscale images from Camp Atterbury dataset: The pink x is the first detection in a given track, with asterisks for each subsequent detection until the circle at the final detection. The image displayed is the last in the sequence and many of the features marked with circles were found on this image. The several dozen feature tracks indicate some positive roll rotation and/or some movement primarily in the positive-Y direction in the camera's frame of reference, though with some outliers.

motion-compensated event frames in Figure 28 also has features clustered more than with the greyscale images, though not as severely as with the Shapes dataset event frames. The Boxes dataset with event frames also tended to lose track of features sooner, i.e. shorter feature tracks, than with greyscale images in Figure 25, which would result in a narrower sample available for the MSCKF update. The Camp Atterbury dataset with event frames in Figure 29 had the severest performance

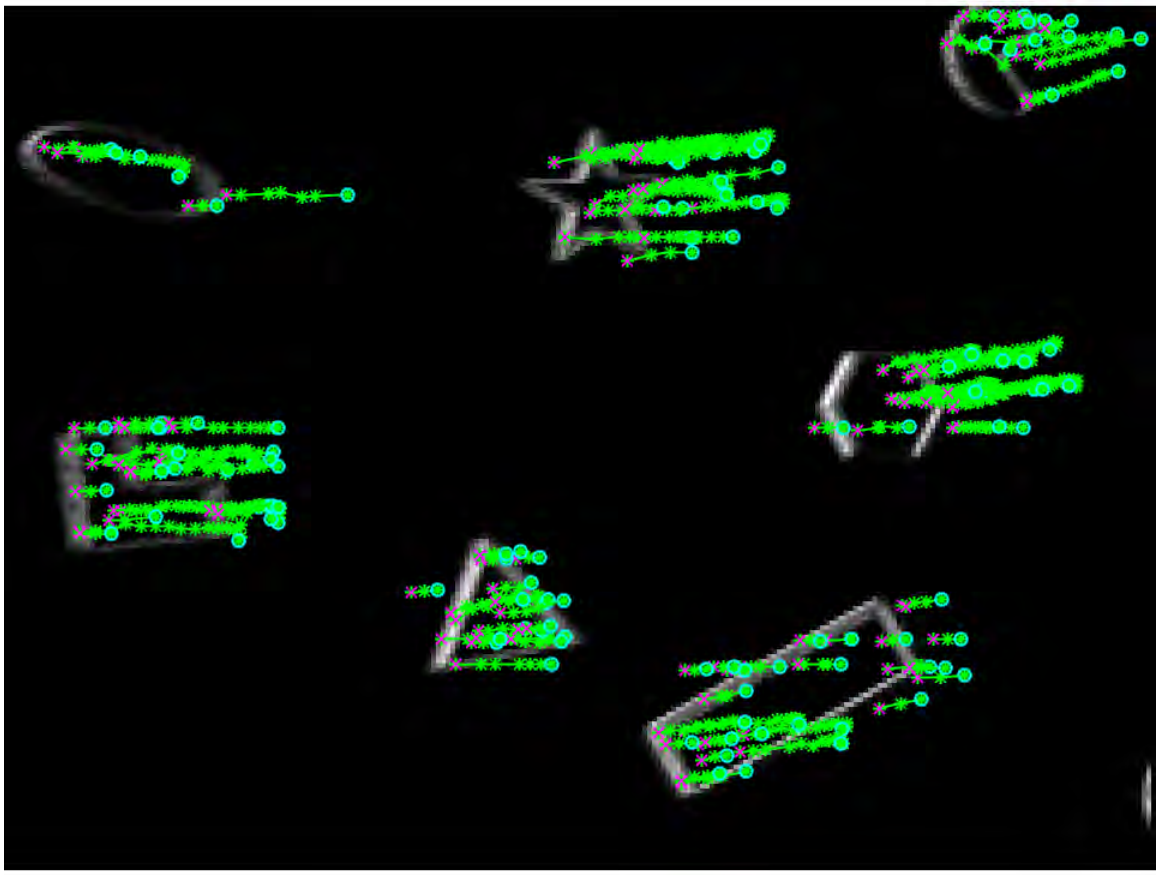

Figure 27: Feature tracks for event frames from Shapes dataset: the pink x is the first detection in a given track, with asterisks for each subsequent detection until the circle at the final detection. The frame displayed is the last in the sequence and many of the features marked with circles were found on this frame. The features are more clustered than with a greyscale image. The couple dozen feature tracks indicate movement in the positive-X direction, though with some outliers.

reduction compared to the greyscale images in Figure 26. One reason for this is that the vast majority of the scene did not cause enough events to create detectable features in the event frames, as shown by the half a dozen feature tracks. Furthermore, it was difficult to sufficiently estimate accurate accelerometer and gyroscope bias estimates, and the depth estimate only came from the altitude measurements without accounting for the orientation of the vehicle. These constraints hindered the ability to create crisp motion-compensated event frames, since the inertial measurement unit (IMU) quickly

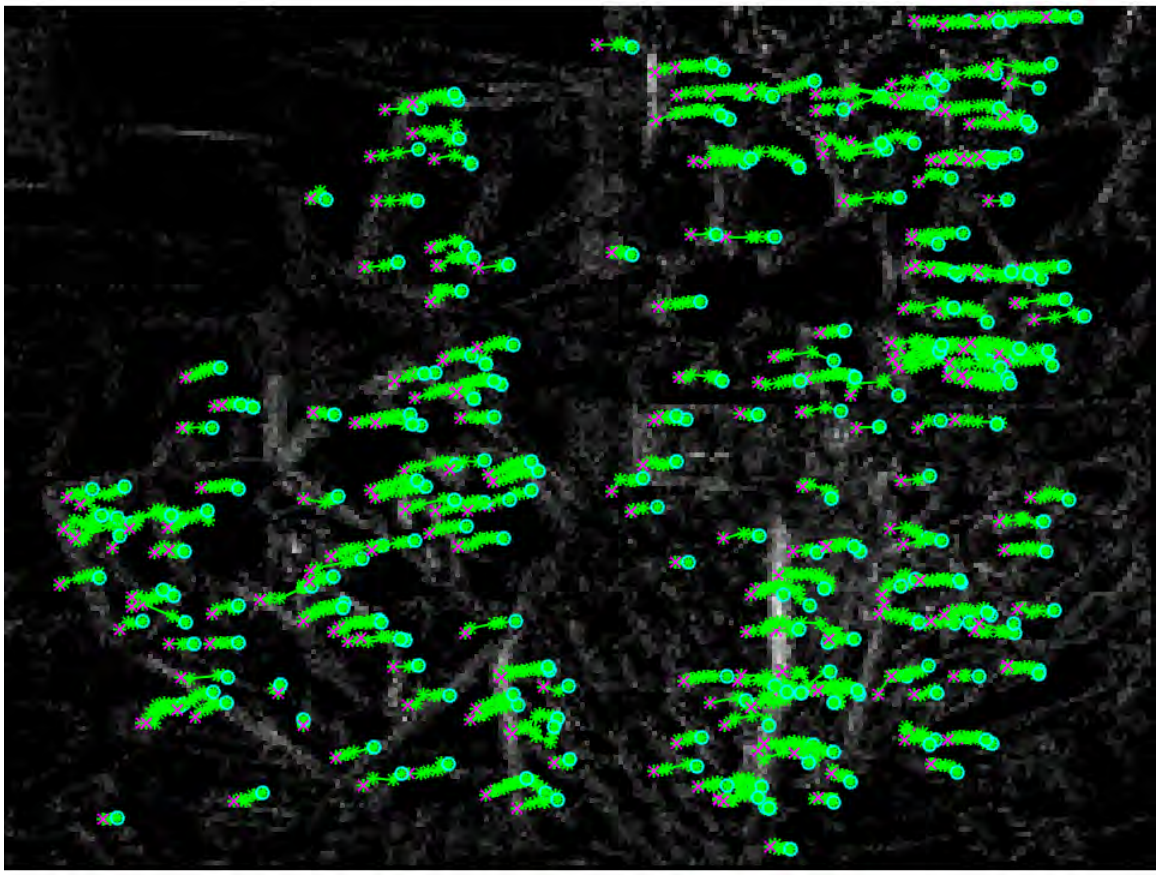

Figure 28: Feature tracks for event frames from Boxes dataset: the pink x is the first detection in a given track, with asterisks for each subsequent detection until the circle at the final detection. The frame displayed is the last in the sequence and many of the features marked with circles were found on this frame. The features are more clustered than with a greyscale image. The couple dozen feature tracks indicate movement in the positive-X direction, though with some outliers.

drifted away from truth over the time window of the batch of events used to create the event frame.

With both greyscale images or motion-compensated event frames, significant outliers on feature tracks were avoided in the feature matching step by ensuring feature matches were within a threshold distance of the latest detected feature, but some minor outliers still remained that could introduce errors into the VO pipeline. Though not implemented in this work, the remaining outliers could be addressed through 2-

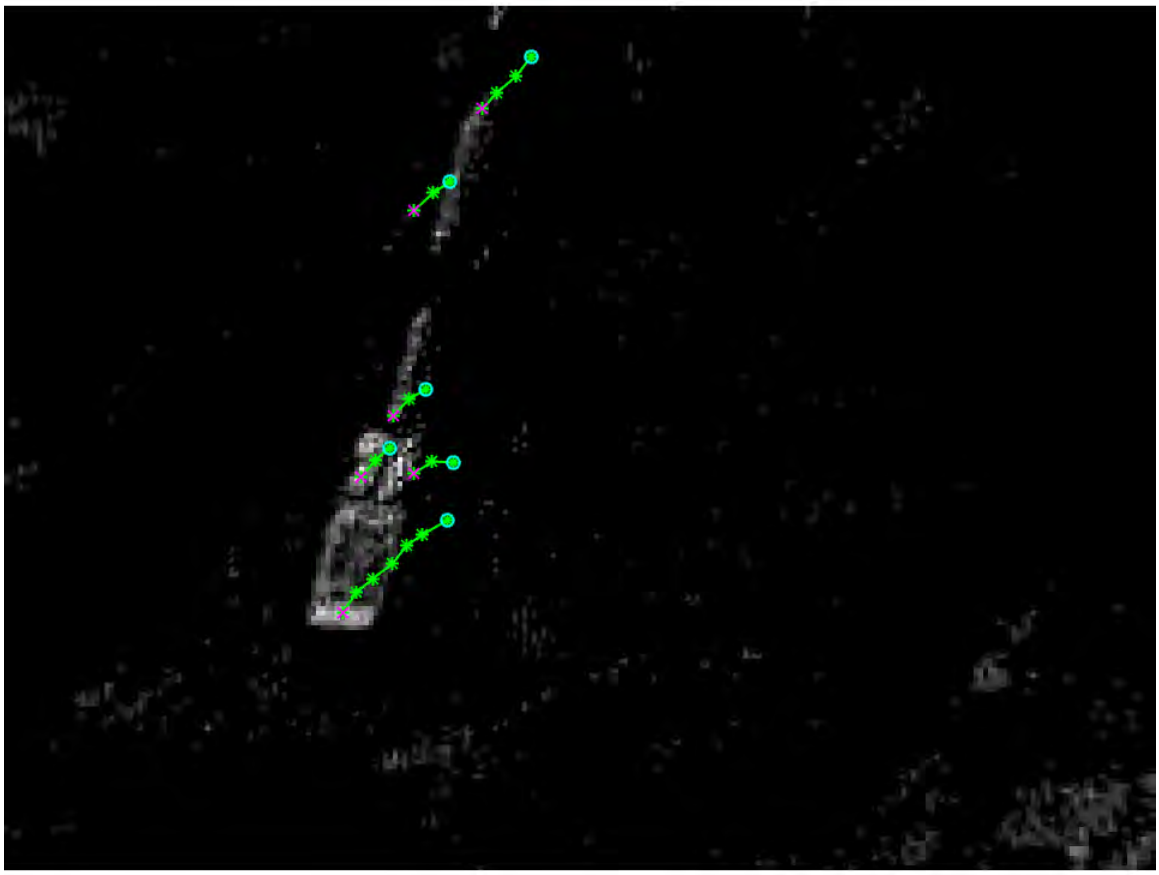

Figure 29: Feature tracks for event frames from Camp Atterbury dataset: the pink x is the first detection in a given track, with asterisks for each subsequent detection until the circle at the final detection. The frame displayed is the last in the sequence and many of the features marked with circles were found on this frame. There are far fewer features detected compared to the greyscale images. The half dozen feature tracks indicate some positive roll rotation and/or some movement primarily in the positive-Y direction in the camera's frame of reference. At least one of the 6 tracks appears to be an outlier.

point random sample consensus (RANSAC) [154] and/or Mahalanobis distance checks [155].

4.5 Camera-IMU Calibration

Camera intrinsic calibration and camera-IMU extrinsic calibration for the Camp Atterbury dataset followed the process explained in Section 3.5.2.1 using a checker-

Table 1: Camera Intrinsic Parameters

Description	**Variable**	**Value**
Focal lengths	f_x	198.444
	f_y	198.826
Image Center	c_x	104.829
	c_y	92.838
Radial Distortion	k_1	-0.394
	k_2	0.156
	k_3	0
Tangential Distortion	p_1	-0.125×10^{-3}
	p_2	-1.629×10^{-3}

board pattern. The camera intrinsic calibration was able to generate the parameters shown in Table 1, leading to an intrinsic camera matrix of

$$K = \begin{bmatrix} 198.444 & 0 & 104.829 \\ 0 & 198.826 & 92.838 \\ 0 & 0 & 1.000 \end{bmatrix}. \tag{76}$$

These camera intrinsic parameters were used in several steps of this research. The distortion parameters were used to undistort greyscale images or simply-integrated event frames. The camera matrix in (76) was used to map features' image plane locations in the camera's reference frame. For motion-compensated event frames, the camera matrix and distortion parameters were used to precompute a look-up table for the undistorted homogeneous coordinates for each discrete pixel location, used in Line 5 of Algorithm 1, and the inverse camera matrix was used to re-project the motion-compensated events onto discrete pixels of an event frame, as shown in Line 13 of Algorithm 1.

To generate the camera-IMU extrinsic parameters used in (26), the camera intrinsic parameters were used with a sequence of checkerboard images and the corresponding IMU measurements by Kalibr to generate estimates of the transformation between

the IMU and camera frames of reference. The results from this transformation shows how the camera and the IMU are very nearly aligned. The rotation matrix from the camera reference frame to the IMU reference frame is

$$\mathbf{R}_C^{IMU} = \begin{bmatrix} 1.000 & 5.880 \times 10^{-3} & -7.712 \times 10^{-3} \\ -5.790 \times 10^{-3} & 1.000 & 11.557 \times 10^{-3} \\ 7.779 \times 10^{-3} & -11.512 \times 10^{-3} & 1.000 \end{bmatrix}, \tag{77}$$

which corresponds to Roll-Pitch-Yaw Euler angles of [0.662°,0.442°,0.337°]. The translation, being the position of the origin of the camera frame of reference in the IMU reference frame, from the transformation matrix, in meters, is

$$\mathbf{p}_C^{IMU} = \begin{bmatrix} 4.451 \times 10^{-3} \\ -8.024 \times 10^{-3} \\ -20.438 \times 10^{-3} \end{bmatrix}. \tag{78}$$

4.6 Initial Bias Estimates

Biases were manually estimated for each dataset to minimize the impact of bad initial motion estimates on the creation of motion-compensated images. Figure 30 shows that, without applying any biases, the Shapes dataset quickly diverges in roll by nearly 90°; pitch and yaw are also inaccurate, though remaining within the same order of magnitude. Position and velocity for the Shapes dataset, without any bias applied, radically diverge from the truth data by one to two orders of magnitude. Utilizing a gyroscope bias of $[-0.05; -0.010; -0.001]^T rad/s$ and an accelerometer bias of $[0.05; 0.245; -0.225]^T\ m/s^2$, provides the results shown in Figure 31. The error in orientation was drastically reduced over the entirety of the Shapes dataset, remaining within 1° of the ground truth. The position and velocity estimates were also improved, compared to results without bias correction applied, with velocity errors remaining

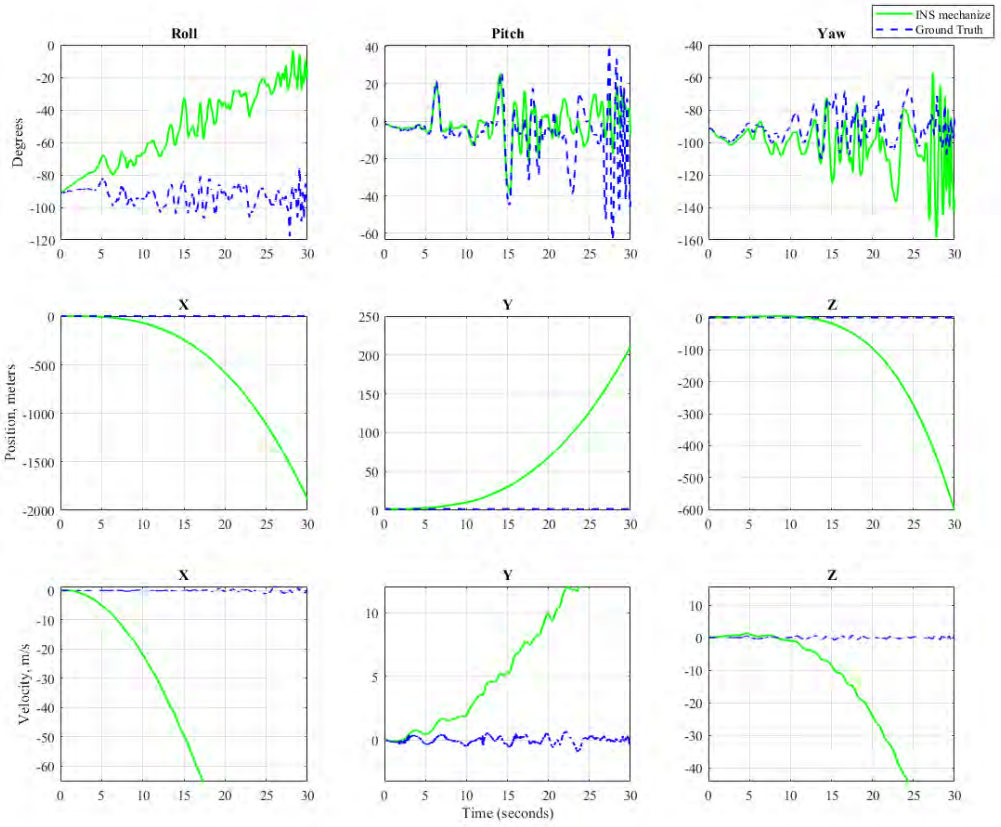

Figure 30: Shapes dataset, INS propagation without bias correction: Pitch and yaw start out relatively close, but roll quickly diverges from truth; all velocities diverge from truth within the first couple seconds, resulting in completely unreliable position estimates.

within the same order of magnitude as the changes in velocity, and position errors remaining under tens of centimeters after several seconds.

The Boxes dataset had similar performance, with Figure 32 showing that, without applying any biases, roll diverged the most out of all orientation states. Position and velocity for the Boxes dataset, without any bias applied, also radically diverge from the truth data by one to two orders of magnitude. Utilizing a gyroscope bias of $[-0.05; -0.010; -0.001]^T\ rad/s$ and an accelerometer bias of $[-0.04; 0.14; -0.1425]^T\ m/s^2$, provides the results shown in Figure 33. The error in orientation was drastically reduced over the entirety of the Boxes dataset, remaining within 1° of the ground

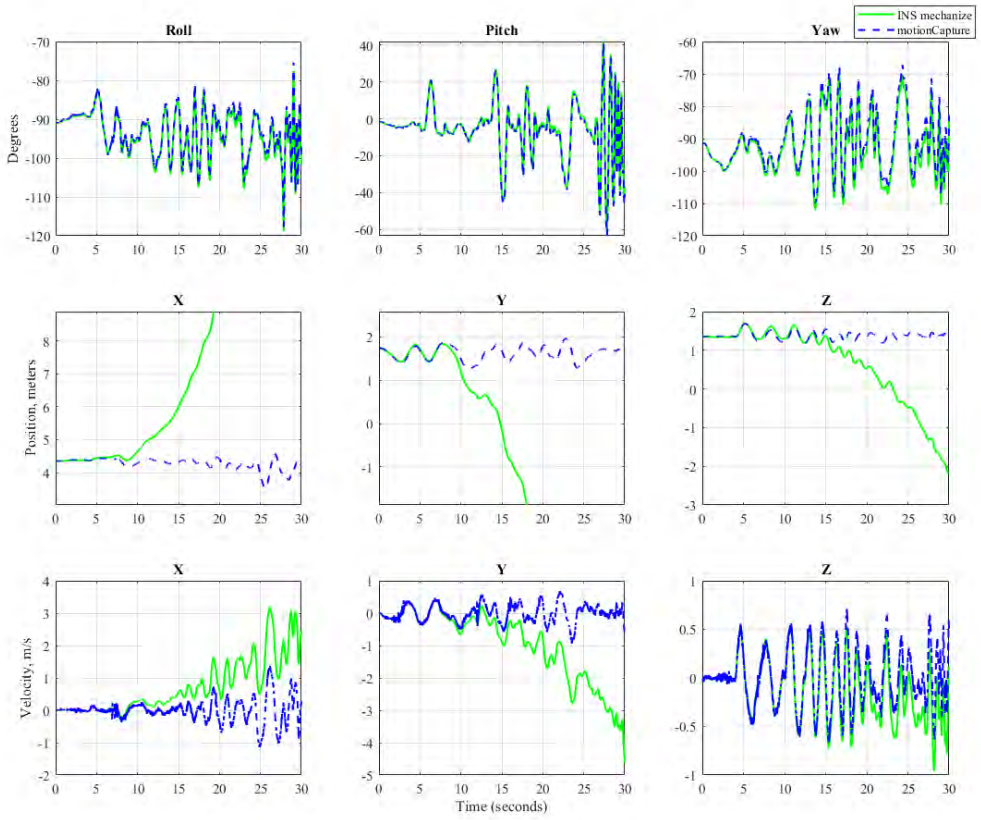

Figure 31: Shapes dataset, INS propagation with bias correction applied: Applying gyroscope bias corrections of $-0.050\,\mathrm{rad/s}$, $-0.010\,\mathrm{rad/s}$ and $-0.001\,\mathrm{rad/s}$ to X, Y and Z axis measurements, respectively, enables INS mechanization to track the orientation even during rapid movement. Applying accelerometer bias corrections of $0.050\,\mathrm{m/s^2}$, $0.245\,\mathrm{m/s^2}$ and $-0.225\,\mathrm{m/s^2}$ enables improved tracking of vehicle position for 5 to 10 seconds, after which it again starts to diverge.

truth. The position and velocity estimates were also improved, compared to results without bias applied, with velocity errors remaining within the same order of magnitude as the changes in velocity, and position errors keeping within tens of centimeters after several seconds.

