UTILIZATION OF EVENT BASED CAMERAS FOR VIDEO FRAME INTERPOLATION

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ONUR SELIM KILIÇ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

AUGUST 2022

Approval of the thesis:

UTILIZATION OF EVENT BASED CAMERAS FOR VIDEO FRAME INTERPOLATION

submitted by **ONUR SELIM KILIÇ** in partial fulfillment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. İlkay Ulusoy
Head of Department, **Electrical and Electronics Engineering** _____

Prof. Dr. A. Aydın Alatan
Supervisor, **Electrical and Electronics Engineering, METU** _____

Examining Committee Members:

Assoc. Prof. Dr. Elif Vural
Electrical and Electronics Engineering, METU _____

Prof. Dr. A. Aydın Alatan
Electrical and Electronics Engineering, METU _____

Assoc. Prof. Dr. Fatih Kamışlı
Electrical and Electronics Engineering, METU _____

Assoc. Prof. Dr. Seniha Esen Yüksel
Electrical and Electronics Engineering, Hacettepe Uni _____

Assist. Prof. Dr. Gökhan Gültekin
Electrical and Electronics Engineering, Yıldırım Beyazıt Uni _____

Date:

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Onur Selim Kılıç

Signature :

ABSTRACT

UTILIZATION OF EVENT BASED CAMERAS FOR VIDEO FRAME INTERPOLATION

Kılıç, Onur Selim
M.S., Department of Electrical and Electronics Engineering
Supervisor: Prof. Dr. A. Aydın Alatan

August 2022, 71 pages

Video Frame Interpolation (VFI) aims to synthesize several frames in the middle of two adjacent original video frames. State-of-the-art frame interpolation techniques create intermediate frames by considering the objects' motions within the frames. However, these approaches adopt a first-order approximation that fails without information between the keyframes. Event cameras are new sensors that provide additional information in the dead time between frames. They measure per-pixel brightness changes asynchronously with high temporal resolution and low latency. The algorithms that use both the event-based information and the original frames overcome the problem of first-order approximation. However, they still have issues related to ghosting and the regions with insufficient events. This thesis aims to utilize visual transformers efficiently to combine event-based information and RGB frames to create better quality intermediate frames. The results show that the proposed video frame interpolation technique surpasses the state-of-the-art methods.

Keywords: Event Based Cameras, Video Frame Interpolation

ÖZ

VİDEO ARADEĞERLEME İÇİN OLAY TABANLI KAMERALARIN KULLANIMI

Kılıç, Onur Selim
Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. A. Aydın Alatan

Ağustos 2022 , 71 sayfa

Video Karesi Aradeğerlendirmesi (VKA), orijinal videonun iki ardışık karesinin arasında kareler sentezlemeyi amaçlar. Modern kare aradeğerlendirme teknikleri, nesnelerin hareketlerini dikkate alarak ara çerçeveler oluşturur. Ancak, bu yaklaşımlar, ana kareler arasında bilgi olmadan başarısız olan birinci dereceden bir yaklaşımı benimser. Olay temelli kameralar, kareler arasındaki ölü zamanda ek bilgi sağlayan yeni sensörlerdir. Yüksek zamansal çözünürlük ve düşük gecikme ile piksel başına parlaklık değişikliklerini eşzamansız olarak ölçerler. Hem olay temelli bilgiyi hem de orijinal çerçeveleri kullanan algoritmalar, birinci dereceden yaklaşım probleminin üstesinden gelir fakat gölgelenme ve olayların yetersiz olduğu bölgelerle ilgili sorunları vardır. Bu tez, kaliteli ara çerçeveler oluşturmak için olay temelli bilgileri ve standart görüntüleri görsel dönüştürücüleri ile birleştirmeyi amaçlamaktadır. Sonuçlar, ortaya konulmuş olan tekniğin en güncel teknikleri geçtiğini göstermektedir.

Anahtar Kelimeler: Olay Temelli Kameralar, Video Kare Enterpolasyonu

To my family and who believed in me...

ACKNOWLEDGMENTS

First and foremost, I would like to express my sincere gratitude to my supervisor, Prof. Dr. A. Aydın Alatan, for his unwavering support from the first day. I have deeply experienced his generosity and understanding over the past few years. With his unparalleled guidance, he has shown me how to conduct and report academic research. I have always felt privileged to be a graduate student of such a brilliant person and be one of the members of his research laboratory.