It proved much more difficult to find accurate biases for the Camp Atterbury dataset. Position information was the only ground truth data with the level of accuracy on par with the ground truth from the other datasets. The roll and pitch

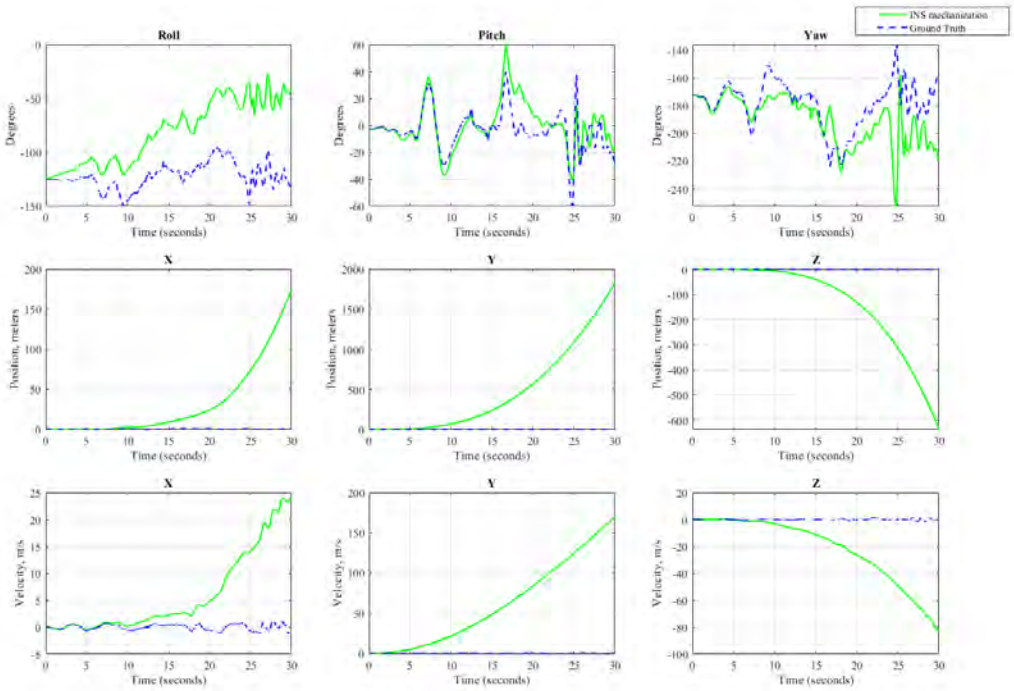

Figure 32: Boxes dataset, INS propagation without bias correction applied: Pitch and yaw start out relatively close, but roll quickly diverges from truth; the X and Z velocities appear to track the rise and fall of truth but the Y velocity quickly diverges from truth within the first couple seconds, resulting in unreliable position estimates.

were taken from the PIXHAWK flight controller but the yaw measurement from the PIXHAWK was unreliable, so the heading was calculated from the ground course position information and used in place of the yaw measurement. The velocity was calculated as the gradient of position. The results from propagating the IMU measurements through INS mechanization without any bias corrections applied are shown in Figure 34. Estimating the biases manually, generally through hand tuning, for the Camp Atterbury dataset proved much more difficult than with Shapes or Boxes dataset, as shown in INS mechanization generated the results in Figure 35. The error during the first 5 seconds was generally reduced for all metrics, but the error diverged significantly after 15 to 20 seconds, which is about as good as can be expected for a cheap, commercial-grade Micro-Electro-Mechanical Systems (MEMS) IMU sensor.

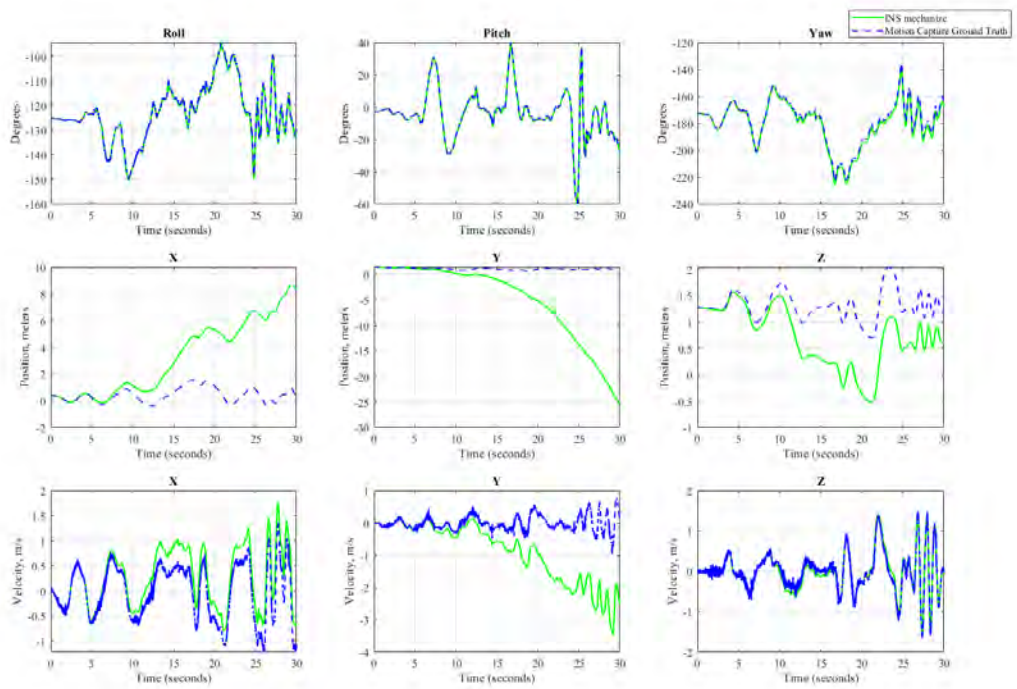

Figure 33: Boxes dataset, INS propagation with bias correction applied: Applying gyroscope bias corrections of $-0.050\,\mathrm{rad/s}$, $-0.010\,\mathrm{rad/s}$ and $-0.001\,\mathrm{rad/s}$ to X, Y and Z axis measurements, respectively, enables INS mechanization to track the orientation even during rapid movement. Applying accelerometer bias corrections of $-0.040\,\mathrm{m/s^2}$, $0.140\,\mathrm{m/s^2}$ and $-0.143\,\mathrm{m/s^2}$ enables improved tracking of vehicle position for 5 to 10 seconds, after which it again starts to diverge. The Y axis had the worst performance, though still an improvement of only diverging by about 30 meters over 30 seconds versus over 1500 meters without any bias correction applied.

Further analysis on the Camp Atterbury dataset not undertaken in this work may find more accurate initial bias estimates that yield better performance.

These estimated biases were used with the MSCKF, but only by propagating the primary states as described in Section 3.4.2, and without incorporating any feature measurements as described in Section 3.4.3, or applying any update steps as described in Section 3.4.6. The results of this application are shown in Figures 36 to 38 as the baseline to evaluate the performance when using feature tracks from images or motion-compensated event frames.

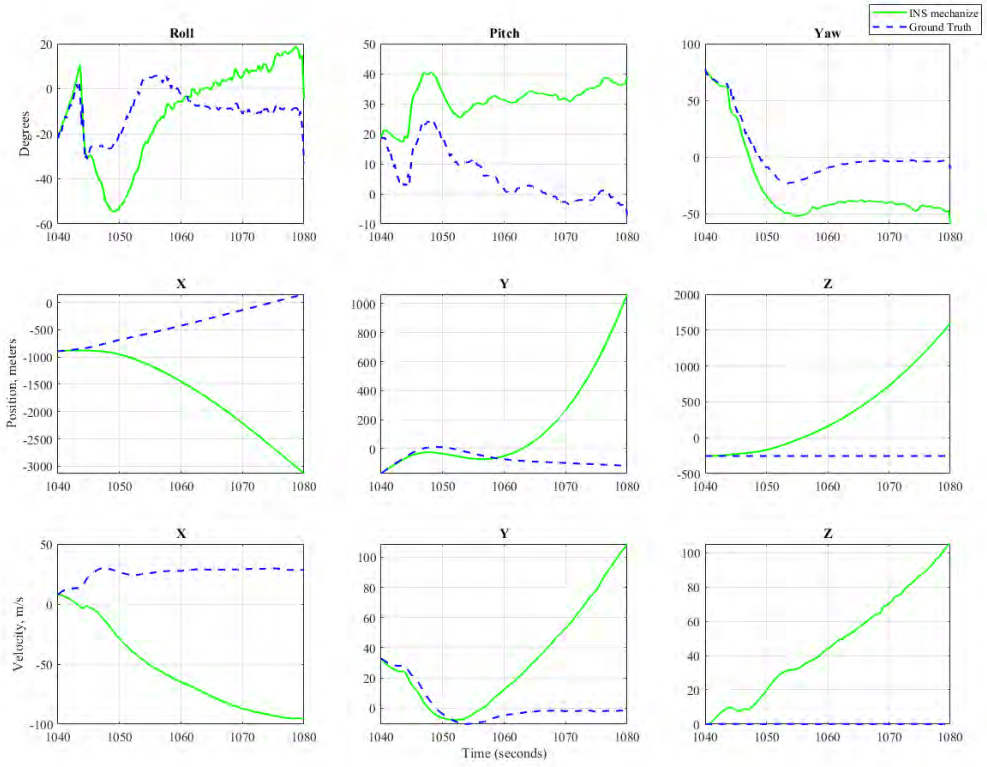

Figure 34: Atterbury dataset, INS propagation without bias correction applied: Roll and yaw start out tracking truth relatively well, but yaw quickly is unable to track truth; the Y velocity appears to track the initial fall of truth but the X and Z velocities immediately diverge from truth, resulting in unreliable position estimates.

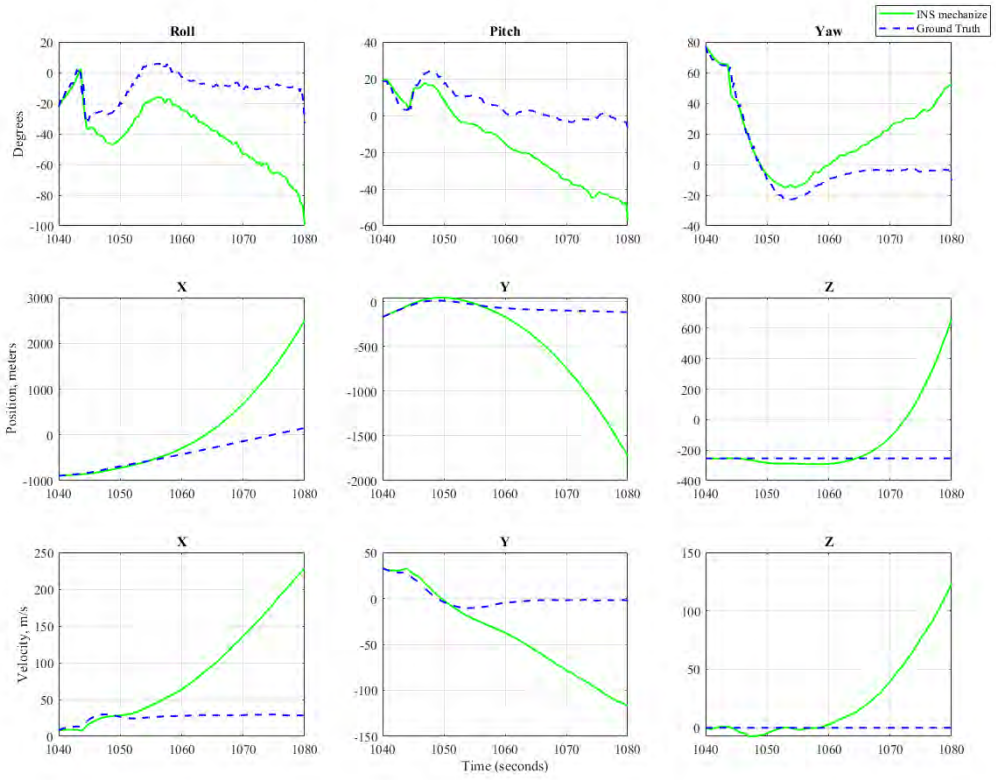

Figure 35: Camp Atterbury dataset, INS propagation with bias correction applied: Applying gyroscope bias corrections of $-0.050\,\mathrm{rad/s}$, $-0.050\,\mathrm{rad/s}$ and $0.005\,\mathrm{rad/s}$ to X, Y and Z axis measurements, respectively, enabled INS mechanization to track the orientation for the first few seconds before diverging. Applying accelerometer bias corrections of $3.000\,\mathrm{m/s^2}$, $-1.000\,\mathrm{m/s^2}$ and $-1.500\,\mathrm{m/s^2}$ enables improved tracking of vehicle position for 5 to 10 seconds, after which it again starts to diverge.

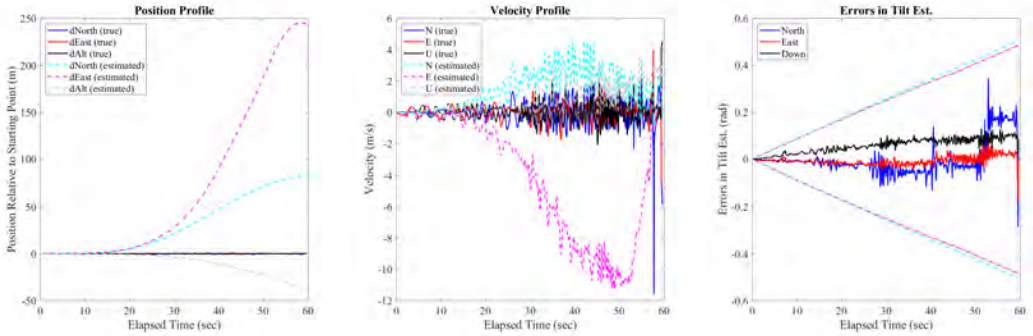

Figure 36: Shapes dataset MSCKF results with only propagation: as shown in Figure 31, the tilt estimate remained within 0.1 to 0.2 radians (5° to 10°) on all axes over the length of the dataset (60 seconds) and the velocity diverges from truth after about 10 seconds. Since this is propagation only, the errors states remain at zero since no update step is applied.

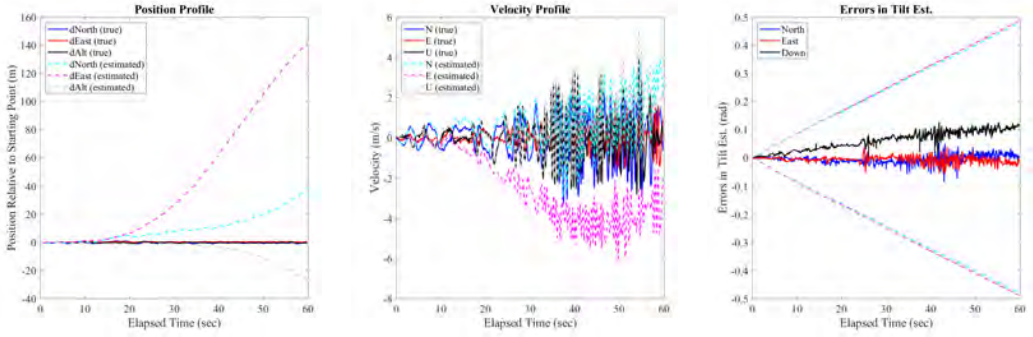

Figure 37: Boxes dataset MSCKF results with only propagation: as shown in Figure 33, the tilt estimate remained within about 0.05 to 0.1 radians (2° to 5°) on all axes over the length of the dataset (60 seconds) and the velocity diverges from truth after about 10 seconds. Since this is propagation only, the errors states remain at zero since no update step is applied.

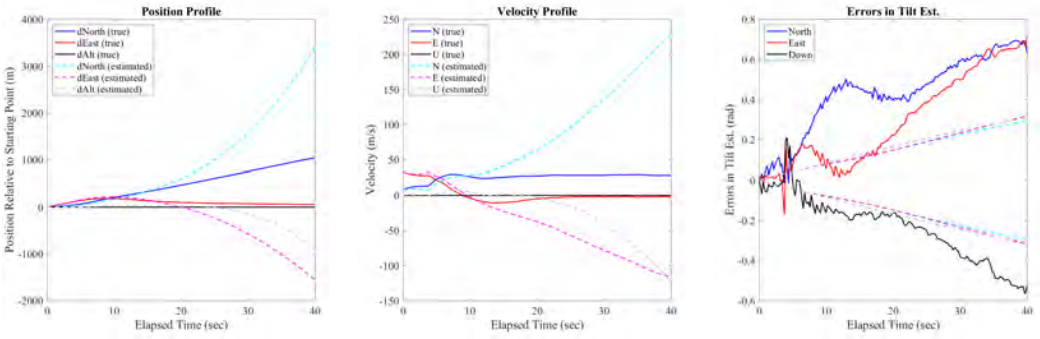

Figure 38: Camp Atterbury dataset MSCKF results with only propagation: as shown in Figure 35, the tilt estimate remained within about 0.1 radians (5°) on all axes for only the first couple seconds and the velocity radically diverges from truth after about 10 to 15 seconds, ending with position estimated several kilometers from truth. Since this is propagation only, the errors states remain at zero since no update step is applied.

4.7 MSCKF with Updates

The complete MSCKF was first tested using the greyscale images. Each image was undistorted using the parameters generated in Section 4.5, then enhanced by equalizing the tonal histogram of the image. Feature detection was then conducted as described in Sections 3.4.3 and 3.4.7, using a KAZE feature detector and retaining the strongest 48 features in each region for a maximum of 768 new features detected for each image. The results from these tests are shown in Figures 39 to 41, which actually show a dramatic decrease in performance after incorporation of the update step. It can also be noted that, in the Shapes and Boxes Datasets, the latter half of the dataset had an increased amount of rapid and sporadic camera movement which caused significant image blurring and less overlap in sequential images, hindering feature detection and matching, and preventing any updates from occurring.

Testing with motion-compensated event frames on the MSCKF is shown in Figures 42 to 44, and demonstrates dramatically worse performance than with greyscale images, which was already worse than simple INS mechanization. Furthermore, once reasonably accurate tracking of the system states is lost, the motion-compensated event frames become completely corrupted and add no value with no matchable features, preventing any chance at recovery.

Other parameter adjustments, described below, were also tested though still with poorer performance than the results shown in this chapter. A sampling of the results from implementing these parameter adjustments are included in Appendix A.

- Vary the value of $\sigma_{\texttt{im}}$ used in the least-squares estimate (LSE) in (47) and for the noise components discussed in Sections 3.4.6.2 and 3.4.6.3. The results shown in this chapter used 1×10^{-3}.

- Vary the distance threshold for feature matching. The results shown in this

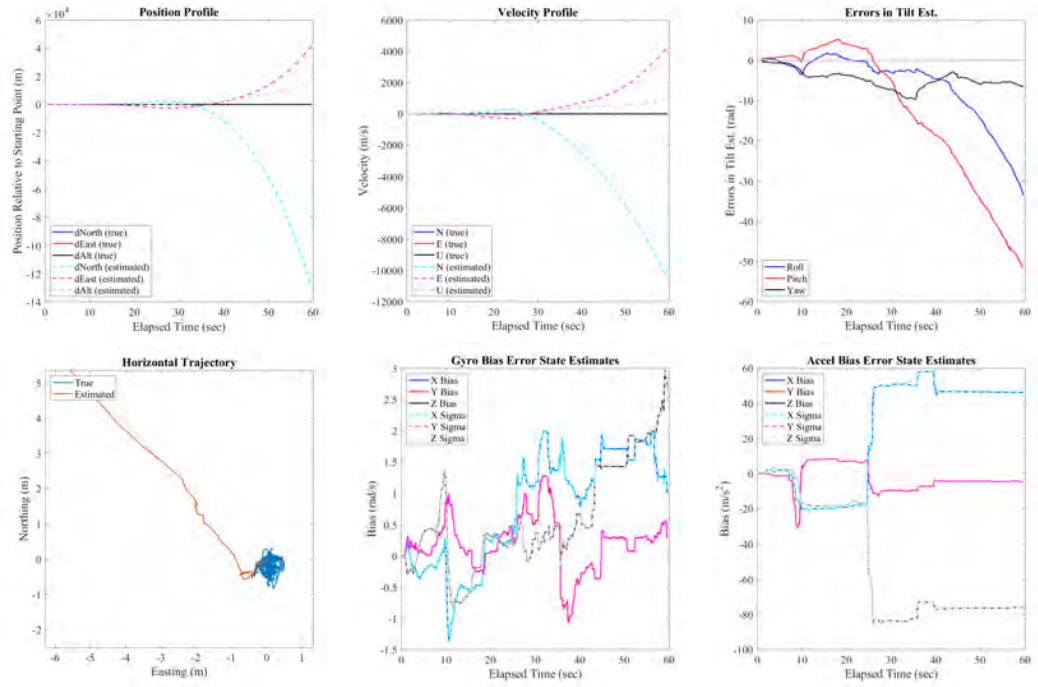

Figure 39: Shapes dataset MSCKF results with greyscale images: the incorporation of the update step does not improve the performance of the MSCKF, but rather makes it dramatically worse compared to the propagation-only results in Figure 36. Further parameter and/or algorithm adjustment is required. Additionally, there is a lack of updates during the latter half of the dataset, as can be easily seen in flat-lining of the accelerometer bias estimates. This is likely due to the increase in rapid and sporadic movements which caused blurring in the images, hindering feature detection and matching.

chapter used a distance of 5 pixels. Increasing the threshold captured more outliers. Decreasing the threshold reduced the number of feature matches especially with rapid movement while using greyscale frames.

- Apply feedback and vary the thresholds used when applying feedback. Feedback is applied when the vector norm of MSCKF position error states surpassed or the vector norm of the of MSCKF tilt error states surpassed $\pi/4$ rad, the INS mechanization states were reset to the corrected position using the MSCKF filter error states. The results shown in this chapter did not apply feedback. Results using feedback with a position error threshold of 1 m and a tilt error

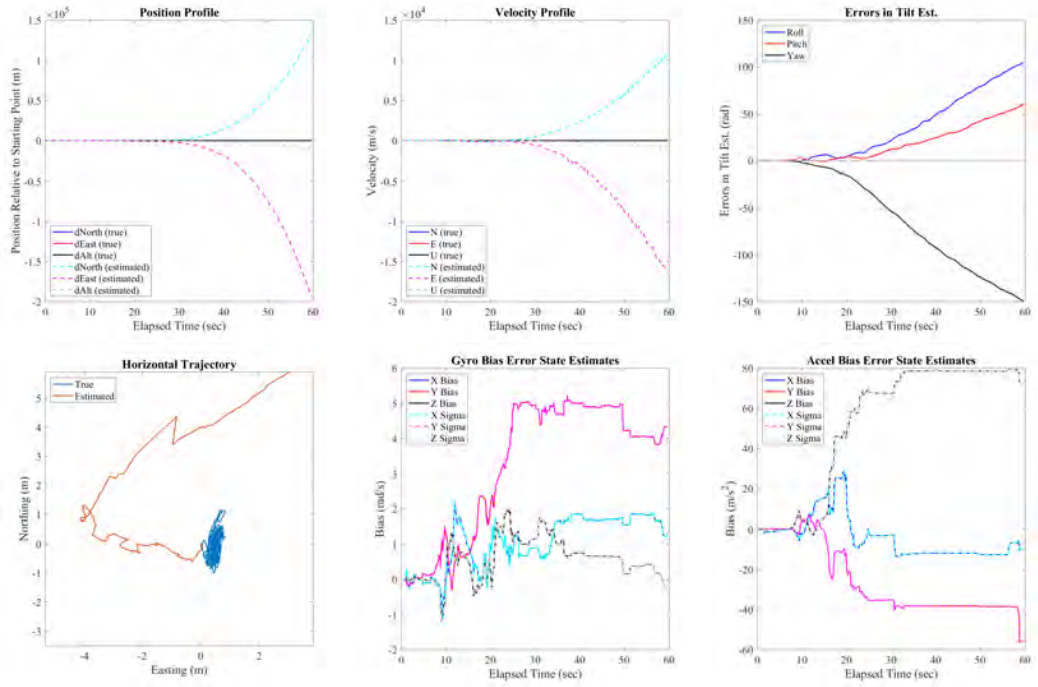

Figure 40: Boxes dataset MSCKF results with greyscale images: the incorporation of the update step does not improve the performance of the MSCKF, but rather makes it dramatically worse compared to the propagation-only results in Figure 37. Further parameter and/or algorithm adjustment is required. Additionally, there is a lack of updates during the latter half of the dataset, as can be easily seen in the flat-lining of accelerometer bias estimates. This is likely due to the increase in rapid and sporadic movements which caused blurring in the images, hindering feature detection and matching.

threshold of $\pi/4$ rad increased the rate of divergence for the system states.

- Vary the time threshold Δt_{IMU} in the motion-compensation algorithm where events are considered to be in the same pose. Recomputing an interpolated pose for a 1 µs difference between events is especially inefficient and unnecessary for creating quality motion-compensated event frames, but too large of a threshold for rapid movement causes blurring. The results shown in this chapter used a threshold of 100 µs.

- Vary the maximum number of events $N_{batch,max}$ and/or the maximum time window length $\Delta t_{batch,max}$ for the batches of events for motion-compensated

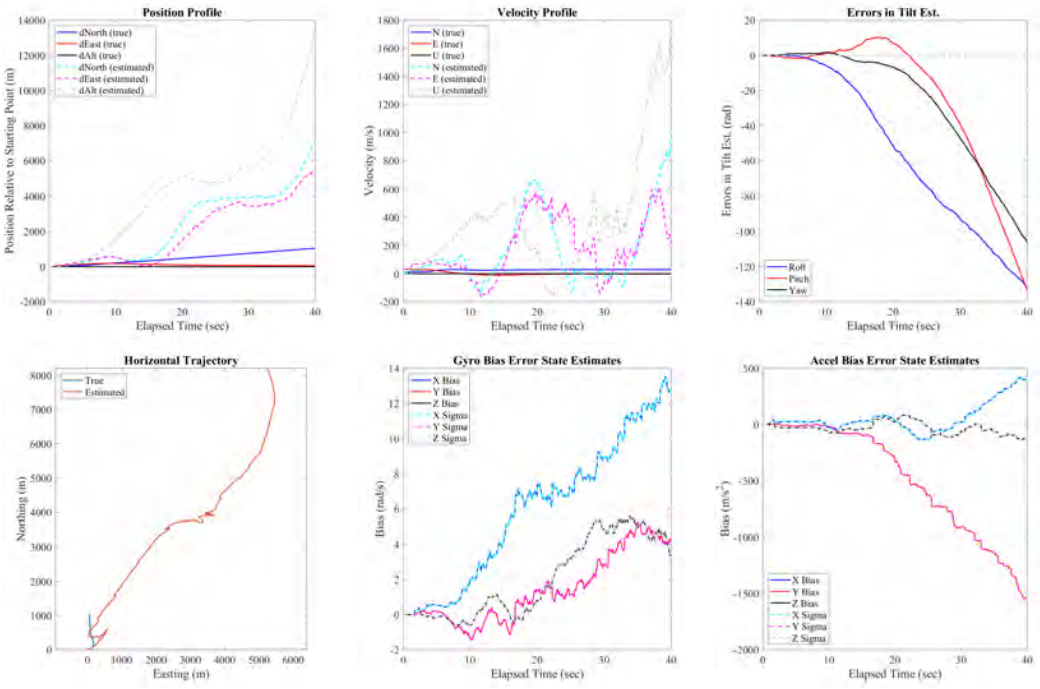

Figure 41: Camp Atterbury dataset MSCKF results with greyscale images: the incorporation of the update step does not improve the performance of the MSCKF, but rather makes it dramatically worse compared to the propagation-only results in Figure 38. Further parameter and/or algorithm adjustment is required.

event frames. The results shown in this chapter for the Boxes dataset used at most 50,000 events or at most a 500 ms time window for each motion-compensated frame. The results shown in this chapter for the Shapes and Camp Atterbury dataset used at most 50,000 events or at most a 500 ms time window for each motion-compensated frame.

- Vary the time and/or event overlap between motion-compensated event frames. For the results shown, a time overlap was used to ensure a maximum time step $\Delta t_{step,max}$ between the start of each event frame of 50 ms, and each event overlapped by at least $N_{step,max}$. If the event rate, due to rapid movement, resulted in more than N_{max} events in $\Delta t_{step,max}$, then the subsequent frame would overlap by $N_{step,max}$ events. For low event rates, an issue arose when the

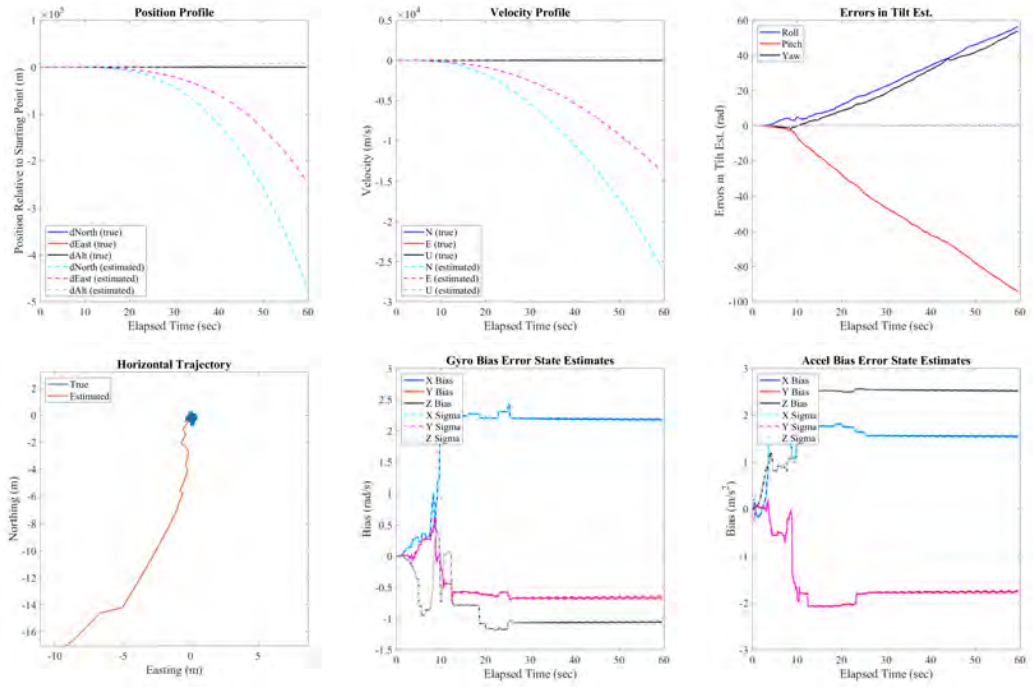

Figure 42: Shapes dataset MSCKF results with event frames: utilizing event frames for measurements does not improve the performance of the MSCKF, but rather makes it dramatically worse than even the poor performance with greyscale images in Figure 39. Further parameter and/or algorithm adjustment is required. Additionally, once reasonably accurate tracking of the system states is lost, the motion-compensated event frames become completely corrupted and add no value, preventing any chance at recovery.

number of events occurring over the maximum time window $\Delta t_{batch,max}$ was less than the minimal overlap $N_{step,max}$.

- Apply a threshold for the residual calculated in (58) from feature tracks where feature tracks exceeding the threshold are discarded as outliers.

- Execute analysis on the evolution of the covariance through state augmentation, propagation, and updates. The results shown implemented $P = (P + P^T)/2$; and $P = P + \epsilon \mathbf{I}$ to help maintain symmetry and positive semi-definite nature of the covariance matrix.

- Skipping the QR decomposition described in (69) to (72).

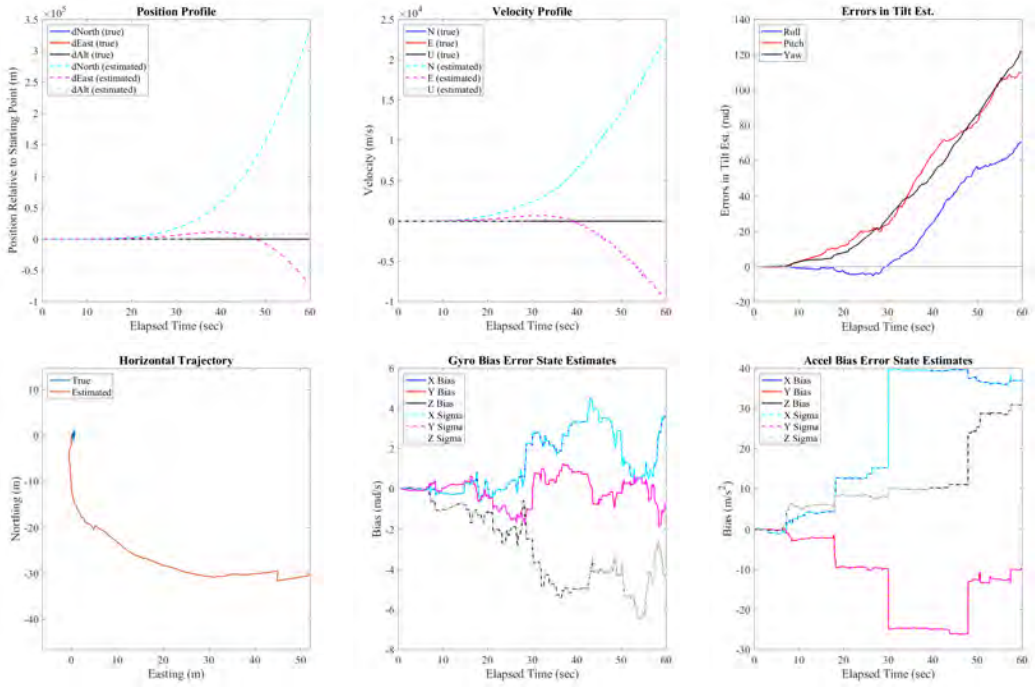

Figure 43: Boxes dataset MSCKF results with event frames: utilizing event frames for measurements does not improve the performance of the MSCKF, but rather makes it dramatically worse than even the poor performance with greyscale images in Figure 40. Further parameter and/or algorithm adjustment is required. Additionally, once reasonably accurate tracking of the system states is lost, the motion-compensated event frames become completely corrupted and add no value, preventing any chance at recovery.

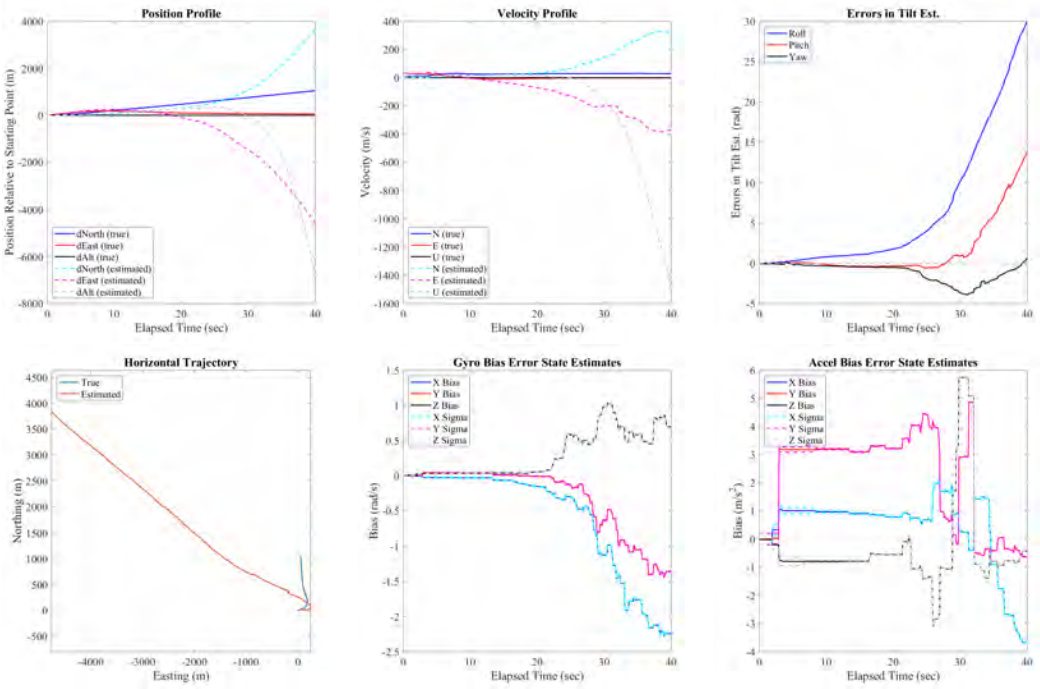

Figure 44: Camp Atterbury dataset MSCKF results with event frames: utilizing event frames for measurements does not improve the performance of the MSCKF, but rather makes it dramatically worse than even the poor performance with greyscale images in Figure 41. Further parameter and/or algorithm adjustment is required. Additionally, once reasonably accurate tracking of the system states is lost, the motion-compensated event frames become completely corrupted and add no value, preventing any chance at recovery.

V. Conclusions

In this thesis, an event-based visual-inertial odometry (EVIO) pipeline, primarily motivated by Zhu's [4] and Rebecq's [6] state-of-the-art research, was implemented and tested. The front-end of this pipeline was fed by the output of a new type of visual sensor, called an event-based camera. An event-based camera operates under a unique paradigm, where each pixel in the sensor independently and asynchronously outputs ON or OFF events for each rise or fall in light log intensity past an established threshold. Batches of event output from the event-based camera were compiled into frames by accumulating events occurring at each pixel. The clarity of these frames was improved by compensating for the motion of the camera calculated through an integrated Micro-Electro-Mechanical Systems (MEMS) inertial measurement unit (IMU). KAZE features [17] were detected on each frame and matched to previously detected features to generate feature tracks to be used by the back-end of the EVIO pipeline.

The back-end of the EVIO pipeline implemented the Multi-State Constrained Kalman Filter (MSCKF) [1] using the Scorpion library of classes and functions developed by the Autonomy and Navigation Technology (ANT) Center at Air Force Institute of Technology (AFIT). The MSCKF estimates errors for the vehicle states calculated by an inertial navigation system (INS) mechanization of the IMU. The feature tracks feed a least-squares estimation of a feature position, which is then used to calculate the residual for Kalman filter update equations.

Additionally, this research conducted the ANT Center's first data collection using an event-based camera, accomplished on a flight test of a fixed-wing unmanned aerial vehicle (UAV). This event-based camera dataset included event-based camera data, greyscale images, and IMU output, all synchronized with high-resolution position information from a PIKSI Multi-Global Navigation Satellite System (GNSS) and

orientation information from the UAV flight controller.

Results for the implementation of motion-compensated frames show clearly contrasted images with well-defined edges, where other methods often result in either sparse and/or blurred imagery. This solution to event-based camera visualization is a viable avenue for clear interpretation of this unique type of data.

The results for this implementation of the MSCKF, fed with greyscale images from a classical "frame-based" camera, resulted in poorer vehicle state estimation than INS mechanization alone. The results were particularly poor for rapid camera movement that resulted in few matched features between scenes. The results of testing the MSCKF with motion-compensated event-based frames resulted in even worse results, as minor errors in the vehicle state corrupted any clarity to be gained from motion-compensation.

Various parameter adjustments and simple outlier rejection methods were attempted, including varying $\sigma_{\texttt{im}}$ for the Kalman filter update equations, and adjusting the various parameters included in the motion compensation algorithm. However, none of these adjustments provided the performance improvements expected from this implementation.

5.1 Future Work

Other adjustments to the motion-compensation and MSCKF implementation in this work that were considered but not fully implemented in this research due to time constraints include the following.

- Incorporate 2-point random sample consensus (RANSAC) [154] on the feature tracks to remove outliers.
- Implement a Mahalanobis distance threshold check [155] during feature matching.

- Utilize Zhu's Robot Operating System (ROS) version of the MSCKF [4].
- Utilize dynamically adjusted threshold parameters for motion-compensated event frames: the time window $\Delta t_{\texttt{IMU}}$ for pose interpolation; the event batch size thresholds $N_{\texttt{batch,max}}$ and $\Delta t_{\texttt{batch,max}}$; and the event batch step thresholds $\Delta t_{\texttt{step,max}}$ and $N_{\texttt{step,max}}$.
- Extended evaluation of the application of QR decomposition described in (69) to (72) and singular value decomposition in (66) and (67) in Section 3.4.6.2.
- Retain all descriptors for a given matched feature, instead of only the initial descriptor, for possibly higher feature matching rates.
- Implement a method to recognize when motion-compensation causes the majority of events to be more blurred than aligned, and correct the vehicle states accordingly.

5.1.1 Event-Based Feature Detection

Separate from improving the implementation of the EVIO pipeline described in this work, alternative approaches to feature detection on the stream of events require more investigation. Other state-of-the-art avenues of research for utilizing event-based cameras for visual odometry (VO) implement feature detection on a stream of events while recognizing that combining batches of events into images and then detecting features is likely not the most optimal method [70]. While motion compensation does take some timing information into account, an ideal feature detection would be independent of any other sensor and would take into account the polarity, timing, and pixel location of each event but only relative to temporally and spatially neighboring events, instead of operating on a complete frame of events. An obstacle to this avenue of research is that it cannot capitalize on the various feature detection methods available for classical frames, and care needs to be taken to avoid computationally

expensive operations on every event and instead focus on efficiently identifying the relevance of each event.

Appendix A. Additional Results

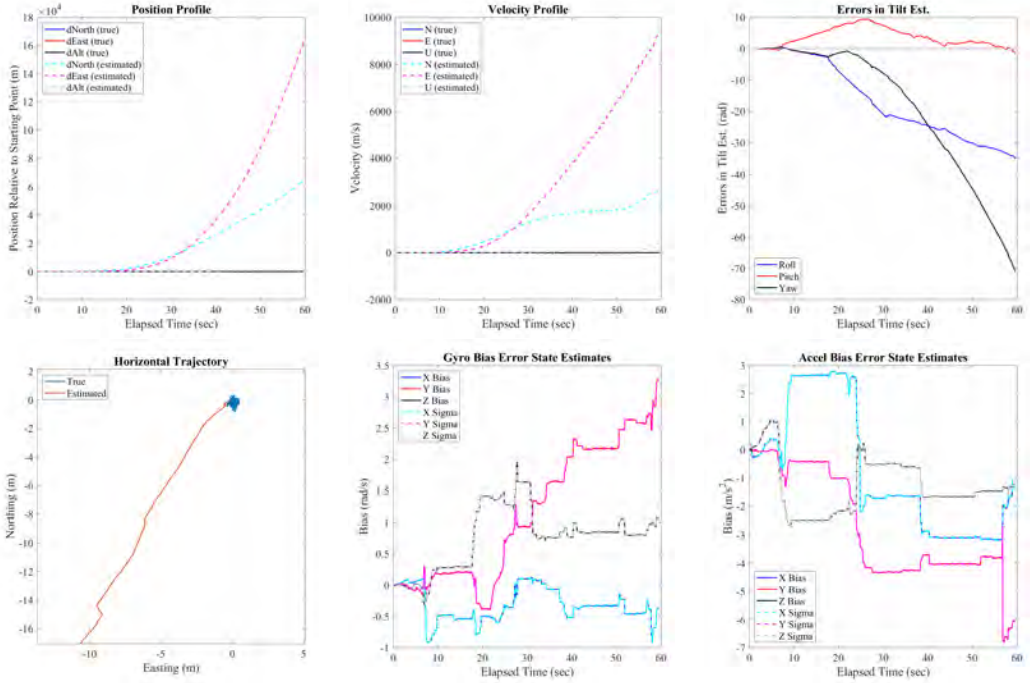

Figure 45: Shapes dataset MSCKF results with greyscale images, discarding outlier features

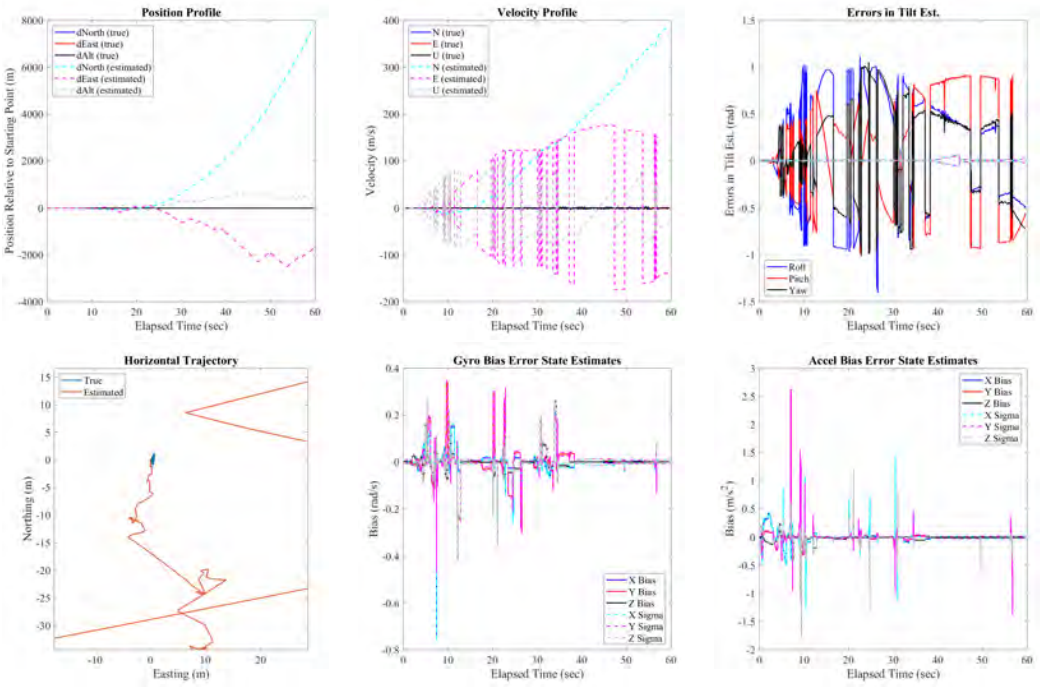

Figure 46: Boxes dataset MSCKF results with greyscale images, discarding outlier features

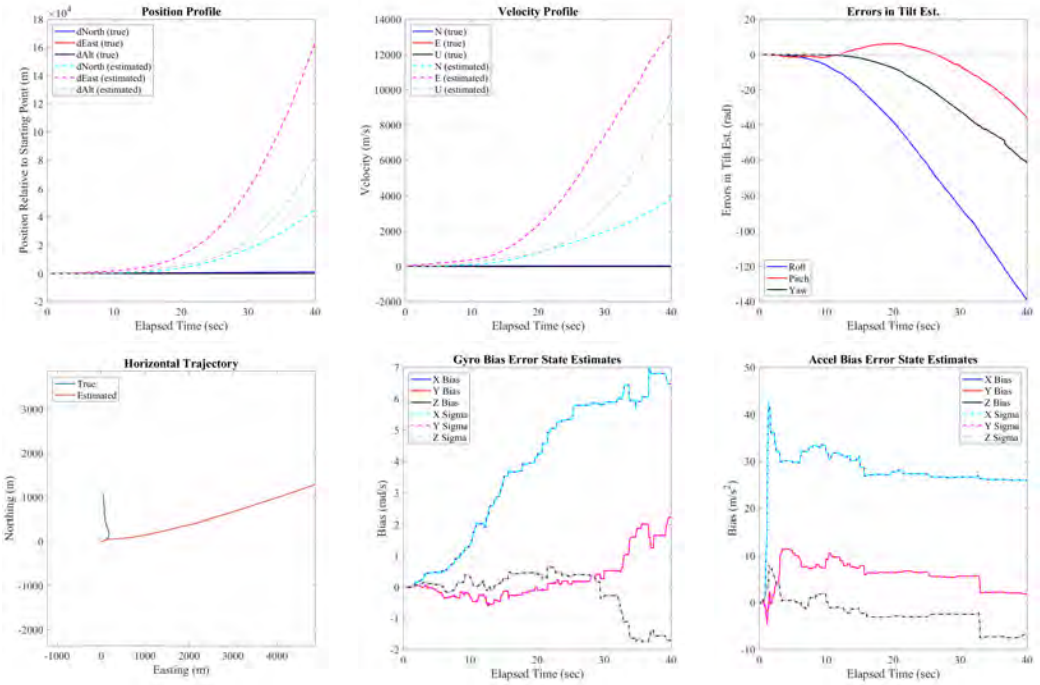

Figure 47: Camp Atterbury dataset MSCKF results with greyscale images, discarding outlier features