Secondly, no thanks would be enough for my family's support. My parents, Tamer - Aysun Kılıç, my brother, Bayram Alper Kılıç, and my sister, Sevde Pınar Kılıç, have made everything all possible with their endless support. They have put their ultimate effort into my academic life from the very first day. They have always stood by me in my sorrows and joys and supported me endlessly.

Moreover, I should thank the group members of the METU Center for Image Analysis. I offer my special thanks to Ahmet Akman, who has helped me through my research and has been a great working partner. I thank Oğul Can for sharing his know-how and experience with me for my studies. I also thank Yeti Ziya Gürbüz for introducing research areas that have fascinated me from the first day.

I also thank my friends, especially my eating partner in crime, who have listened to me about my research during our dinners, lunches, travels, and studies. Undoubtedly, I would not have been where I am today without their encouragement and belief in me.

TABLE OF CONTENTS

ABSTRACT v

ÖZ vi

ACKNOWLEDGMENTS viii

TABLE OF CONTENTS ix

LIST OF TABLES xii

LIST OF FIGURES xiii

LIST OF ABBREVIATIONS xvi

CHAPTERS

1 INTRODUCTION 1

1.1 Motivation 1

1.2 Scope of the Thesis 4

1.3 Outline of the Thesis 5

2 EVENT-BASED CAMERAS: FUNDAMENTALS AND PRELIMINARY EXPERIMENTS 7

2.1 Operation of Event-based Cameras 7

2.2 Representations of Event Data 9

2.3 Pros and Cons of Event-based Cameras 12

2.4 Preliminary Experiments by an Event-based Camera 14

2.4.1 Tests With Small Fast-Moving Objects 14
2.4.2 Tests with Flying Drones 18
2.4.3 Discussion 20
2.5 Event-based Datasets and Event Camera Simulators 21
3 VIDEO FRAME INTERPOLATION USING DEEP LEARNING METHODS 25
3.1 Introduction 25
3.2 Video Frame Interpolation 26
3.2.1 Video Frame Interpolation Metrics 26
3.2.1.1 Peak Signal to Noise Ratio 26
3.2.1.2 Structural Similarity Index Measure 27
3.3 Traditional Video Frame Interpolation Techniques 28
3.4 Learning Based Video Frame Interpolation Methods 29
3.4.1 Frame-Based Interpolation Methods 29
3.4.1.1 Warping Based Methods 29
3.4.1.2 Kernel Based Methods 30
3.4.2 Multi-Camera Interpolation Methods 30
3.5 Video Frame Interpolation Transformer [84] 31
3.5.1 Architecture for Video Frame Interpolation Transformer [84] . 32
4 VIDEO FRAME INTERPOLATION USING EVENT-BASED INFORMATION 35
4.1 Introduction 35
4.2 Event-based Video Frame Interpolation 35
4.2.1 Event-only VFI Approaches 36

4.2.2 Joint Event and Frame Based VFI Approaches 36
4.3 Time Lens [90]: Event-based Video Frame Interpolation 36
4.3.1 Performed Ablation Studies for TimeLens Algorithm 38
5 PROPOSED METHOD 41
5.1 Motivation 41
5.2 Key Observations 42
5.3 Proposed Architecture 42
5.3.1 Training Details 46
5.4 Experimental Results 46
5.4.1 Test Results 47
5.4.2 Ablation Studies 49
5.4.3 Discussion 50
6 CONCLUSIONS & FUTURE WORKS 55
6.1 Conclusions 55
6.2 Future Works 56
APPENDICES
A ARCHITECTURE UTILIZED IN VIDEO FRAME INTERPOLATION . . 57
A.1 Vision Transformers 57
A.1.1 Multi-Head Self/Cross Attention 57
REFERENCES 61

LIST OF TABLES

TABLES

Table 2.1 A comparison of several event representation techniques for event-based camera data utilized in machine learning [5]. 11

Table 4.1 Ablation study presented in TimeLens [90]. 38

Table 4.2 Our ablation study conducted on BS-ERGB dataset with image sizes 256×256. 39