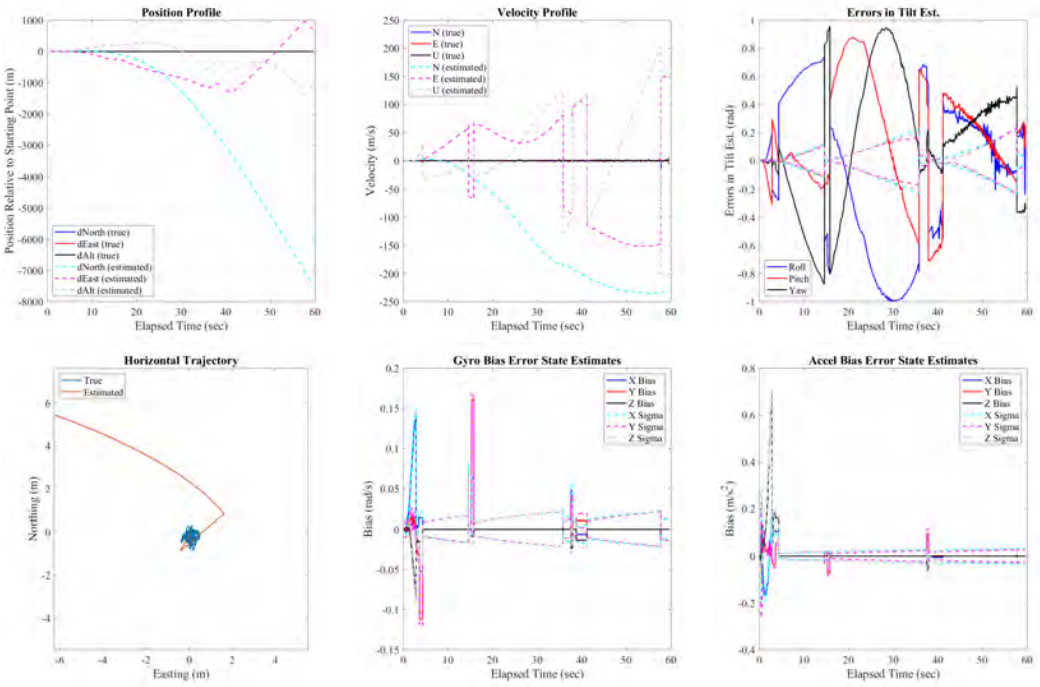

Figure 48: Shapes dataset MSCKF results with event frames, discarding outlier features

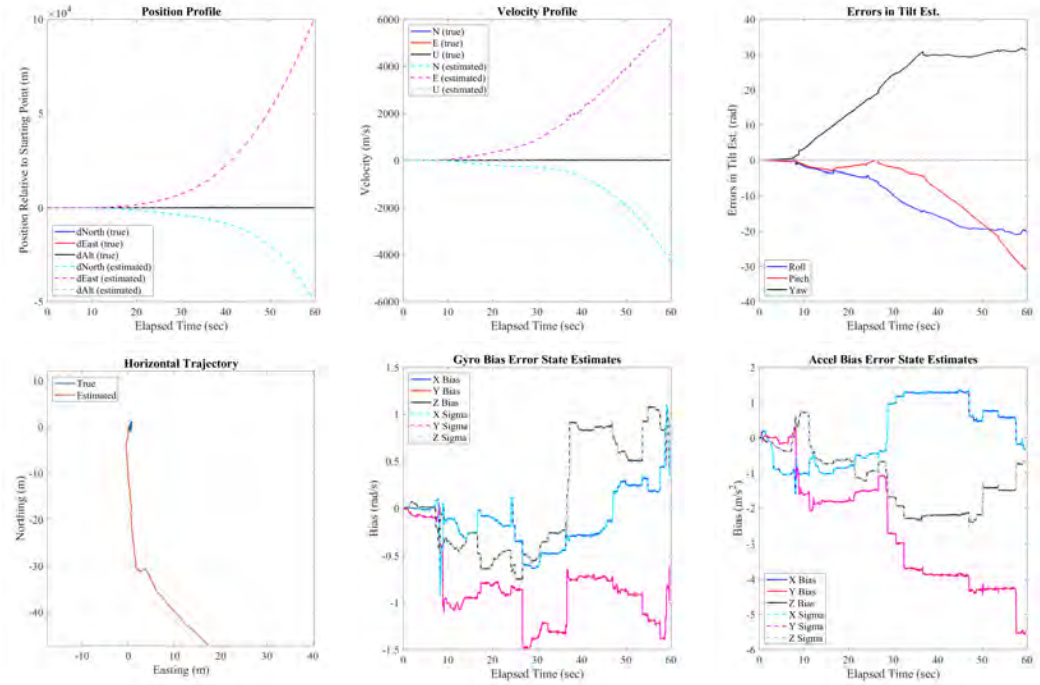

Figure 49: Boxes dataset MSCKF results with event frames, discarding outlier features

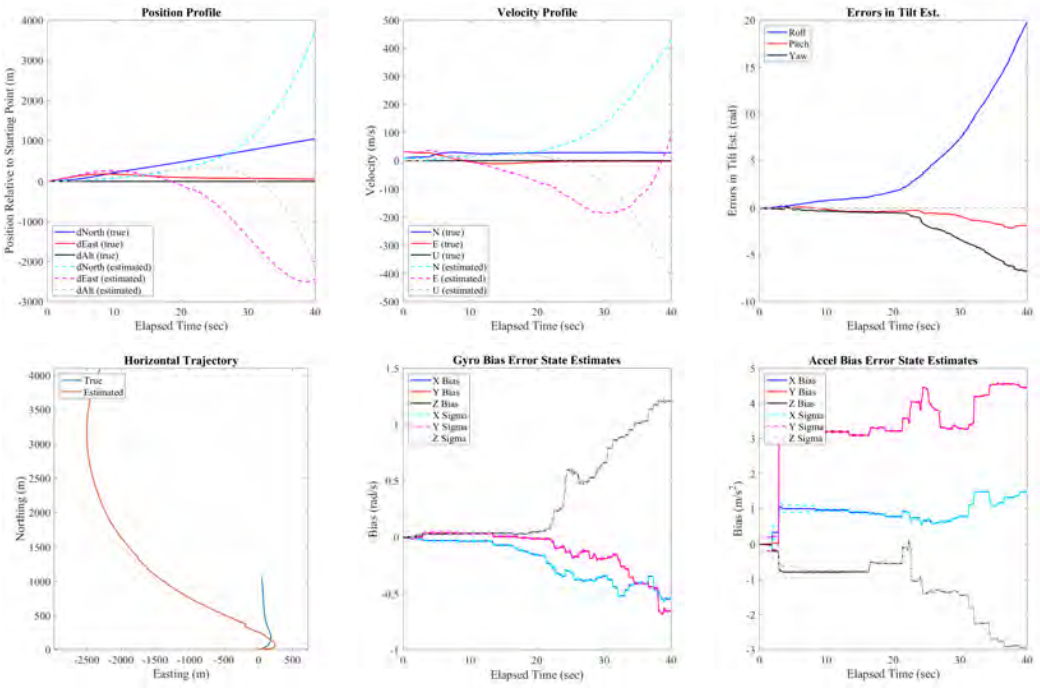

Figure 50: Camp Atterbury dataset MSCKF results with event frames, discarding outlier features

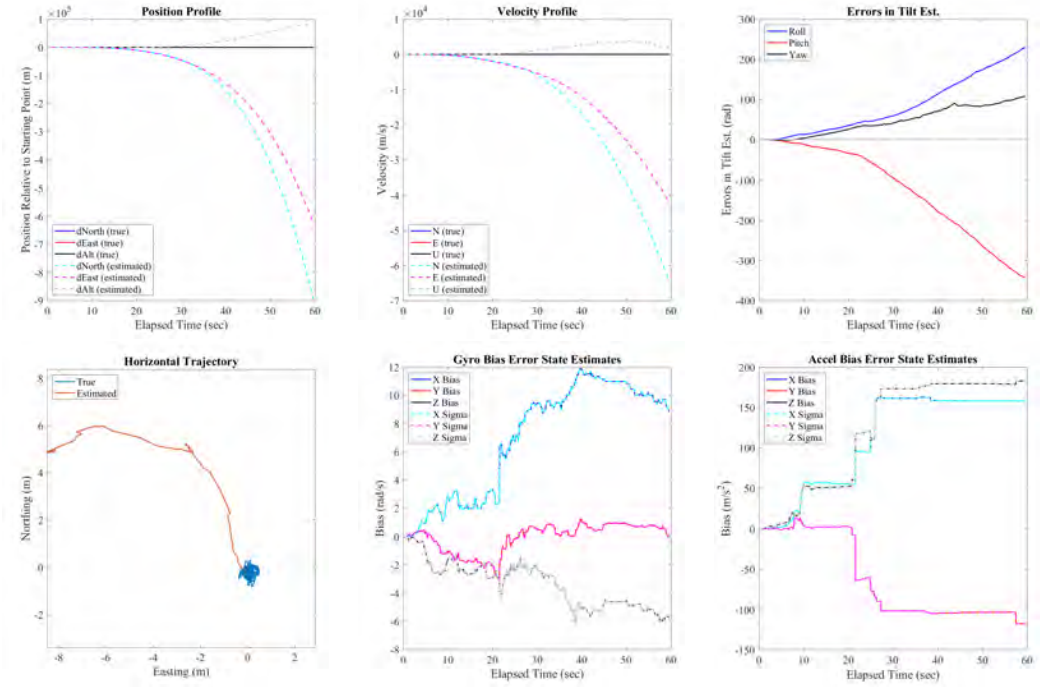

Figure 51: Shapes dataset MSCKF results with greyscale images, not using QR decomposition

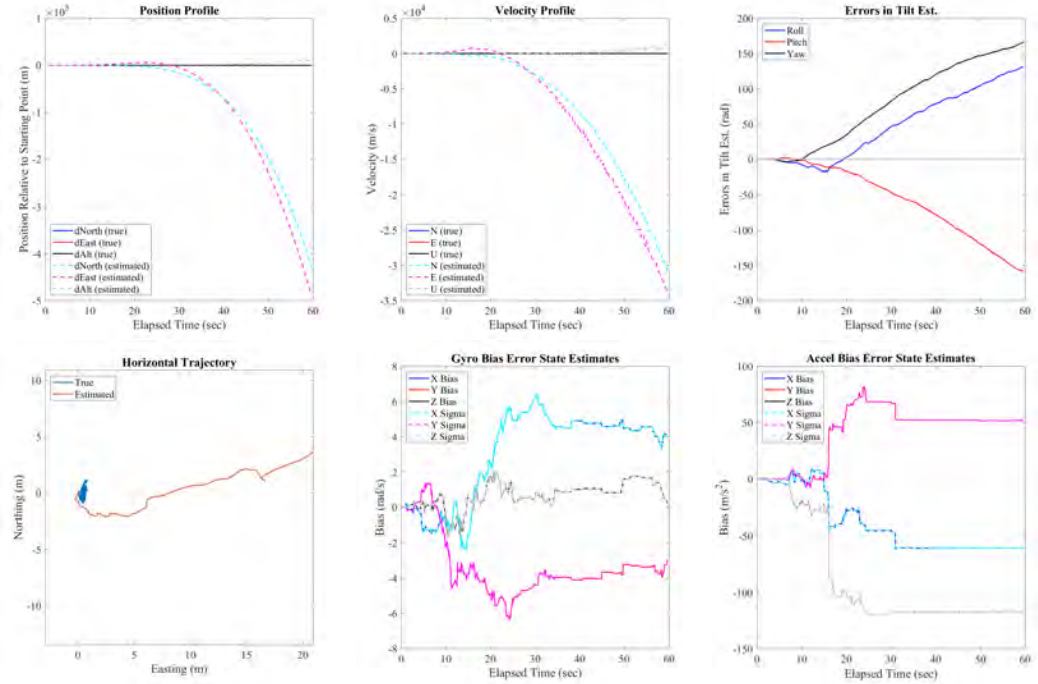

Figure 52: Boxes dataset MSCKF results with greyscale images, not using QR decomposition

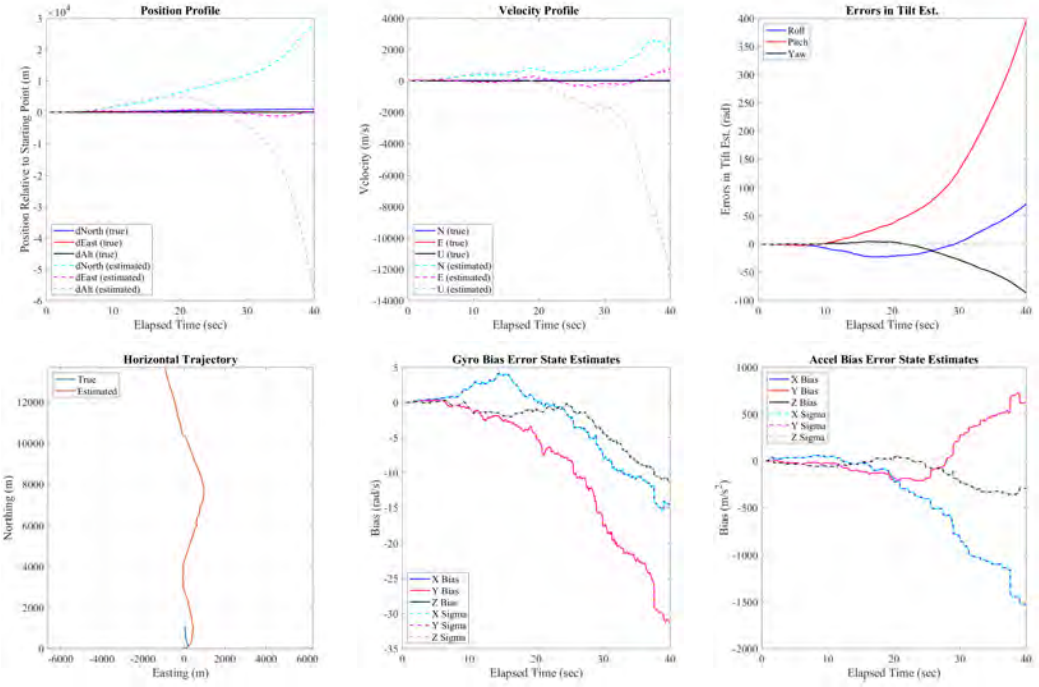

Figure 53: Camp Atterbury dataset MSCKF results with greyscale images, not using QR decomposition

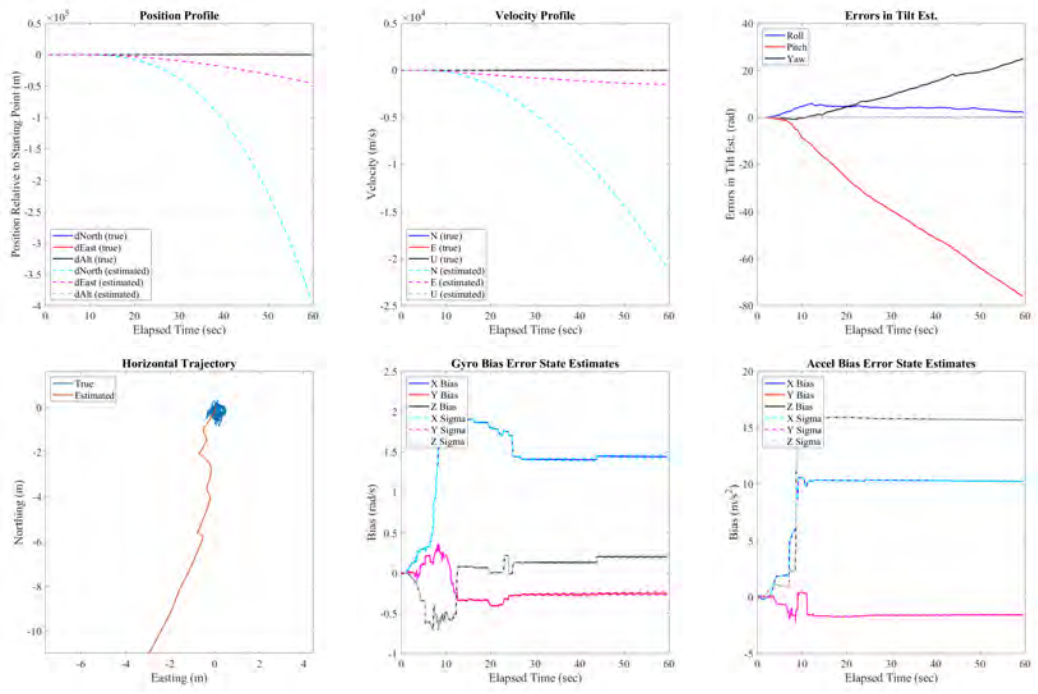

Figure 54: Shapes dataset MSCKF results with event frames, not using QR decomposition

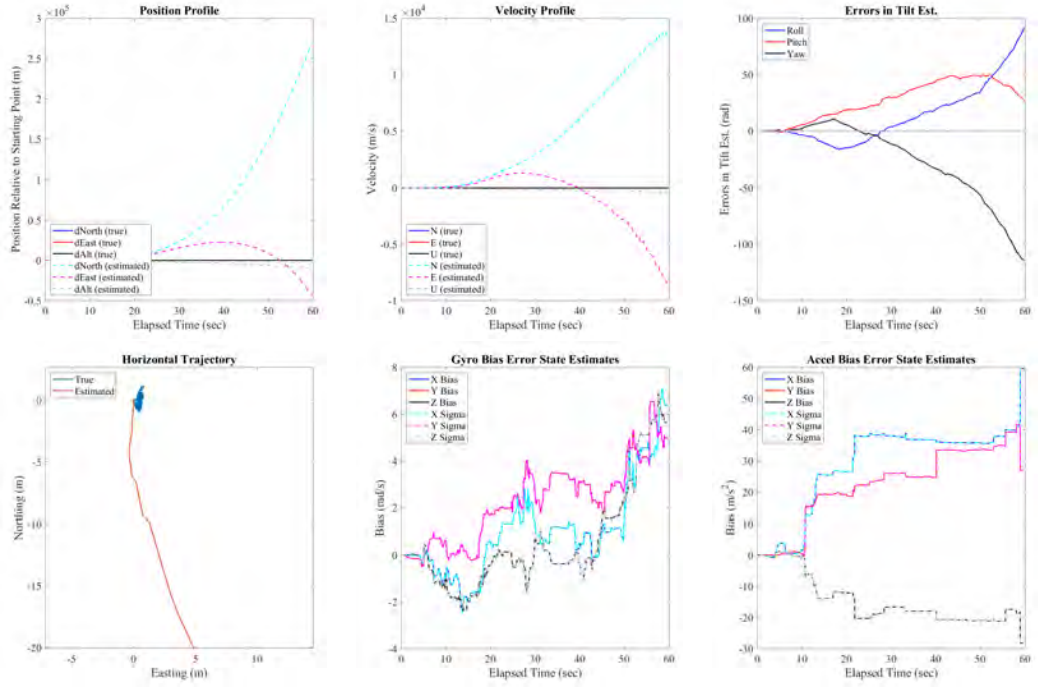

Figure 55: Boxes dataset MSCKF results with event frames, not using QR decomposition

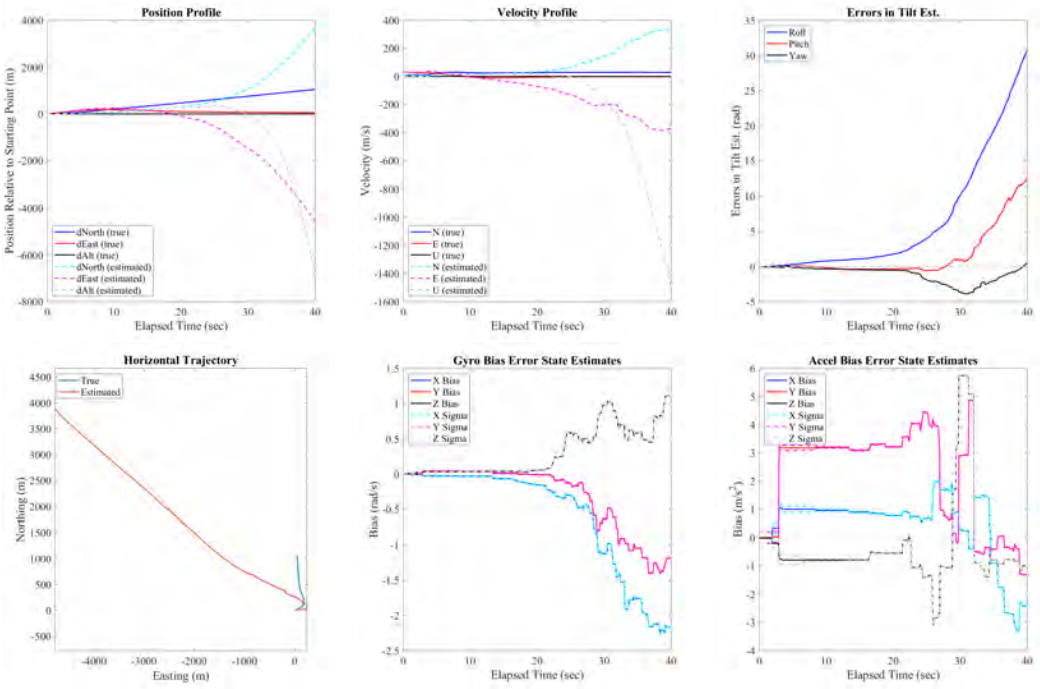

Figure 56: Camp Atterbury dataset MSCKF results with event frames, not using QR decomposition

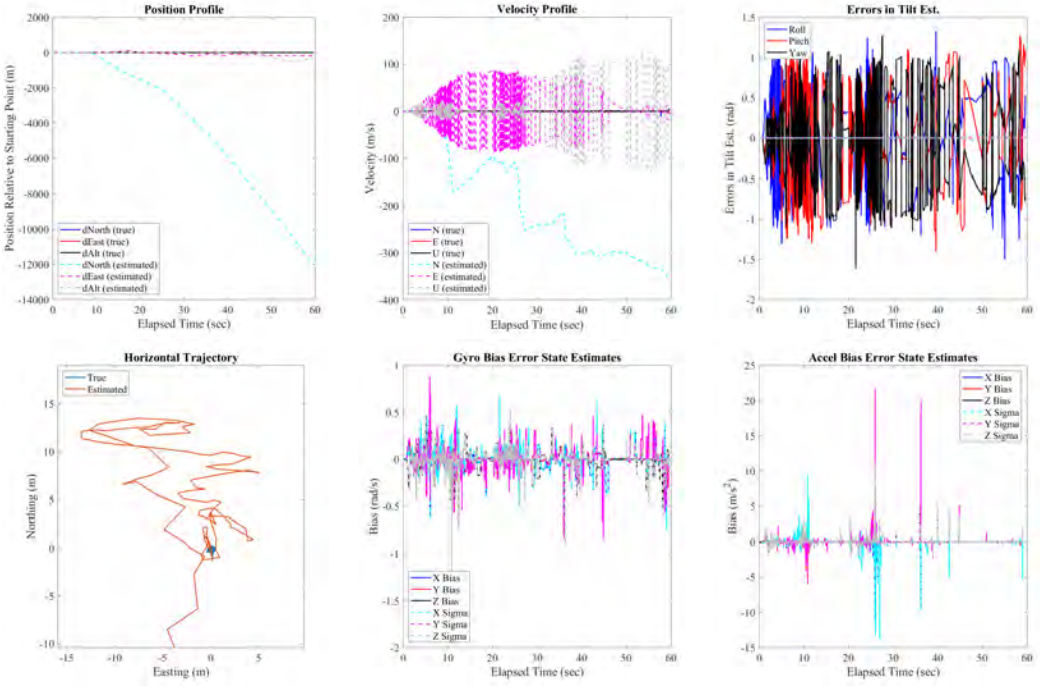

Figure 57: Shapes dataset MSCKF results with greyscale images, using feedback

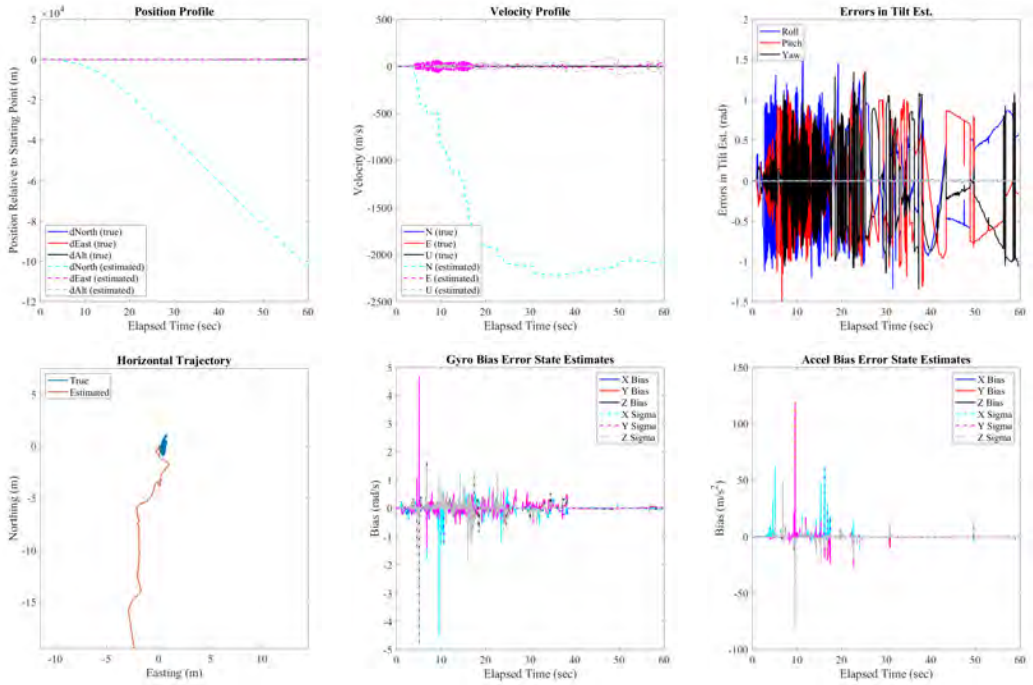

Figure 58: Boxes dataset MSCKF results with greyscale images, using feedback

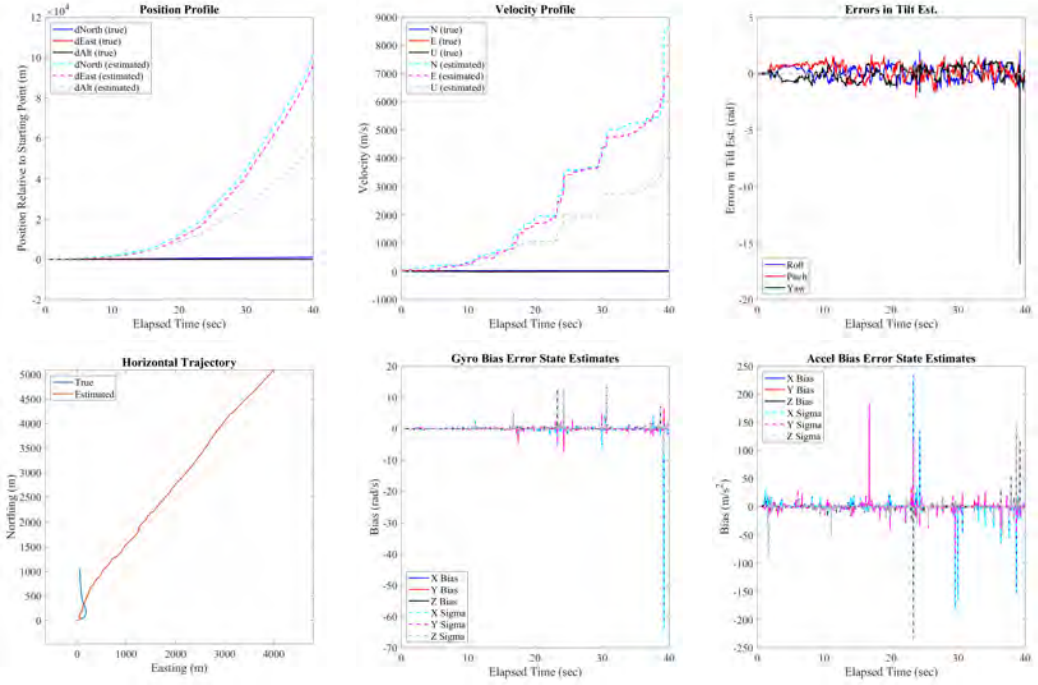

Figure 59: Camp Atterbury dataset MSCKF results with greyscale images, using feedback

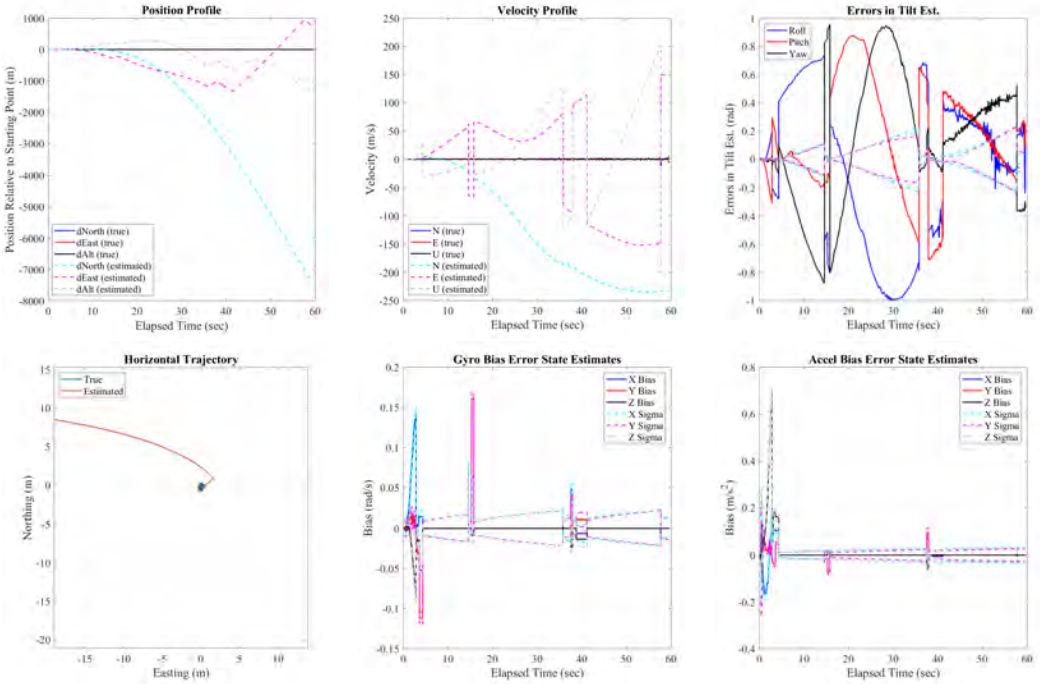

Figure 60: Shapes dataset MSCKF results with event frames, using feedback

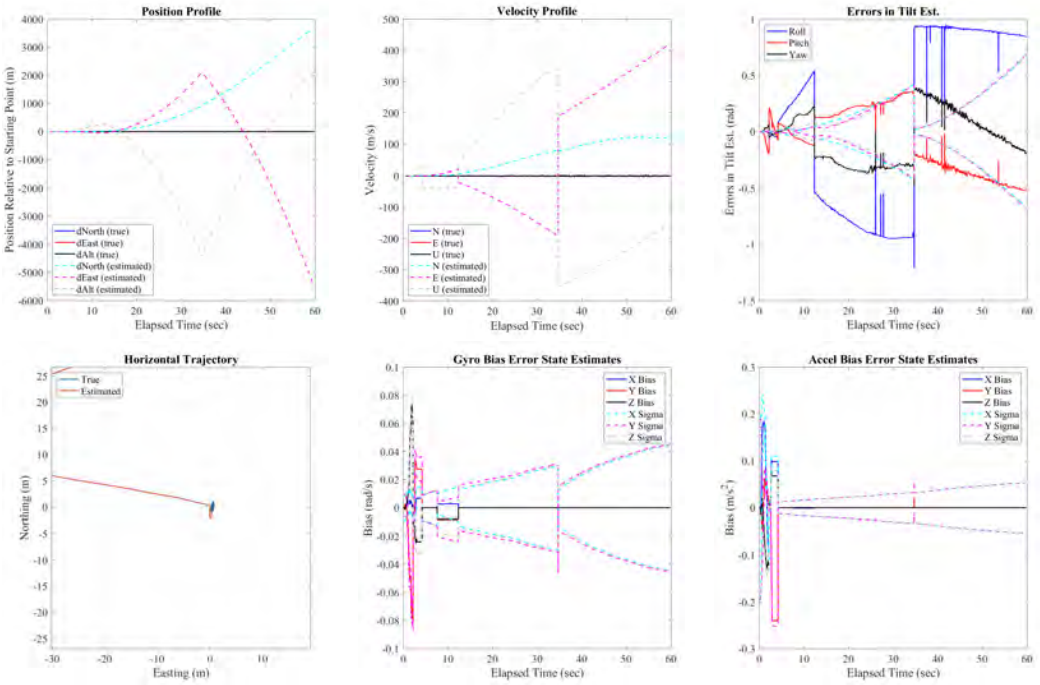

Figure 61: Boxes dataset MSCKF results with event frames, using feedback

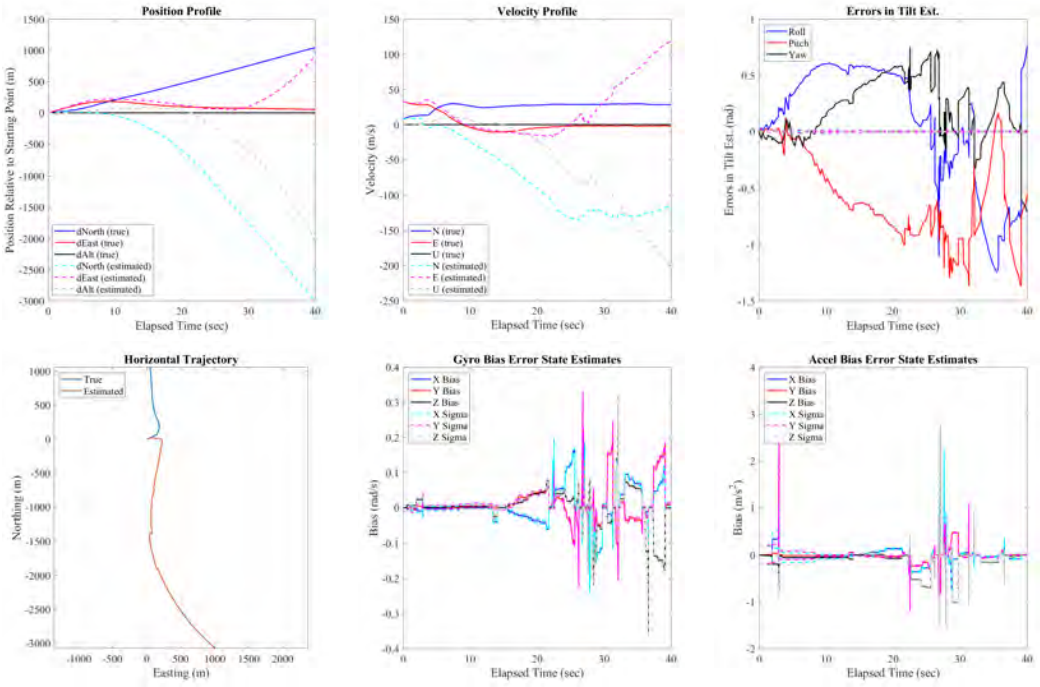

Figure 62: Camp Atterbury dataset MSCKF results with event frames, using feedback

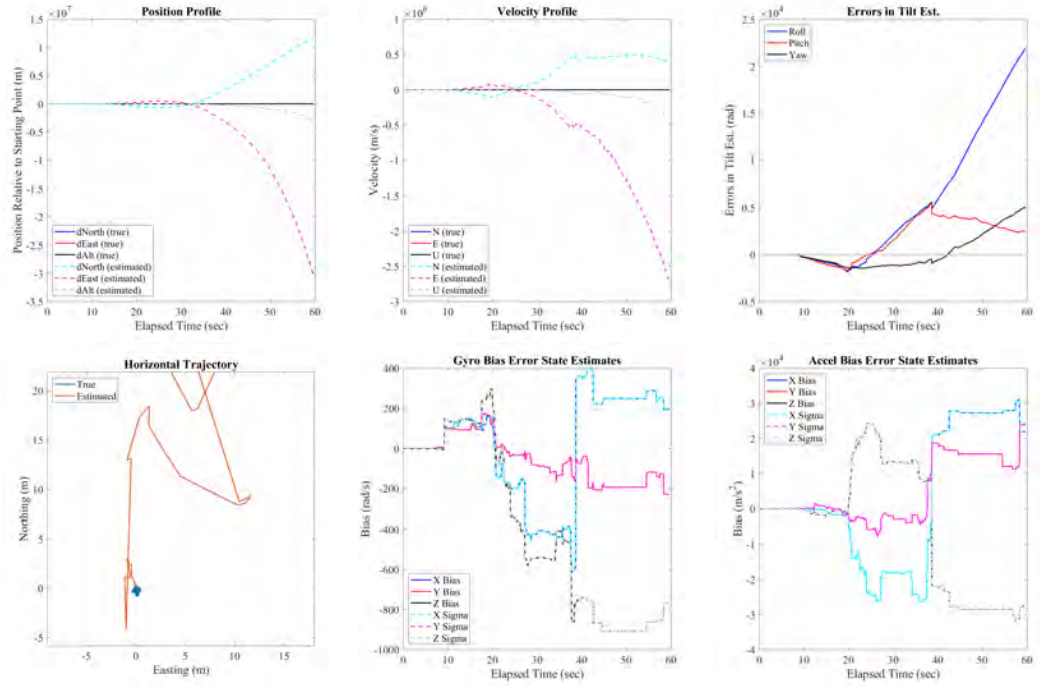

Figure 63: Shapes dataset MSCKF results with greyscale images, using $\sigma_{\texttt{im}} = 1 \times 10^{-6}$

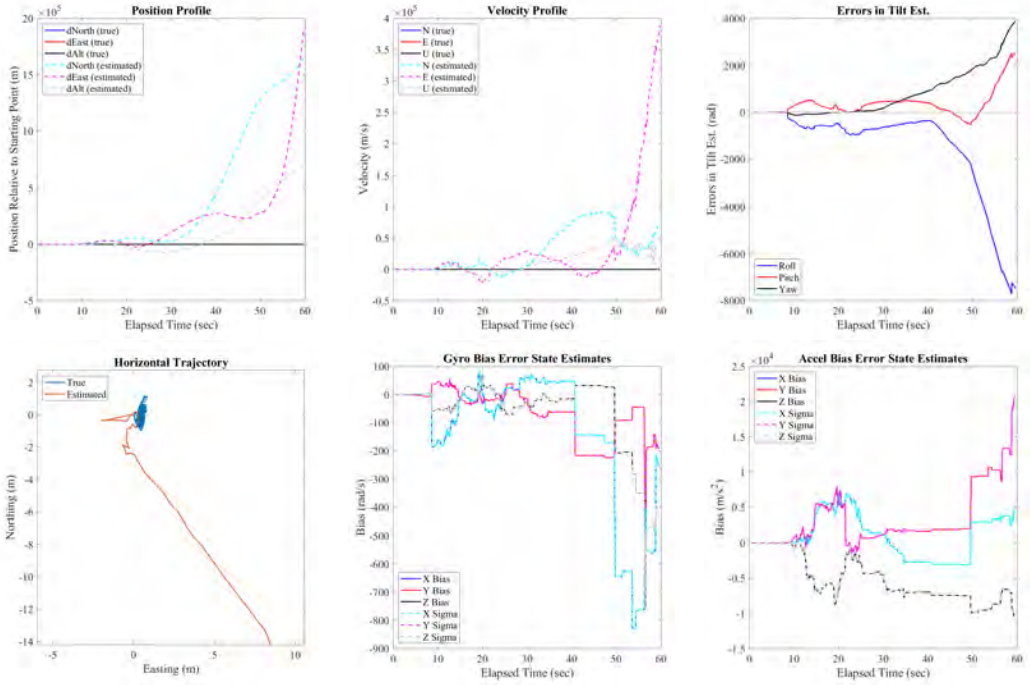

Figure 64: Boxes dataset MSCKF results with greyscale images, using $\sigma_{\texttt{im}} = 1 \times 10^{-6}$

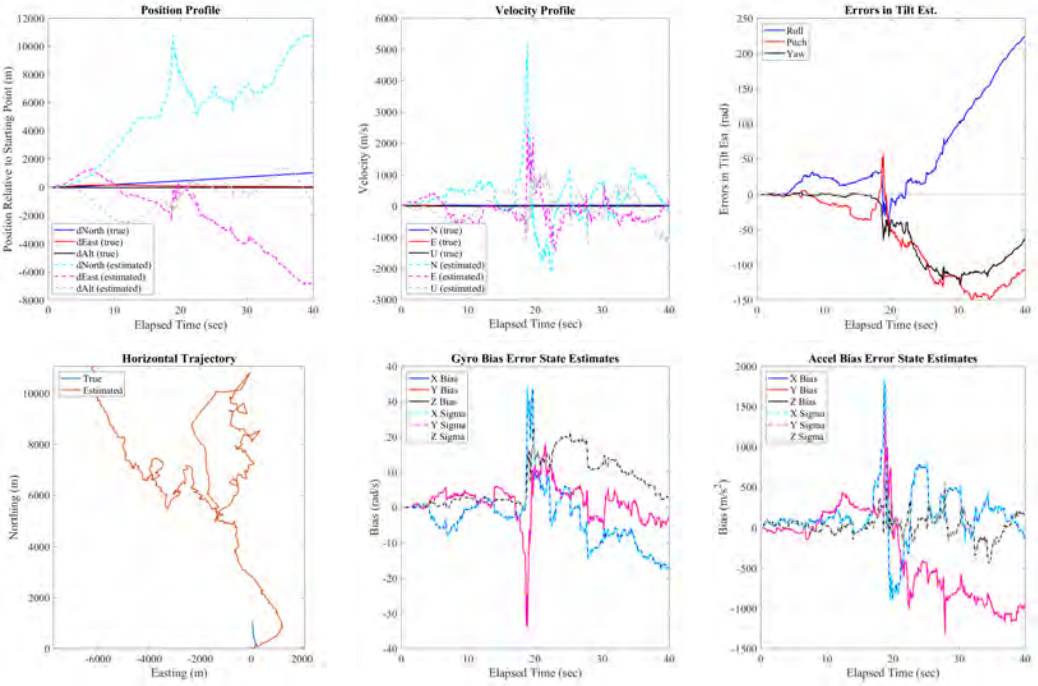

Figure 65: Camp Atterbury dataset MSCKF results with greyscale images, using $\sigma_{\texttt{im}} = 1 \times 10^{-6}$

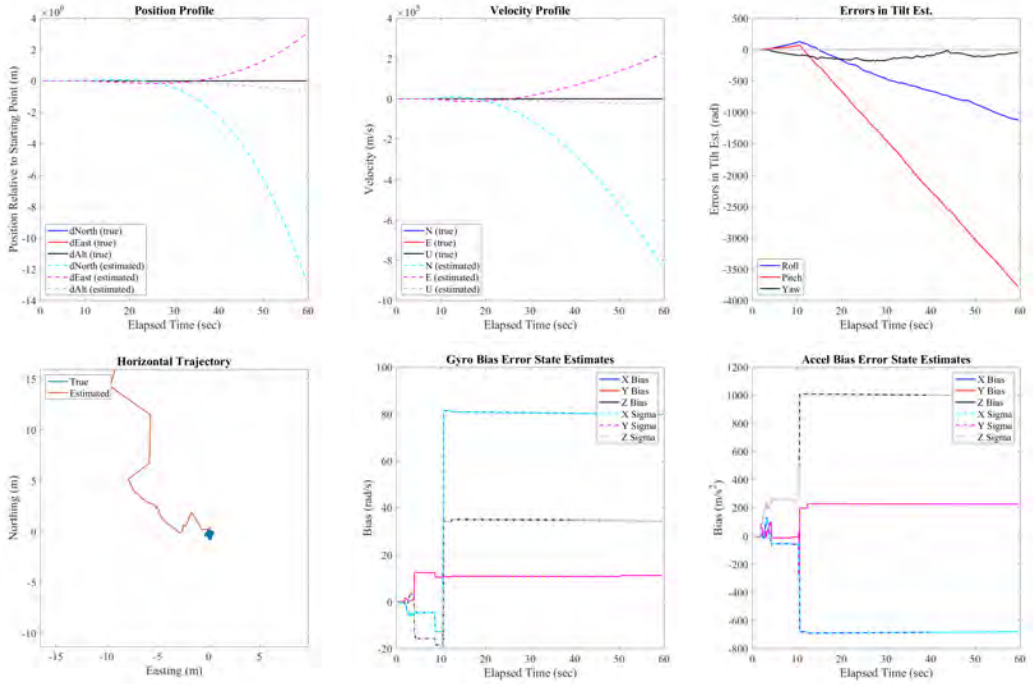

Figure 66: Shapes dataset MSCKF results with event frames, using $\sigma_{\texttt{im}} = 1 \times 10^{-6}$

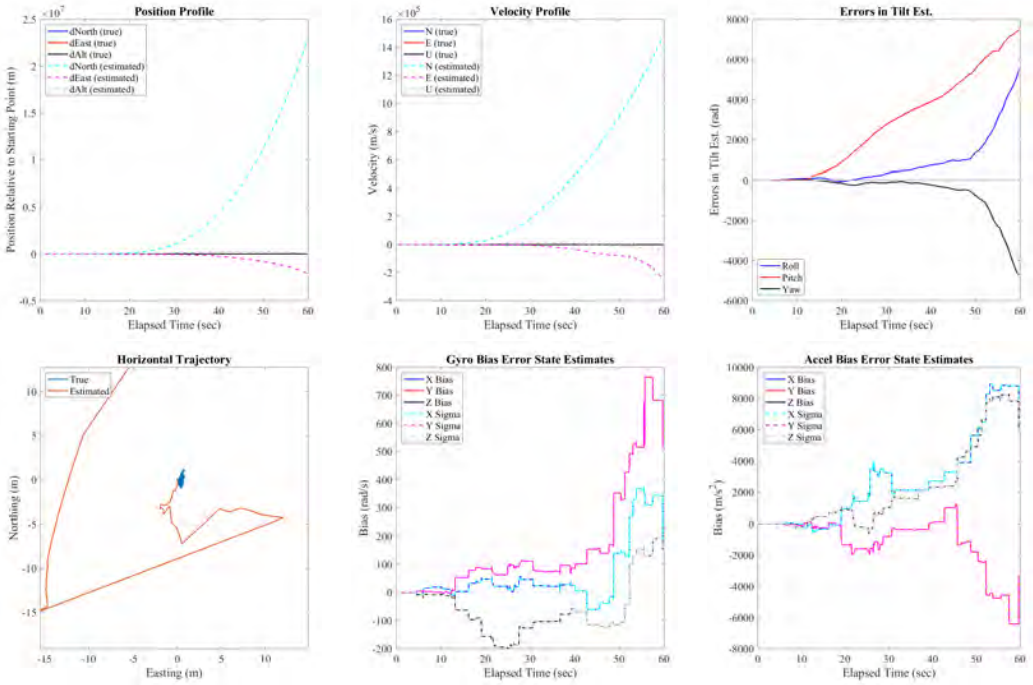

Figure 67: Boxes dataset MSCKF results with event frames, using $\sigma_{\texttt{im}} = 1 \times 10^{-6}$