Table 5.1 Comparison of the proposed method in BS-ERGB dataset in low resolution. 47

Table 5.2 Comparison of the proposed method in BS-ERGB dataset in full scale. 47

Table 5.3 Ablation study for multi-head self-attention. 49

Table 5.4 Ablation study for the first stage of our proposed network. 49

LIST OF FIGURES

FIGURES

Figure 1.1 Computational and cost requirements over the last few years [55]. 2

Figure 1.2 Illustration of light capturing mechanism of event-based cameras [29]. 3

Figure 2.1 General hardware structure of DAVIS event-based camera [25]. . 8

Figure 2.2 The visualization of voxel grid representation for 5 consecutive frames with. The time between each consecutive frames is divided into 64 grids for this example. 10

Figure 2.3 The visualization of voxel grid representation for 5 consecutive frames with. The time between each consecutive frames is divided into 6 grids for this example. 10

Figure 2.4 Cameras used in the preliminary experiments. 14

Figure 2.5 Test set up for small fast-moving objects. 15

Figure 2.6 An typical output for fast-small moving object scenario by 2.6a event-based camera and 2.6b RGB camera with MoG2 algorithm. . . . 16

Figure 2.7 An typical output for fast-larger moving object scenario by 2.7c event-based camera and 2.7b RGB camera with MoG2 algorithm. . . . 17

Figure 2.8 A typical indoor drone test scenario. 19

Figure 2.9 A typical outdoor drone test scenario. 20

Figure 2.10 Example images of BS-ERGB dataset. [89] 22

Figure 3.1 An example to video frame interpolation [29]. 25

Figure 3.2 The optical flow constructed by [34]. 30

Figure 3.3 An example kernel based algorithm [62] 30

Figure 3.4 Overview of the proposed Video Frame Interpolation Transformer [84]. 32

Figure 3.5 Separable Spatial-Temporal Swin (SepSTS) attention layers by [84]. 33

Figure 4.1 The overall workflow of the TimeLens [90]. The figures show the loss function that has used to train each module. Similar modules are in the same color across the figures. 37

Figure 5.1 Events given to MSA and max pooling blocks. 44

Figure 5.2 SynBlock architecture. 44

Figure 5.3 Number of parameters versus PSNR graph for down scaled images. 48

Figure 5.4 Number of parameters versus PSNR graph for full scaled images. 48

Figure 5.5 Events given to SepSTS encoder block. 50

Figure 5.6 Comparison of 5.6a ground truth image, 5.6d TimeLens [90], 5.6c Video Frame Interpolation Transformer [84] and 5.6b our result for a ball sequence. 52

Figure 5.7 Comparison of 5.7a ground truth image, 5.7d TimeLens [90], 5.7c Video Frame Interpolation Transformer [84] and 5.7b our result for an elastic band sequence. 53

Figure 5.8 Comparison of 5.8a ground truth image, 5.8d TimeLens [90], 5.8c Video Frame Interpolation Transformer [84] and 5.8b our result for a fire sequence. 54

Figure A.1 The structure of the attention mechanism. Left: Scaled Dot-Product Attention. Right: Multi-Head Attention Mechanism [51]. . . . 58

LIST OF ABBREVIATIONS

AE-VFI	Attention Event Video Frame Interpolation
AI	Artificial Intelligence
CNN	Convolutional Neural Network
CV	Computer Vision
DAVIS	Dynamic and Active Vision Sensor
DVS	Dynamic Vision Sensors
EST	Event Spike Tensor
FPS	Frame per Second
LN	Layer Normalization
MOG2	Gaussian Mixture-based Foreground Segmentation Algorithm
MSA	Multi Head Self-Attention
MLP	Multi-Layer Perception
MSE	Mean Square Error
NLP	Natural Language Processing
PSNR	Peak-Signak to Noise Ratio
ResNet	Residual Neural Network
RGB	Red-Green-Blue
SLAM	Simultaneous Localization and Mapping
SSIM	Structural Similarity Index Measure
STS	Spatial-Temporal Swin Attention
SepSTS	Separable Spatial-Temporal Swin Attention
SynBlock	Frame Synthesis Blocks
SAE	Surface Active Events
VFI	Video Frame Interpolation