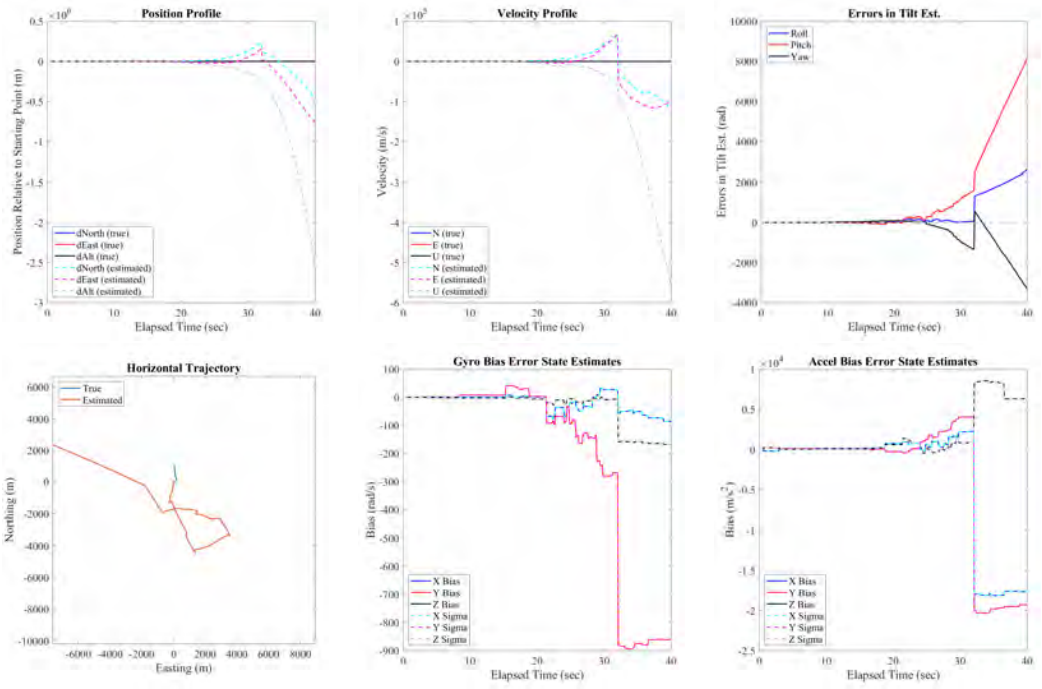

Figure 68: Camp Atterbury dataset MSCKF results with event frames, using $\sigma_{\texttt{im}} = 1 \times 10^{-6}$

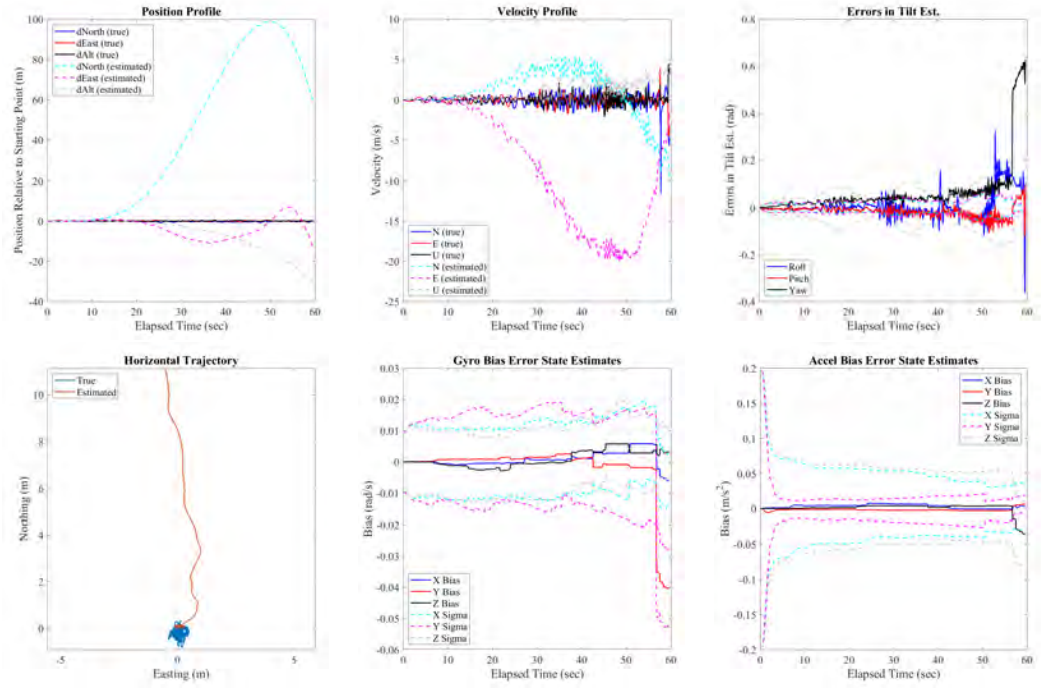

Figure 69: Shapes dataset MSCKF results with greyscale images, using $\sigma_{\texttt{im}} = 1$

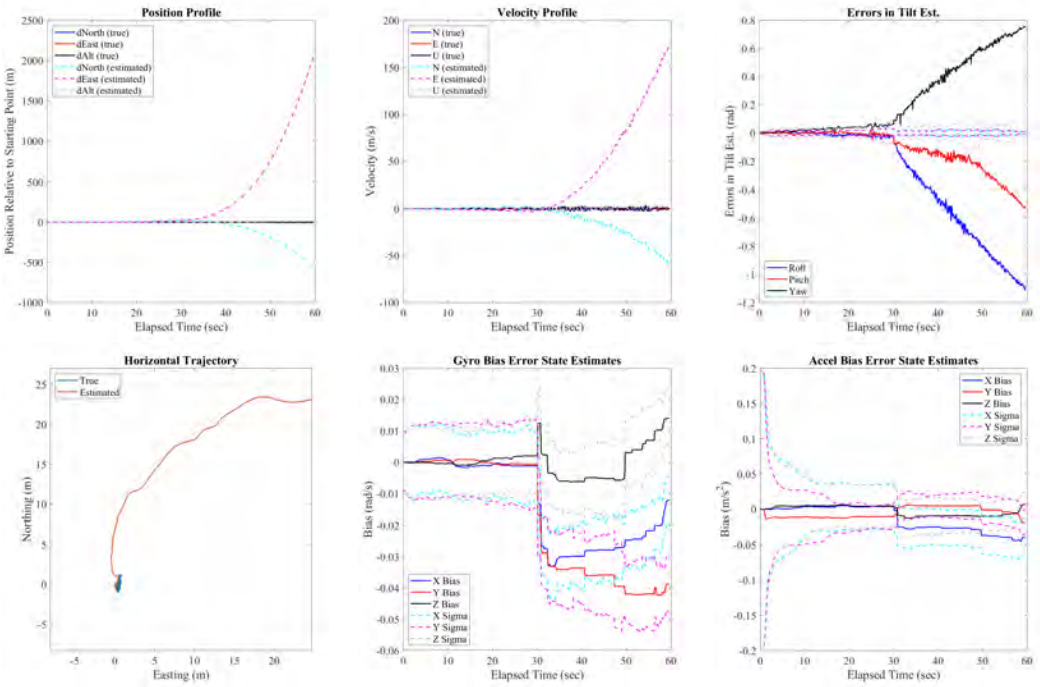

Figure 70: Boxes dataset MSCKF results with greyscale images, using $\sigma_{\texttt{im}} = 1$

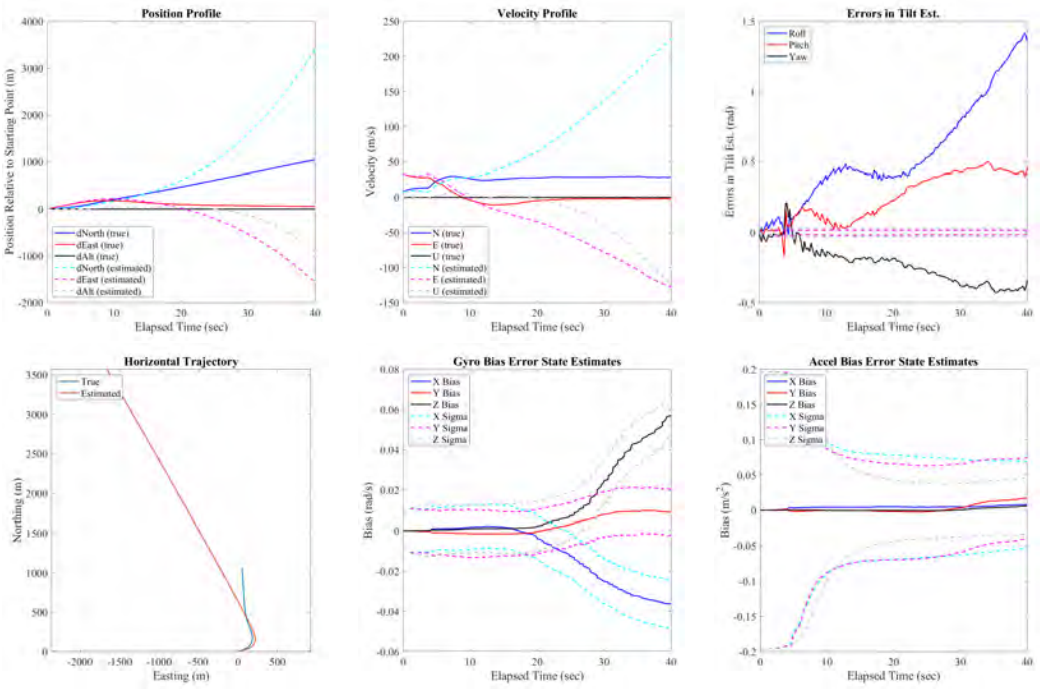

Figure 71: Camp Atterbury dataset MSCKF results with greyscale images, using $\sigma_{\texttt{im}} = 1$

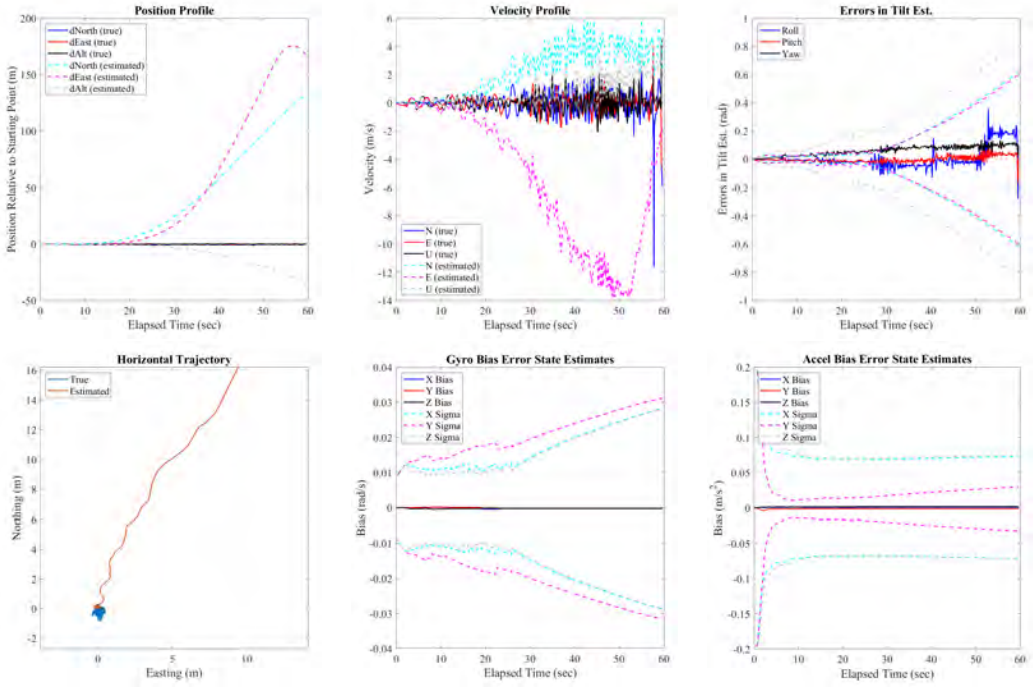

Figure 72: Shapes dataset MSCKF results with event frames, using $\sigma_{\texttt{im}} = 1$

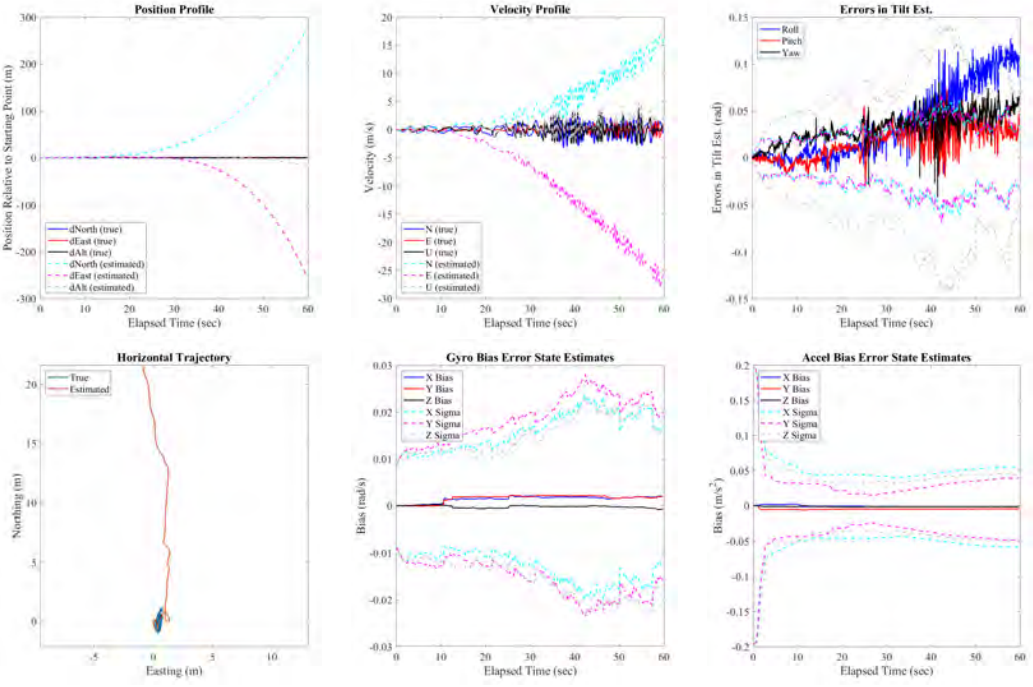

Figure 73: Boxes dataset MSCKF results with event frames, using $\sigma_{\texttt{im}} = 1$

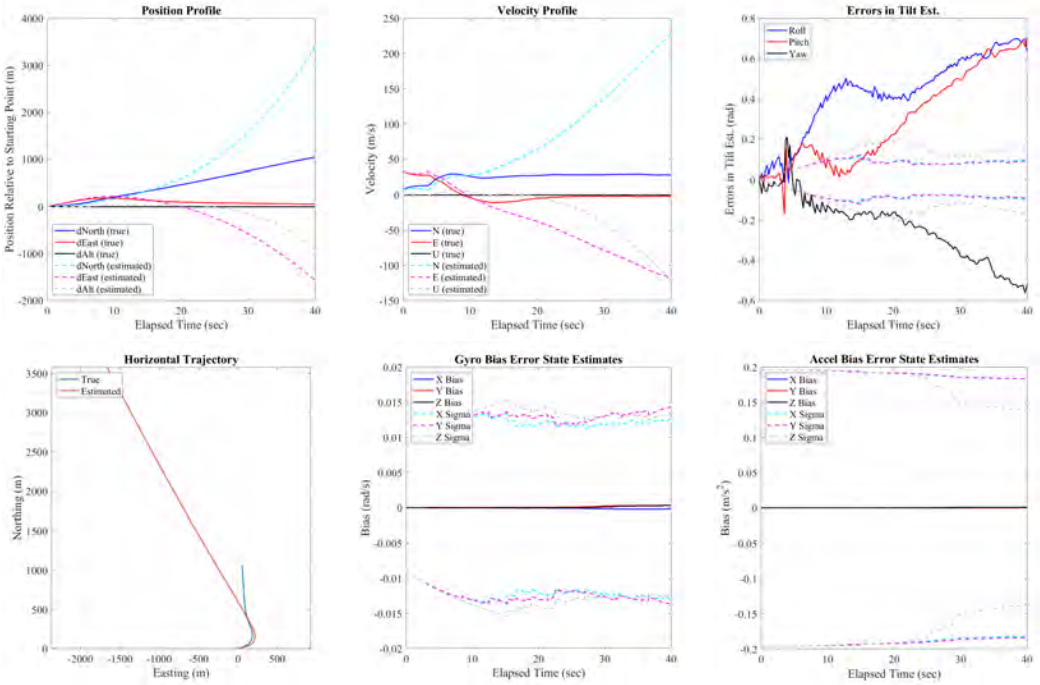

Figure 74: Camp Atterbury dataset MSCKF results with event frames, using $\sigma_{\texttt{im}} = 1$

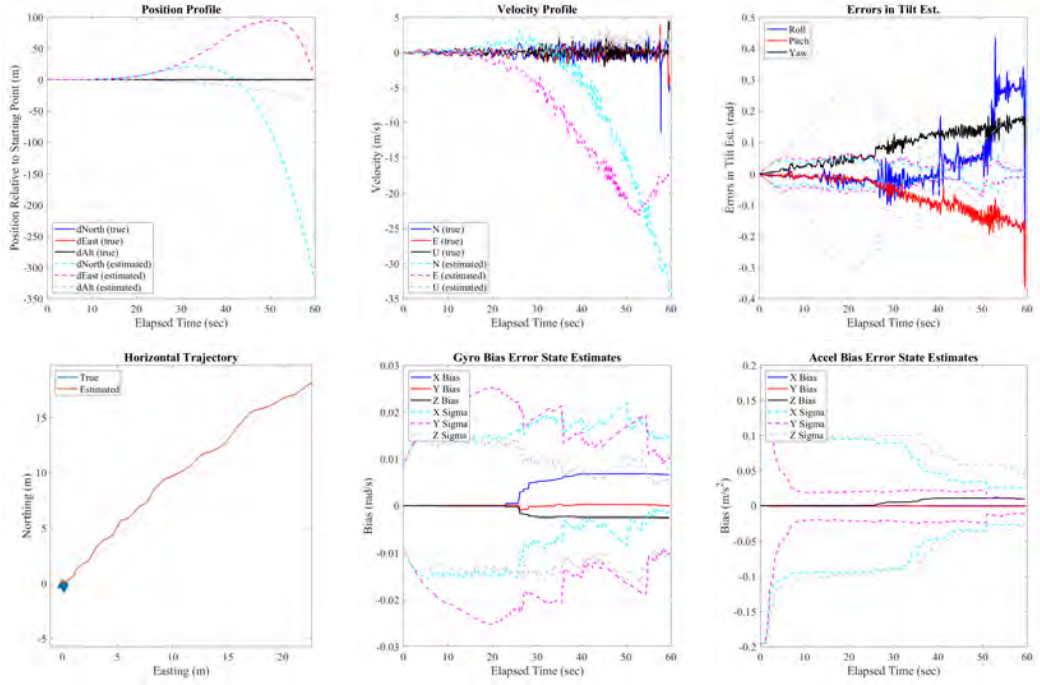

Figure 75: Shapes dataset MSCKF results with greyscale images, using $\sigma_{\texttt{im}} = 10$

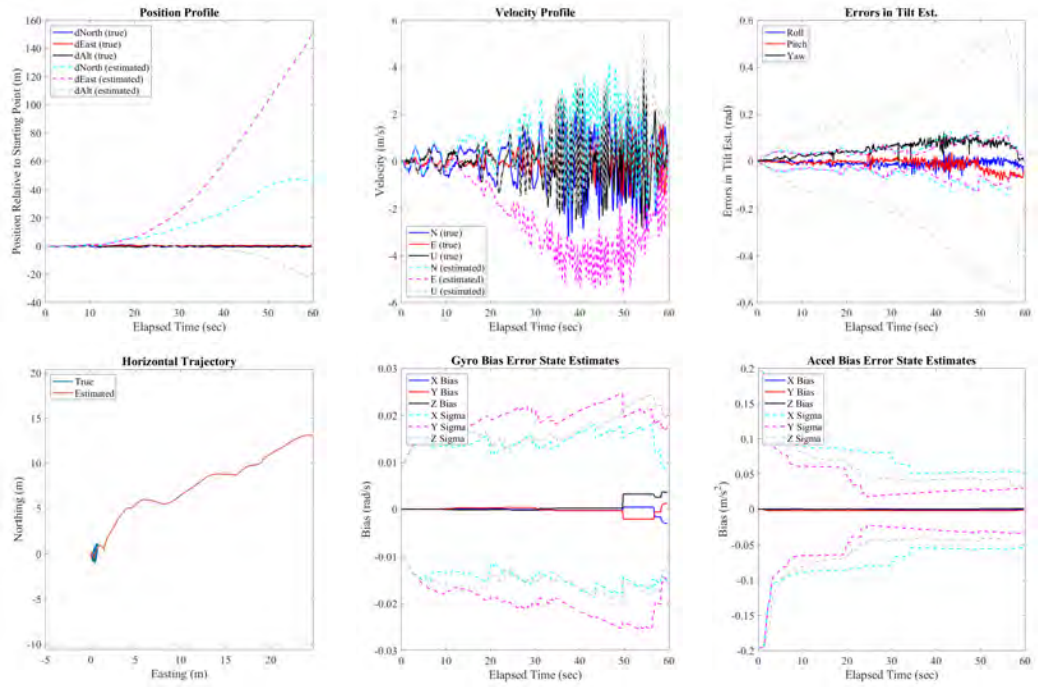

Figure 76: Boxes dataset MSCKF results with greyscale images, using $\sigma_{\texttt{im}} = 10$

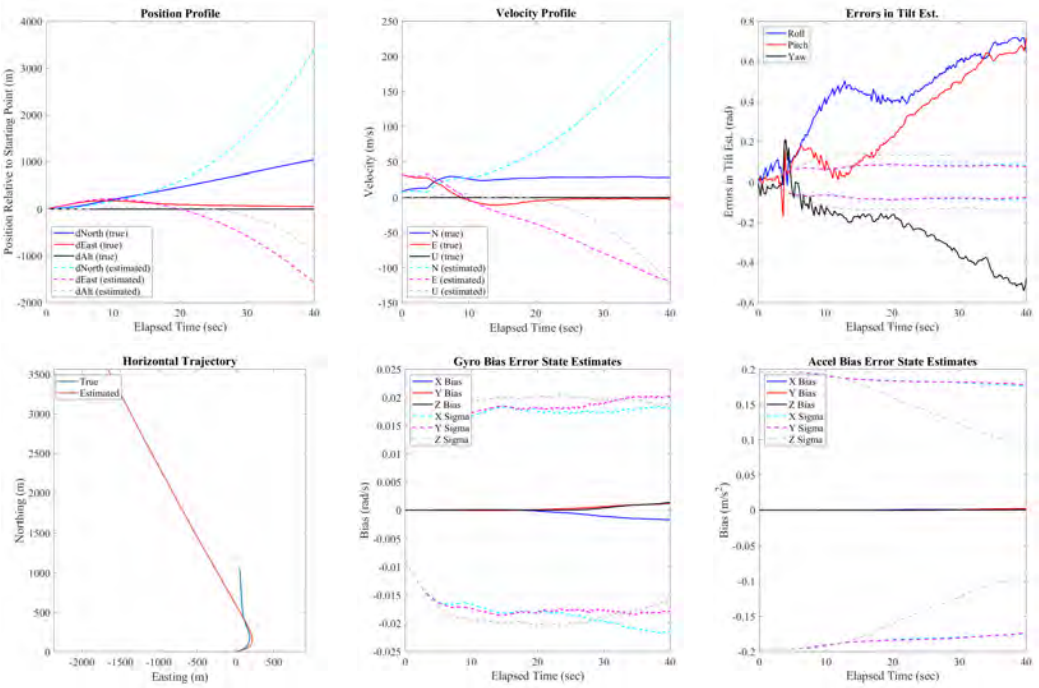

Figure 77: Camp Atterbury dataset MSCKF results with greyscale images, using $\sigma_{\texttt{im}} = 10$

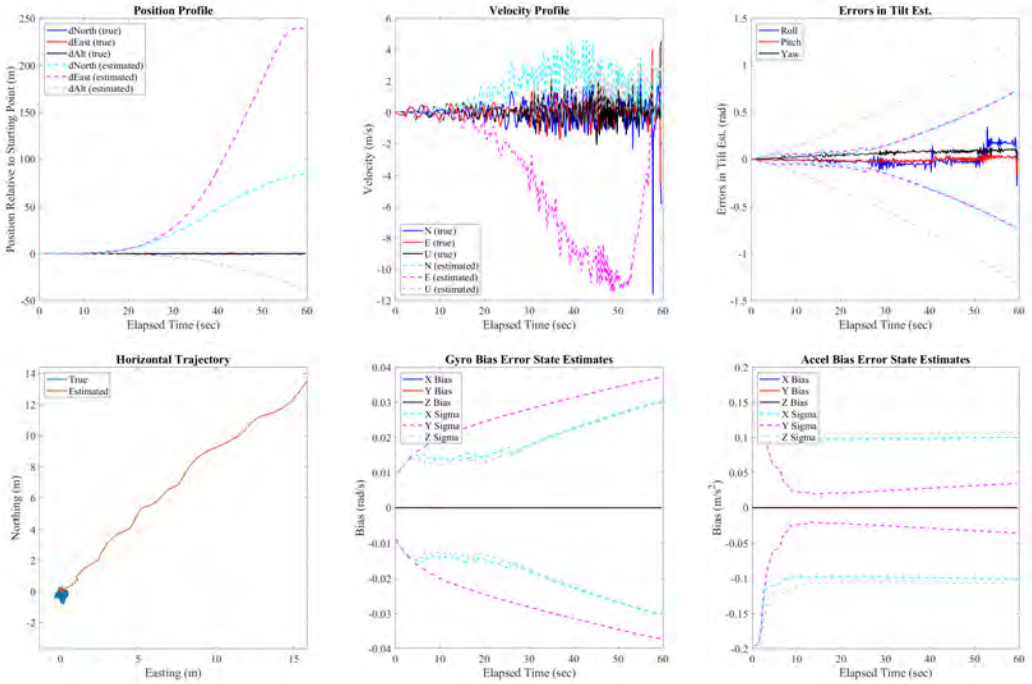

Figure 78: Shapes dataset MSCKF results with event frames, using $\sigma_{\texttt{im}} = 10$

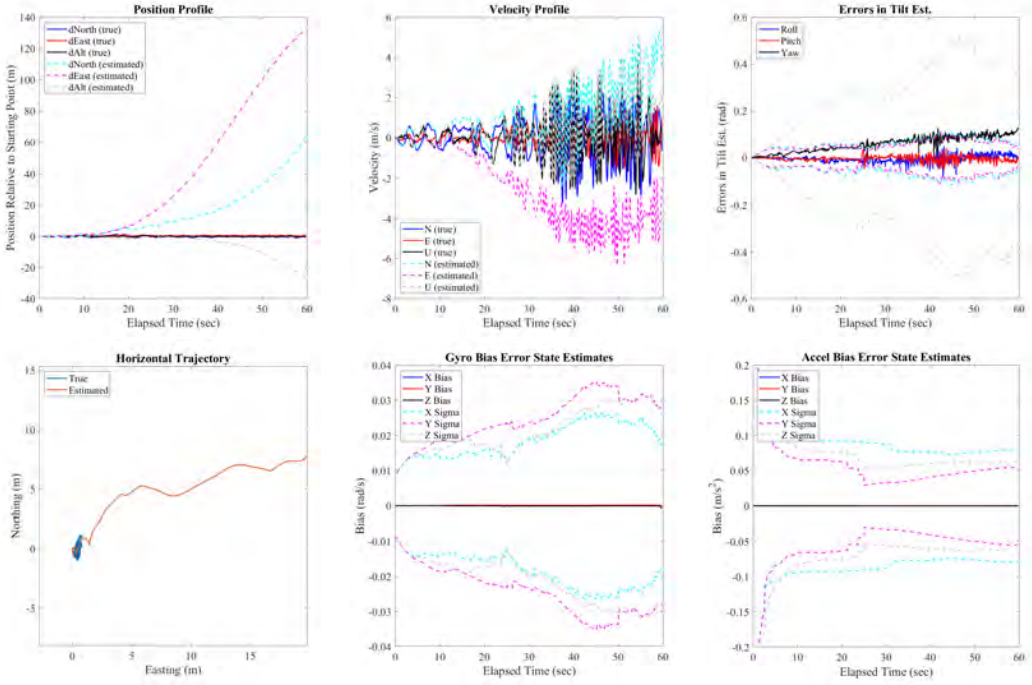

Figure 79: Boxes dataset MSCKF results with event frames, using $\sigma_{\texttt{im}} = 10$

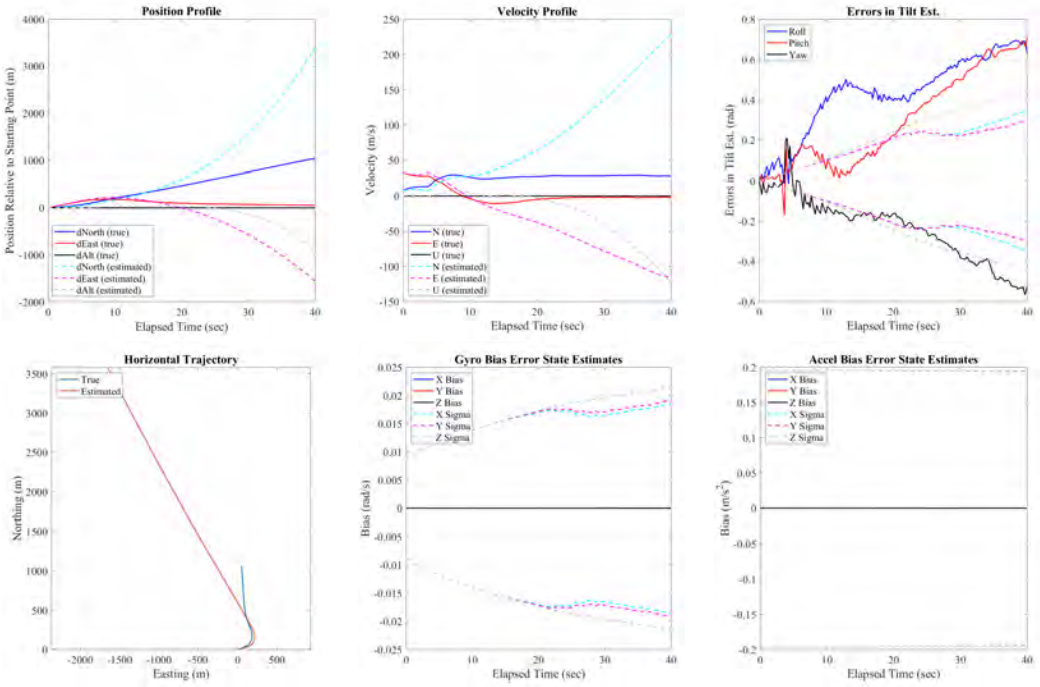

Figure 80: Camp Atterbury dataset MSCKF results with event frames, using $\sigma_{\texttt{im}} = 10$

Bibliography

1. Anastasios I. Mourikis and Stergios I. Roumeliotis. A multi-state constraint Kalman filter for vision-aided inertial navigation. In *Proceedings - IEEE International Conference on Robotics and Automation*, 2007.

2. Scorpion Documentation, 2018.

3. Elias Mueggler, Henri Rebecq, Guillermo Gallego, Tobi Delbruck, and Davide Scaramuzza. The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and SLAM. *International Journal of Robotics Research*, 36(2):142–149, 2017.

4. Alex Zihao Zhu, Nikolay Atanasov, and Kostas Daniilidis. Event-Based Visual Inertial Odometry. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017-Janua:5816–5824, 2017.

5. Alex Zihao Zhu, Nikolay Atanasov, and Kostas Daniilidis. Event-based feature tracking with probabilistic data association. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

6. Henri Rebecq, Timo Horstschaefer, and Davide Scaramuzza. Real-time Visual-Inertial Odometry for Event Cameras using Keyframe-based Nonlinear Optimization. *British Machine Vision Conference*, (September):1–8, 2017.

7. David Nister, Oleg Naroditsky, and James Bergen. Visual odometry. *Computer Vision and Pattern Recognition (CVPR), Proceedings of the 2004 IEEE Computer Society Conference on*, 2004.

8. Mohinder S. Grewal and Angus P. Andrews. Applications of Kalman Filtering in Aerospace 1960 to the Present. *IEEE Control Systems*, 2010.

9. Mark Maimone, Yang Cheng, and Larry Matthies. Two years of visual odometry on the Mars Exploration Rovers. *Journal of Field Robotics*, 2007.

10. Yang Cheng, Mark W. Maimone, and Larry Matthies. Visual odometry on the Mars Exploration Rovers: A tool ensure accurate driving and science imaging. *IEEE Robotics and Automation Magazine*, 2006.

11. Davide Scaramuzza and Friedrich Fraundorfer. Visual Odometry, Part I: The First 30 Years and Fundamentals. *IEEE Robotics and Automation Magazine*, 2011.

12. Friedrich Fraundorfer and Davide Scaramuzza. Visual Odometry, Part II: Matching, Robustness, Optimization, and Applications. *IEEE Robotics & Automation Magazine*, (June), 2012.

13. David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 2004.

14. Friedrich Fraundorfer and Davide Scaramuzza. Visual odometry Part 2: Matching, Robustness, Optimization, and Applications. *Robotics and Automation Magazine*, (June):78—-90, 2012.

15. Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding*, 2008.

16. Motilal Agrawal, Kurt Konolige, and Morten Rufus Blas. CenSurE: Center surround extremas for realtime feature detection and matching. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2008.

17. Pablo Fernández Alcantarilla, Adrien Bartoli, and Andrew J. Davison. KAZE features. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2012.

18. Alexandre Alahi, Raphael Ortiz, and Pierre Vandergheynst. FREAK: Fast retina keypoint. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2012.

19. Dg Viswanathan. Features from Accelerated Segment Test (FAST). *Homepages.Inf.Ed.Ac.Uk*, 2009.

20. C. Harris and M. Stephens. A Combined Corner and Edge Detector. In *Procedings of the Alvey Vision Conference 1988*, 1988.

21. Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. BRIEF: Binary robust independent elementary features. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2010.

22. Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: An efficient alternative to SIFT or SURF. In *Proceedings of the IEEE International Conference on Computer Vision*, 2011.

23. Stefan Leutenegger, Margarita Chli, and Roland Y. Siegwart. BRISK: Binary Robust invariant scalable keypoints. In *Proceedings of the IEEE International Conference on Computer Vision*, 2011.

24. Jakob Engel, Jurgen Sturm, and Daniel Cremers. Semi-dense visual odometry for a monocular camera. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1449–1456, Sydney, Australia, dec 2013.

25. Stephen Williams, Vadim Indelman, Michael Kaess, Richard Roberts, John J. Leanoard, and Frank Dellaert. Concurrent filtering and smoothing: A parallel architecture for real-time navigation and full smoothing. *International Journal of Robotics Research*, 33(12):1544–1568, 2014.

26. Jakob Engel, Thomas Sch, and Daniel Cremers. LSD-SLAM: Large-scale Direct Monocular SLAM. In *European Conference on Computer Vision*, pages 834–849, 2014.

27. Christian Forster, Matia Pizzoli, and Davide Scaramuzza. SVO: Fast semi-direct monocular visual odometry. *Proceedings - IEEE International Conference on Robotics and Automation*, (May):15–22, 2014.

28. A. Quattrini Li, A. Coskun, S. M. Doherty, and S. Ghasemlou. Experimental Comparison of open source Vision based State Estimation Algorithms. In *Int Symposium on Experimental Robotics*, number January, pages 775–786, 2016.

29. Davide Scaramuzza, Michael C Achtelik, Lefteris Doitsidis, Friedrich Fraundorfer, Elias Kosmatopoulos, Agostino Martinelli, Markus W Achtelik, Margarita Chli, Savvas Chatzichristofis, Laurent Kneip, Daniel Gurdan, Lionel Heng, Gim Hee Lee, Simon Lynen, Lorenz Meier, Marc Pollefeys, Alessandro Renzaglia, Roland Siegwart, Jan Carsten Stumpf, Petri Tanskanen, Chiara Troiani, and Stephan Weiss. Vision-Controlled Micro Flying Robots. *IEEE Robotics & Automation magazine*, (September):26–40, 2014.

30. Michael Burri, Janosch Nikolic, Pascal Gohl, Thomas Schneider, Joern Rehder, Sammy Omari, Markus W. Achtelik, and Roland Siegwart. The EuRoC micro aerial vehicle datasets. *International Journal of Robotics Research*, 35(10):1157–1163, 2016.

31. T Machin, John F Raquet, D Jacques, and D Venable. Real-time implementation of vision-aided navigation for small fixed-wing unmanned aerial systems. *29th International Technical Meeting of the Satellite Division of the Institute of Navigation, ION GNSS 2016*, 2:1305–1311, 2016.

32. Kishore Konda and Roland Memisevic. Learning Visual Odometry with a Convolutional Network. *International Conference on Computer Vision Theory and Applications*, pages 486–490, 2015.

33. Alex Kendall, Matthew Grimes, and Roberto Cipolla. PoseNet: A convolutional network for real-time 6-dof camera relocalization. *Proceedings of the IEEE International Conference on Computer Vision*, 2015 Inter:2938–2946, 2015.

34. Philipp Fischer, Alexey Dosovitskiy, Eddy Ilg, ..., and Thomas Brox. FlowNet: Learning Optical Flow with Convolutional Networks. In *IEEE International Conference on Computer Vision (ICCV)*, page 8, 2015.

35. Gabriele Costante, Michele Mancini, Paolo Valigi, and Thomas A. Ciarfuglia. Exploring Representation Learning With CNNs for Frame-to-Frame Ego-Motion Estimation. *IEEE Robotics and Automation Letters*, 1(1):18–25, 2016.

36. Sen Wang, Ronald Clark, Hongkai Wen, and Niki Trigoni. End-to-End, Sequence-to-Sequence Probabilistic Visual Odometry through Deep Neural Networks. *International Journal of Robotics Research*, 2016.

37. Sen Wang, Ronald Clark, Hongkai Wen, Niki Trigoni, Sen Wang, Ronald Clark, Niki Trigoni, Hongkai Wen, Niki Trigoni, Sen Wang, Ronald Clark, Niki Trigoni, Hongkai Wen, and Niki Trigoni. DeepVO: Towards End-to-End Visual Odometry with Deep Recurrent Convolutional Neural Networks. *Proceedings* -

IEEE International Conference on Robotics and Automation, pages 2043–2050, 2017.

38. Alec Graves, Steffen Lim, Thomas Fagan, and Kevin Mcfall. Visual Odometry using Convolutional Neural Networks. *The Kennesaw Journal of Undergraduate Research (KJUR)*, 5(3), 2017.

39. Ronald Clark, Sen Wang, Andrew Markham, Niki Trigoni, and Hongkai Wen. VidLoc: A Deep Spatio-Temporal Model for 6-DoF Video-Clip Relocalization. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

40. Ruihao Li, Sen Wang, Zhiqiang Long, and Dongbing Gu. UnDeepVO: Monocular Visual Odometry through Unsupervised Deep Learning. 2017.

41. Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1647–1655, 2017.

42. Ronald Clark, Sen Wang, Hongkai Wen, Andrew Markham, and Niki Trigoni. VINet: Visual-Inertial Odometry as a Sequence-to-Sequence Learning Problem. *Association for the Advancement of Artificial Intelligence (www.aaai.org)*, 2017.

43. Saeed Reza Kheradpisheh, Mohammad Ganjtabesh, Simon J. Thorpe, and Timothée Masquelier. STDP-based spiking deep convolutional neural networks for object recognition. *Neural Networks*, 99, 2018.

44. Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. On-Manifold Preintegration for Real-Time Visual-Inertial Odometry. *IEEE Transactions on Robotics*, 33(1), 2017.

45. Stefan Leutenegger, Paul Furgale, Vincent Rabaud, Margarita Chli, Kurt Konolige, and Roland Siegwart. Keyframe-Based Visual-Inertial SLAM Using Nonlinear Optimization. In *Proceedings of Robotics: Science and Systems*, page 0, 2013.

46. Stefan Leutenegger, Simon Lynen, Michael Bosse, Roland Siegwart, and Paul Furgale. Keyframe-based visual-inertial odometry using nonlinear optimization. *International Journal of Robotics Research*, 34(3):314–334, 2015.

47. Vladyslav Usenko, Jakob Engel, Jörg Stuckler, Daniel Cremers, Jörg Stückler, Daniel Cremers, Jörg Stuckler, and Daniel Cremers. Direct Visual-Inertial Odometry with Stereo Cameras. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 2016-June, 2016.

48. Shih-Chii Chii Liu, Tobi Delbruck, Giacomo Indiveri, Adrian Whatley, Rodney Douglas, and Adrian Whatley. *Event-based neuromorphic systems*. John Wiley \& Sons, 2015.

49. Tobi Delbruck. Frame-free dynamic digital vision. *Proceedings of Intl. Symp. on Secure-Life Electronics, Advanced Electronics for Quality Life and Society*, pages 21–26, 2008.

50. Georg Rupert Müller. *Event Based Vision for Autonmous High-Speed Robots*. PhD thesis, 2013.

51. Zhenjiang Ni. *Asynchronous Event Based Vision : Algorithms and Applications to Microrobotics*. PhD thesis, {Universit{\'e} Pierre et Marie Curie-Paris VI}, 2014.

52. Christian Peter Brändli. *Event-Based Machine Vision*. PhD thesis, 2015.

53. Patrick Lichtsteiner, Christoph Posch, and Tobi Delbruck. A 128 x 128 120dB 15us Latency Asynchronous Temporal Contrast Vision Sensor. *IEEE Journal of Solid-State Circuits*, 43(2):566–576, 2008.

54. Gregory Cohen, Saeed Afshar, André Van Schaik, Andrew Wabnitz, Travis Bessell, Mark Rutten, and Brittany Morreale. Event-based Sensing for Space Situational Awareness. *Advanced Maui Optical and Space Surveillance Technologies Conference (AMOS) – www.amostech.com*, 2017.

55. Davide Scaramuzza. Tutorial on Event-based Vision for High-Speed Robotics. In *IROS 2015: Proc.\ of the 2nd Workshop on Alternative Sensing for Robot Perception*, pages 1–78, 2015.

56. Christian Brandli, Raphael Berner, Minhao Yang, Shih Chii Liu, and Tobi Delbruck. A 240 x 180 130 dB 3 μs latency global shutter spatiotemporal vision sensor. *IEEE Journal of Solid-State Circuits*, 2014.

57. Christoph Posch, Daniel Matolin, and Rainer Wohlgenannt. An asynchronous time-based image sensor. In *Circuits and Systems, 2008. ISCAS 2008. IEEE International Symposium on*, pages 2130–2133. IEEE, 2008.

58. Christoph Posch, Daniel Matolin, Rainer Wohlgenannt, Michael Hofstatter, Peter Schon, Martin Litzenberger, Daniel Bauer, Heinrich Garn, Michael Hofstätter, Peter Schön, Martin Litzenberger, Daniel Bauer, and Heinrich Garn. Live Demonstration: Asynchronous Time-based Image Sensor (ATIS) Camera with Full-Custom AE Processor. In *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, pages 1392–1392. IEEE, may 2010.

59. Christian Brandli, Lorenz Muller, and Tobi Delbruck. Real-time, high-speed video decompression using a frame- and event-based DAVIS sensor. *Proceedings - IEEE International Symposium on Circuits and Systems*, pages 686–689, 2014.

60. Christoph Posch, Teresa Serrano-Gotarredona, Bernabe Linares-Barranco, and Tobi Delbruck. Retinomorphic event-based vision sensors: Bioinspired cameras with spiking output. *Proceedings of the IEEE*, 102(10):1470–1484, 2014.

61. Anup Vanarse, Adam Osseiran, and Alexander Rassau. A review of current neuromorphic approaches for vision, auditory, and olfactory sensors, 2016.

62. Viviane S. Ghaderi, Marcello Mulas, Vinicius Felisberto Santos Pereira, Lukas Everding, David Weikersdorfer, and Jorg Conradt. A wearable mobility device for the blind using retina-inspired dynamic vision sensors. *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, 2015-Novem:3371–3374, 2015.

63. Lukas Everding, Lennart Walger, Viviane S. Ghaderi, and Jörg Conradt. A mobility device for the blind with improved vertical resolution using dynamic vision sensors. *2016 IEEE 18th International Conference on e-Health Networking, Applications and Services, Healthcom 2016*, pages 0–4, 2016.

64. Christian Reinbacher, Gottfried Graber, and Thomas Pock. Real-Time Intensity-Image Reconstruction for Event Cameras Using Manifold Regularisation. In *2016 British Machine Vision Conference (BMVC)*, 2016.

65. Hanme Kim, Ankur Handa, Ryad Benosman, Sio-Hoi Ieng, and Andrew Davison. Simultaneous Mosaicing and Tracking with an Event Camera. In *Proceedings of the British Machine Vision Conference 2014*, 2014.

66. Henri Rebecq, Guillermo Gallego, Elias Mueggler, and Davide Scaramuzza. EMVS: Event-Based Multi-View Stereo—3D Reconstruction with an Event Camera in Real-Time. *International Journal of Computer Vision*, pages 1–21, 2017.

67. Guillermo Gallego, Christian Forster, Elias Mueggler, and Davide Scaramuzza. Event-based Camera Pose Tracking using a Generative Event Model. *Arxiv*, 1:1–7, 2015.

68. Patrick Bardow, Andrew J. Davison, and Stefan Leutenegger. Simultaneous Optical Flow and Intensity Estimation from an Event Camera. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 884–892, 2016.

69. Han Chao Liu, Fang Lue Zhang, David Marshall, Luping Shi, and Shi Min Hu. High-speed video generation with an event camera. *Visual Computer*, 33(6-8):749–759, 2017.

70. Guillermo Gallego, Henri Rebecq, and Davide Scaramuzza. A Unifying Contrast Maximization Framework for Event Cameras, with Applications to Motion, Depth, and Optical Flow Estimation. 2018.

71. T. Delbruck, V. Villanueva, and L. Longinotti. Integration of dynamic vision sensor with inertial measurement unit for electronically stabilized event-based vision. *Proceedings - IEEE International Symposium on Circuits and Systems*, pages 2636–2639, 2014.

72. Elias Mueggler, Christian Forster, Nathan Baumli, Guillermo Gallego, and Davide Scaramuzza. Lifetime estimation of events from Dynamic Vision Sensors.

Proceedings - IEEE International Conference on Robotics and Automation, 2015-June(June):4874–4881, 2015.

73. Xavier Lagorce, Garrick Orchard, Francesco Galluppi, Bertram E. Shi, and Ryad B. Benosman. HOTS: A Hierarchy of Event-Based Time-Surfaces for Pattern Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(7), 2017.

74. L. A. Camunas-Mesa, T. Serrano-Gotarredona, B. Linares-Barranco, S. Ieng, and R. Benosman. Event-driven stereo vision with orientation filters. In *Proceedings - IEEE International Symposium on Circuits and Systems*, 2014.