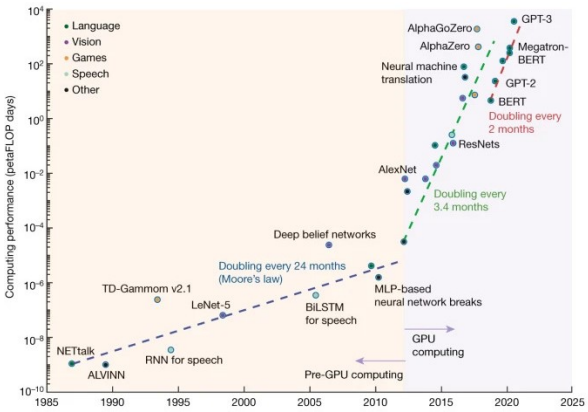
CHAPTER 1

INTRODUCTION

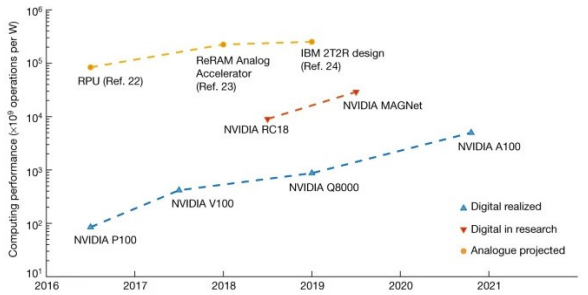
1.1 Motivation

A graduate student at Caltech, Misha Mahowald, started to work with Prof. Carver Mead in 1986 which was the first work to combine biological and engineering perspectives together on the stereo problem. In 1991, an image of a cat on the cover of a popular science journal[2], obtained by the novel "Silicon Retina" showed the world the exciting sight of neuromorphic engineering. Many researchers worldwide are still trying to understand how the brain works and to build one on a computer chip.

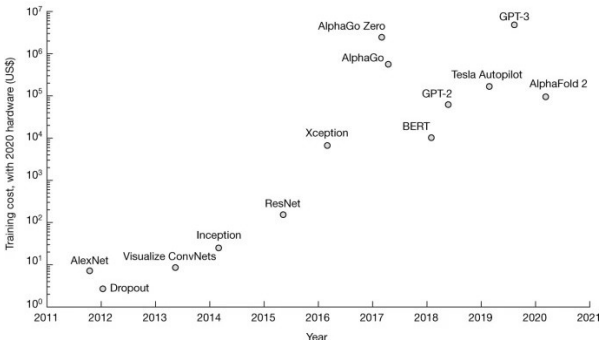
Computers today consume much too much energy. They are not sustainable platforms for the sophisticated artificial intelligence (AI) applications that are becoming more popular in our lives. Until 2012, processing power consumption doubled every twenty-four months; this has lately been reduced to every two months, which is shown in Figure 1.1a. As shown in Figure 1.1b, a combination of smart architecture and software-hardware co-design has produced remarkable advancements. At the research and development stage, even significant performance improvements have been achieved, and it is expected that we can accomplish much more. Unfortunately, traditional computer technologies will be unable to meet the demand in the near future. This demand is particularly noticeable when the expensive cost of training for the most complicated deep learning models is considered. Figure 1.1c shows the increasing demand over the years. Alternative techniques are required. Fortunately, the situation can be improved by drawing inspiration from biology, which takes a whole different approach to memory and processing, storing information in a completely different way or functioning directly on signals, and utilizing significant parallelism.



(a) The four-decade growth in computing power requirement.



(b) AI hardware efficiency over the years.



(c) Costs of training AI models.

Figure 1.1: Computational and cost requirements over the last few years [55].

Event cameras (event cameras and event-based cameras are used interchangeably) are a type of sensor that tries to emulate the functioning of biological retinas, such as [48], [70], [10], and [86], which aim to solve classical as well as new computer vision and robotic tasks. Event cameras can deliver new levels of performance and

efficiency in a wide range of applications which attracted both the academia and the industry. This result is due to the advantages that they offer to tackle problems that are difficult with standard frame-based image sensors. With their advanced hardware, event cameras are very promising in problems such as high-speed motion estimation [22], [26] or high dynamic range (HDR) imaging [75].