75. Luis A. Camuñas-Mesa, Teresa Serrano-Gotarredona, Sio H. Ieng, Ryad B. Benosman, and Bernabe Linares-Barranco. On the use of orientation filters for 3D reconstruction in event-driven stereo vision. *Frontiers in Neuroscience*, (8 MAR), 2014.

76. Henri Rebecq, Timo Horstschaefer, and Davide Scaramuzza. Real-time Visual-Inertial Odometry for Event Cameras using Keyframe-based Nonlinear Optimization. *British Machine Vision Conference*, (September):1–8, 2017.

77. E. Piatkowska, A. N. Belbachir, and M. Gelautz. Cooperative and asynchronous stereo vision for dynamic vision sensors. *Measurement Science and Technology*, 25(5), 2014.

78. Ewa Piatkowska, Jurgen Kogler, Nabil Belbachir, and Margrit Gelautz. Improved Cooperative Stereo Matching for Dynamic Vision Sensors with Ground Truth Evaluation. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2017-July:370–377, 2017.

79. Mohsen Firouzi and Jörg Conradt. Asynchronous Event-based Cooperative Stereo Matching Using Neuromorphic Silicon Retinas. *Neural Processing Letters*, 43(2):311–326, 2016.

80. Paul Rogister, Ryad Benosman, Sio Hoi Ieng, Patrick Lichtsteiner, and Tobi Delbruck. Asynchronous event-based binocular stereo matching. *IEEE Transactions on Neural Networks and Learning Systems*, 23(2):347–353, 2012.

81. Hongjie Liu, Christian Brandli, Chenghan Li, Shih Chii Liu, and Tobi Delbruck. Design of a spatiotemporal correlation filter for event-based sensors. *Proceedings - IEEE International Symposium on Circuits and Systems*, 2015-July:722–725, 2015.

82. Alireza Khodamoradi and Ryan Kastner. O(N)-Space Spatiotemporal Filter for Reducing Noise in Neuromorphic Vision Sensors. *IEEE Transactions on Emerging Topics in Computing*, 48(5):1172–1183, 2018.

83. A. Linares-Barranco, F. Gomez-Rodriguez, V. Villanueva, L. Longinotti, and T. Delbruck. A USB3.0 FPGA event-based filtering and tracking framework for dynamic vision sensors. *Proceedings - IEEE International Symposium on Circuits and Systems*, 2015-July:2417–2420, 2015.

84. Yuji Nozaki and Tobi Delbruck. Temperature and Parasitic Photocurrent Effects in Dynamic Vision Sensors. *IEEE Transactions on Electron Devices*, 64(8):3239–3245, 2017.

85. Quentin Sabatier, Sio Hoi Ieng, and Ryad Benosman. Asynchronous Event-Based Fourier Analysis. *IEEE Transactions on Image Processing*, 26(5):2192–2202, 2017.

86. Ryad Benosman, Sio Hoï Ieng, Paul Rogister, and Christoph Posch. Asynchronous event-based Hebbian epipolar geometry. *IEEE Transactions on Neural Networks*, 22(11):1723–1734, 2011.

87. Gregory Cohen, Saeed Afshar, Garrick Orchard, Jonathan Tapson, Ryad Benosman, and Andre van Schaik. Spatial and Temporal Downsampling in Event-Based Visual Classification. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–15, 2018.

88. Gregory Kevin Cohen. *Event-Based Feature Detection, Recognition and Classification*. PhD thesis, Université Pierre et Marie Curie - Paris VI, 2016.

89. Christian Brandli, Jonas Strubel, Susanne Keller, Davide Scaramuzza, and Tobi Delbruck. ELiSeD-An event-based line segment detector. *2016 2nd International Conference on Event-Based Control, Communication, and Signal Processing, EBCCSP 2016 - Proceedings*, 2016.

90. Xavier Clady, Sio Hoi Ieng, and Ryad Benosman. Asynchronous event-based corner detection and matching. *Neural Networks*, 66:91–106, 2015.

91. Valentina Vasco, Arren Glover, and Chiara Bartolozzi. Fast event-based Harris corner detection exploiting the advantages of event-driven cameras. In *IEEE International Conference on Intelligent Robots and Systems*, volume 2016-Novem, pages 4144–4149, 2016.

92. Elias Mueggler, Chiara Bartolozzi, and Davide Scaramuzza. Fast Event-based Corner Detection. *British Machine Vision Conference*, 1:1–11, 2017.

93. Xavier Lagorce, Sio Hoi Ieng, and Ryad Benosman. Event-based features for robotic vision. In *IEEE International Conference on Intelligent Robots and Systems*, pages 4214–4219, 2013.

94. Xavier Lagorce, Sio Hoi Ieng, Xavier Clady, Michael Pfeiffer, and Ryad B. Benosman. Spatiotemporal features for asynchronous event-based data. *Frontiers in Neuroscience*, 9(FEB), 2015.

95. Min Liu and Tobi Delbruck. Block-matching optical flow for dynamic vision sensors: Algorithm and FPGA implementation. *Proceedings - IEEE International Symposium on Circuits and Systems*, pages 0–3, 2017.

96. Alex Junho A.J. Lee and Ayoung Kim. Event-based real-time optical flow estimation. In *International Conference on Control, Automation and Systems*, volume 2017-Octob, pages 787–791, 2017.

97. Bodo Rueckauer and Tobi Delbruck. Evaluation of event-based algorithms for optical flow with ground-truth from inertial measurement sensor. *Frontiers in Neuroscience*, 10(APR):1–17, 2016.

98. Tobias Brosch, Stephan Tschechne, and Heiko Neumann. On event-based optical flow detection. *Frontiers in Neuroscience*, 9(APR), 2015.

99. J. Conradt. On-board real-time optic-flow for miniature event-based vision sensors. In *2015 IEEE International Conference on Robotics and Biomimetics, IEEE-ROBIO 2015*, 2015.

100. Ryad Benosman, Charles Clercq, Xavier Lagorce, Sio Hoi Ieng, and Chiara Bartolozzi. Event-based visual flow. *IEEE Transactions on Neural Networks and Learning Systems*, 25(2):407–417, 2014.

101. Ryad Benosman, Sio Hoi Ieng, Charles Clercq, Chiara Bartolozzi, and Mandyam Srinivasan. Asynchronous frameless event-based optical flow. *Neural Networks*, 27:32–37, 2012.

102. Xavier Clady, Jean Matthieu Maro, Sébastien Barré, and Ryad B. Benosman. A motion-based feature for event-based pattern recognition. *Frontiers in Neuroscience*, 10(JAN), 2017.

103. David Tedaldi, Guillermo Gallego, Elias Mueggler, and Davide Scaramuzza. Feature Detection and Tracking with the Dynamic and Active-pixel Vision Sensor (DAVIS). *International Conference on Event-based Control, Communication, and Signal Processing*, 2016.

104. Francisco Barranco, Cornelia Fermuller, and Yiannis Aloimonos. Contour motion estimation for asynchronous event-driven cameras. *Proceedings of the IEEE*, 102(10), 2014.

105. Xavier Clady, Charles Clercq, Sio Hoi Ieng, Fouzhan Houseini, Marco Randazzo, Lorenzo Natale, Chiara Bartolozzi, and Ryad Benosman. Asynchronous visual event-based time-to-contact. *Frontiers in Neuroscience*, (8 FEB), 2014.

106. Lionel Fillatre. Bayes classification for asynchronous event-based cameras. In *2015 23rd European Signal Processing Conference, EUSIPCO 2015*, pages 824–828, 2015.

107. David Reverter Valeiras, Sihem Kime, Sio Hoi Ieng, and Ryad Benjamin Benosman. An event-based solution to the Perspective-n-Point problem. *Frontiers in Neuroscience*, 10(MAY), 2016.

108. Garrick Orchard, Cedric Meyer, Ralph Etienne-Cummings, Christoph Posch, Nitish Thakor, and Ryad Benosman. HFirst: A Temporal Approach to Object Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(10), 2015.

109. S. Hoseini, G. Orchard, A. Yousefzadeh, B. Deverakonda, T. Serrano-Gotarredona, and B. Linares-Barranco. Passive localization and detection of quadcopter UAVs by using dynamic vision sensor. *5th Iranian Joint Congress on Fuzzy and Intelligent Systems - 16th Conference on Fuzzy Systems and 14th Conference on Intelligent Systems, CFIS 2017*, pages 81–85, 2017.

110. Elias Mueggler, Nathan Baumli, Flavio Fontana, and Davide Scaramuzza. Towards evasive maneuvers with quadrotors using dynamic vision sensors. *2015 European Conference on Mobile Robots, ECMR 2015 - Proceedings*, 2015.

111. Arren Glover and Chiara Bartolozzi. Robust visual tracking with a freely-moving event camera. In *IEEE International Conference on Intelligent Robots and Systems*, volume 2017-Septe, pages 3769–3776, 2017.

112. Chiara Bartolozzi, Francesco Rea, Charles Clercq, Daniel B. Fasnacht, Giacomo Indiveri, Michael Hofstätter, and Giorgio Metta. Embedded neuromorphic vision for humanoid robots. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2011.

113. Chiara Bartolozzi. Event-Driven Sensing for a Humanoid Robot. In *IEEE International Conference on Robotics and Automation*, 2017.

114. David Reverter Valeiras, Garrick Orchard, Sio Hoi Ieng, and Ryad B. Benosman. Neuromorphic event-based 3D pose estimation. *Frontiers in Neuroscience*, 9(JAN), 2016.

115. Hongjie Liu, Diederik Paul Moeys, Gautham Das, Daniel Neil, Shih Chii Liu, and Tobi Delbruck. Combined frame- and event-based detection and tracking. In *Proceedings - IEEE International Symposium on Circuits and Systems*, volume 2016-July, 2016.

116. Xi Peng, Bo Zhao, Rui Yan, Huajin Tang, and Zhang Yi. Bag of events: An efficient probability-based feature extraction method for AER image sensors. *IEEE Transactions on Neural Networks and Learning Systems*, 28(4), 2017.

117. Evangelos Stromatias, Miguel Soto, Teresa Serrano-Gotarredona, and Bernabé Linares-Barranco. An event-driven classifier for spiking neural networks fed with synthetic or dynamic vision sensor data. *Frontiers in Neuroscience*, 11(JUN), 2017.

118. Jun Haeng Lee, Tobi Delbruck, Michael Pfeiffer, Paul K.J. J Park, Chang-woo Woo Shin, Hyunsurk Eric Ryu, Byung Chang Kang, and Senior Member. Real-Time Gesture Interface Based on Event-Driven Processing from Stereo Silicon Retinas. *IEEE Transactions on Neural Networks and Learning Systems*, 25(12):2250–2263, 2014.

119. Junhaeng Lee, T Delbruck, Paul K J Park, Michael Pfeiffer, Chang-woo Shin, Hyunsurk Ryu, Byung Chang Kang, Junhaeng Lee, T Delbruck, Paul K J Park, Michael Pfeiffer, Chang-woo Shin, Hyunsurk Ryu, and Byung Chang Kang. Live Demonstration : Gesture-Based remote control using stereo pair of dynamic vision sensors Gesture-Based remote control using stereo pair of dynamic vision sensors. In *Circuits and Systems (ISCAS), 2012 IEEE International Symposium on*, pages 741–745, 2012.

120. Arnon Amir, Brian Taba, David Berg, Timothy Melano, Jeffrey McKinstry, Carmelo Di Nolfo, Tapan Nayak, Alexander Andreopoulos, Guillaume Garreau, Marcela Mendoza, Jeff Kusnitz, Michael Debole, Steve Esser, Tobi Delbruck, Myron Flickner, Dharmendra Modha, Carmelo Di Nolfo, Tapan Nayak, Alexander Andreopoulos, Guillaume Garreau, Marcela Mendoza, Jeff Kusnitz, Michael Debole, Steve Esser, Tobi Delbruck, Myron Flickner, and Dharmendra

Modha. A low power, fully event-based gesture recognition system. In *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, volume 2017-Janua, pages 7388–7397, 2017.

121. Z. Ni, C. Pacoret, R. Benosman, S. Ieng, and S. Régnier. Asynchronous event-based high speed vision for microparticle tracking. *Journal of Microscopy*, 245(3):236–244, 2012.

122. Tat-Jun Chin, Samya Bagchi, Anders Eriksson, and André van Schaik. Star Tracking for Spacecraft Attitude Estimation using an Event Camera. *Cvpr*, (1), 2019.

123. T. Delbruck, M. Pfeiffer, R. Juston, G. Orchard, E. Muggler, A. Linares-Barranco, and M. W. Tilden. Human vs. computer slot car racing using an event and frame-based DAVIS vision sensor. In *Proceedings - IEEE International Symposium on Circuits and Systems*, volume 2015-July, 2015.

124. Guillermo Gallego and Davide Scaramuzza. Accurate Angular Velocity Estimation With an Event Camera. *IEEE Robotics and Automation Letters*, 2(2):632–639, 2017.

125. Christian Reinbacher, Gottfried Munda, and Thomas Pock. Real-time panoramic tracking for event cameras. In *2017 IEEE International Conference on Computational Photography, ICCP 2017 - Proceedings*, 2017.

126. David Weikersdorfer and Jorg Conradt. Event-based particle filtering for robot self-localization. In *2012 IEEE International Conference on Robotics and Biomimetics, ROBIO 2012 - Conference Digest*, pages 866–870, 2012.

127. Michael Milford, Hanme Kim, Stefan Leutenegger, and Andrew Davison. Towards Visual SLAM with Event-based Cameras. *RSS Workshop*, (Figure 2):1–8, 2015.

128. Guillermo Gallego, Jon E.A. Lund, Elias Mueggler, Henri Rebecq, Tobi Delbruck, and Davide Scaramuzza. Event-based, 6-DOF Camera Tracking from Photometric Depth Maps. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

129. David Weikersdorfer, David B. Adrian, Daniel Cremers, and Jorg Conradt. Event-based 3D SLAM with a depth-augmented dynamic vision sensor. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 359–364, 2014.

130. Henri Rebecq, Timo Horstschaefer, Guillermo Gallego, and Davide Scaramuzza. EVO: A Geometric Approach to Event-Based 6-DOF Parallel Tracking and Mapping in Real Time. *IEEE Robotics and Automation Letters*, 2(2):593–600, 2017.

131. Elias Mueggler, Guillermo Gallego, and Davide Scaramuzza. Continuous-Time Trajectory Estimation for Event-based Vision Sensors. In *Robotics: Science and Systems XI*, 2015.

132. Guillermo Gallego, Jon E A Lund, Elias Mueggler, Henri Rebecq, Tobi Delbruck, and Davide Scaramuzza. Event-based, 6-DOF Camera Tracking for High-Speed Applications. *arXiv preprint arXiv:1607.03468*, pages 1–8, 2016.

133. Elias Mueggler, Basil Huber, and Davide Scaramuzza. Event-based, 6-DOF pose tracking for high-speed maneuvers. *IEEE International Conference on Intelligent Robots and Systems*, pages 2761–2768, 2014.

134. Hanme Kim, Stefan Leutenegger, and Andrew J. Davison. Real-time 3D reconstruction and 6-DoF tracking with an event camera. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 9910 LNCS, 2016.

135. Beat Kueng, Elias Mueggler, Guillermo Gallego, and Davide Scaramuzza. Low-latency visual odometry using event-based feature tracks. *IEEE International Conference on Intelligent Robots and Systems*, 2016-Novem:16–23, 2016.

136. Andrea Censi and Davide Scaramuzza. Low-latency event-based visual odometry. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 703–710, 2014.

137. Antoni Rosinol Vidal, Henri Rebecq, Timo Horstschaefer, and Davide Scaramuzza. Ultimate SLAM? Combining Events, Images, and IMU for Robust Visual SLAM in HDR and High Speed Scenarios. *IEEE Robotics and Automation Letters*, 3(2):994–1001, 2018.

138. Alex Zihao Zhu, Dinesh Thakur, Tolga Ozaslan, Bernd Pfrommer, Vijay Kumar, and Kostas Daniilidis. The Multi Vehicle Stereo Event Camera Dataset : An Event Camera Dataset for 3D Perception. *IEEE Robotics and Automation Letters*, 19104(c):1–8, 2018.

139. Teresa Serrano-Gotarredona and Bernabé Linares-Barranco. Poker-DVS and MNIST-DVS. Their history, how they were made, and other details. *Frontiers in Neuroscience*, 9(DEC), 2015.

140. Iulia Alexandra Lungu, Federico Corradi, and Tobi Delbruck. Live demonstration: Convolutional neural network driven by dynamic vision sensor

playing RoShamBo. In *Proceedings - IEEE International Symposium on Circuits and Systems*, 2017.

141. Garibaldi Pineda Garcia, Patrick Camilleri, Qian Liu, and Steve Furber. PyDVS: An extensible, real-time Dynamic Vision Sensor emulator using off-the-shelf hardware. *2016 IEEE Symposium Series on Computational Intelligence, SSCI 2016*, 2017.

142. Garrick Orchard, Ajinkya Jayawant, Gregory K. Cohen, and Nitish Thakor. Converting static image datasets to spiking neuromorphic datasets using saccades. *Frontiers in Neuroscience*, 9(NOV), 2015.

143. WG Breckenridge. Quaternions proposed standard conventions. *Jet Propulsion Laboratory (JPL), Interoffice Memorandum (IOM)*, pages 343—-79, 1999.

144. D. Titterton and J. Weston. *Strapdown inertial navigation technology*. Institute of Engineering and Technology (IET), 2nd edition, 2004.

145. Dm Henderson. Euler Angles, Quaternions, and Transformation Matrices. *NASA JSC Report*, 1977.

146. J. M. M. Montiel, Javier Civera, and Andrew J. Davison. Unified Inverse Depth Parametrization for Monocular SLAM. *Proceedings of Robotics Science & Systems*, 2006.

147. Bill Triggs, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon. Bundle Adjustment — A Modern Synthesis Vision Algorithms: Theory and Practice. *Vision Algorithms: Theory and Practice*, 2000.

148. Thomas Debrunner. AedatTools, 2018.

149. Joern Rehder, Janosch Nikolic, Thomas Schneider, Timo Hinzmann, and Roland Siegwart. Extending kalibr: Calibrating the extrinsics of multiple IMUs and of individual axes. In *Proceedings - IEEE International Conference on Robotics and Automation*, 2016.

150. Paul Furgale, Joern Rehder, and Roland Siegwart. Unified temporal and spatial calibration for multi-sensor systems. In *IEEE International Conference on Intelligent Robots and Systems*, 2013.

151. Paul Furgale, Timothy D. Barfoot, and Gabe Sibley. Continuous-time batch estimation using temporal basis functions. In *Proceedings - IEEE International Conference on Robotics and Automation*, 2012.

152. Jérôme Maye, Paul Furgale, and Roland Siegwart. Self-supervised calibration for robotic systems. In *IEEE Intelligent Vehicles Symposium, Proceedings*, 2013.

153. Luc Oth, Paul Furgale, Laurent Kneip, and Roland Siegwart. Rolling shutter camera calibration. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2013.

154. Chiara Troiani, Agostino Martinelli, Christian Laugier, and Davide Scaramuzza. 2-Point-based outlier rejection for camera-IMU systems with applications To micro aerial vehicles. In *Proceedings - IEEE International Conference on Robotics and Automation*, 2014.

155. Edward J. Bedrick, Jodi Lapidus, and Joseph F. Powell. Estimating the Mahalanobis distance from mixed continuous and discrete data. *Biometrics*, 2000.

Acronyms

AFIT Air Force Institute of Technology. iv, 1, 3, 23, 27, 83

AFRL Air Force Research Laboratory. 1

AMD age-related macular degeneration. 17

ANT Autonomy and Navigation Technology. iv, 1, 3, 4, 22, 23, 27, 43, 83

APS active pixel sensor. 16, 18

ATIS Asynchronous Time-based Image Sensor. 17

BA background activity. 18

CCD charge-coupled device. 17, 20

CMOS complementary metal-oxide semiconductor. 13

DAVIS dynamic and active-pixel vision sensor. 15, 16, 17, 20, 22, 43, 44

DCM direct cosine matrix. 8, 22, 30

DOF degrees-of-freedom. iv, 3, 5, 11, 16, 20

DVS dynamic vision sensor. vii, 12, 13, 14, 15, 16, 17, 18, 21, 43

EKF extended Kalman Filter. 3, 29, 32, 37, 39, 41

ETH Swiss Federal Institute of Technology. 4, 15

EVIO event-based visual-inertial odometry. iv, 3, 4, 20, 22, 27, 46, 52, 83, 85

EVO Event-based Visual Odometry. 20

GNSS Global Navigation Satellite System. 50, 83

GPS Global Positioning System. 1, 5, 11, 20, 24

GRASP General Robotics, Automation, Sensing and Perception. 20

HDR high-dynamic range. 12

IMU inertial measurement unit. iv, 3, 5, 11, 16, 20, 28, 29, 30, 31, 32, 33, 43, 44, 63, 66, 67, 71, 83

INI Institute of Neuroinformatics. 4, 15, 20

INS inertial navigation system. 26, 27, 28, 29, 32, 33, 69, 71, 73, 76, 77, 83, 84

JPL Jet Propulsion Laboratory. 23

LCM Lightweight Communications and Marshalling. 43, 44

LLA latitude-longitude-altitude. 33

LSE least-squares estimate. 37, 38, 40, 76

MEMS Micro-Electro-Mechanical Systems. iv, 3, 70, 83

MSCKF Multi-State Constrained Kalman Filter. iv, viii, ix, x, xi, 3, 4, 22, 27, 32, 38, 46, 52, 59, 60, 62, 71, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105

NED north-east-down. 28, 32, 34

QA quality assurance. 17

RANSAC random sample consensus. 65, 84

ROS Robot Operating System. 43, 85

RPG Robotics and Perception Group. 4, 20, 22, 42, 44, 46, 48, 50

SSA space situational awareness. 18

UAV unmanned aerial vehicle. iv, 3, 22, 43, 49, 83, 84

USB Universal Serial Bus. 15

UZH University of Zürich. 4, 15, 42

VIO visual-inertial odometry. 11

VO visual odometry. iv, 1, 2, 4, 5, 11, 12, 24, 42, 51, 64, 85

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
21–03–2019	Master's Thesis	Sept 2017 — Mar 2019

4. TITLE AND SUBTITLE

Event-Based Visual-Inertial Odometry
on a Fixed-Wing Unmanned Aerial Vehicle

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Kaleb J. Nelson

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT-ENG-MS-19-M-048

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

AFRL/RYWN
Building 620
WPAFB OH 45433-7765
DSN 713-4418, COMM 937-713-4418
Email: jeffrey.hebert@us.af.mil

10. SPONSOR/MONITOR'S ACRONYM(S)

AFRL/RYWN

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A:
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Event-based cameras are a new type of visual sensor that operate under a unique paradigm. These cameras provide asynchronous data on the log-level changes in light intensity for individual pixels, independent of other pixels' measurements. Through the hardware-level approach to change detection, these cameras can achieve microsecond fidelity, millisecond latency, ultra-wide dynamic range, and all with very low power requirements. The advantages provided by event-based cameras make them excellent candidates for visual odometry for UAVs. A visual odometry pipeline was implemented with a front-end algorithm for generating motion-compensated event-frames feeding a Multi-State Constraint Kalman Filter (MSCKF) back-end implemented using Scorpion. This pipeline was tested on a public dataset and data collected from an ANT Center UAV flight test.

15. SUBJECT TERMS

event-based cameras, event-based visual-inertial odometry (EVIO), neuromorphic engineering, visual odometry (VO), visual-inertial odometry (VIO), multi-state constraint Kalman Filter (MSCKF), Extended Kalman Filter (EKF)

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Captain Kaleb J. Nelson, AFIT/ENG
U	U	U	UU	144	19b. TELEPHONE NUMBER *(include area code)* (937) 255-3636, x4556; kaleb.nelson@afit.edu