Event cameras are visual sensors that simply only record visual appearance changes in the scene, in contrast to their conventional counterparts, which capture light radiance at every point in the scene. See Figure 1.2 below.

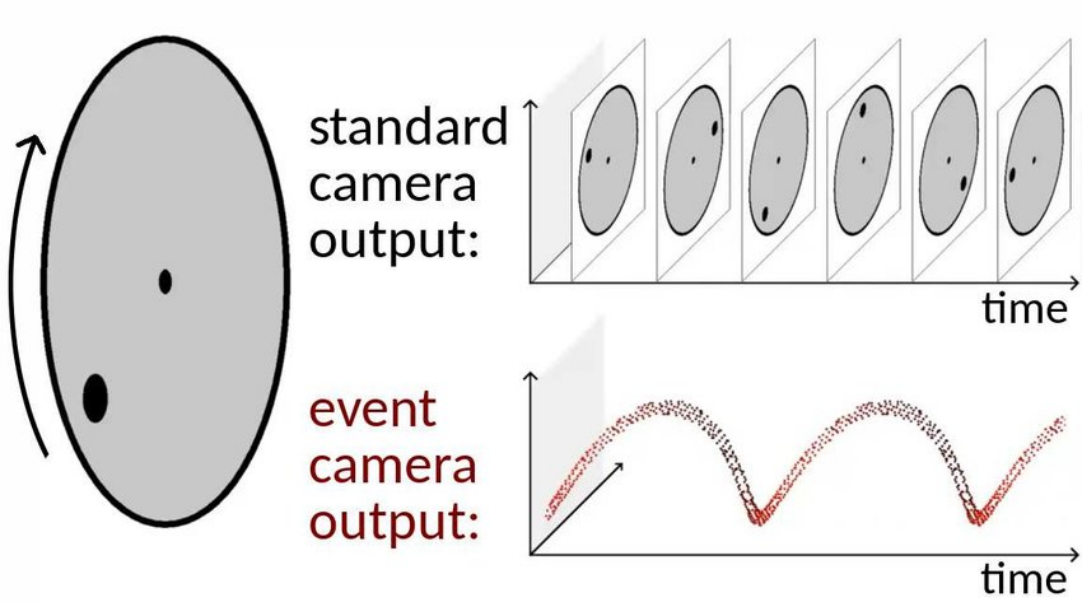


Figure 1.2: Illustration of light capturing mechanism of event-based cameras [29].

Event cameras change the way in acquiring visual information. This change is due to the fact that they do not use a clock that determines the instances in which the camera takes an image. Instead, they work asynchronously to capture the light in the changing areas of the scene. With their high temporal resolution, low latency (in the order of microseconds), very high dynamic range ($140dB$ vs. $60dB$ of standard cameras), and low power consumption, event cameras are very useful in robotics and wearable applications under challenging scenarios.

The main areas in which event cameras have advantages over other cameras include real-time interaction systems, robotics, and wearable devices [20]in which Uncontrolled lightning conditions, latency, and power [50] are essential. Additionally, event

cameras are used for object tracking [21],[30], surveillance [49], monitoring, and gesture recognition [64], [43], [3].

Event cameras are also used for depth estimation [79], [72], structured light 3D scanning [9], [100], HDR image reconstruction [75], [18], [39] and Simultaneous Localization and Mapping (SLAM) [40], [73], [92]. Latest research has shown that event cameras can also be used for image deblurring [66] or star tracking [16], [15]. As event cameras become more available for public use, the applications of event-based cameras are becoming diverse.

As a different and exciting application, capturing high-resolution videos with high frame rates often requires the use of professional high-speed cameras, which are out of reach for most regular users. Modern mobile device manufacturers have attempted to include more affordable sensors with comparable functionality into their systems, but they are still limited by the enormous memory needs and high power consumption associated with these sensors.

Video Frame Interpolation (VFI) solves this aforementioned problem by transforming moderate frame rate sequences into high frame rate videos in post-processing. However, using only keyframes to construct intermediate frames is a suboptimal solution, as the objects can move faster than the frame rate of the camera and cause motion blur or ghosting artifacts. Event-based cameras are not as expensive as the high frame rate cameras that are inaccessible to general users, and they provide valuable information in between the *deadtime* between keyframes.

Our motivation is to use both the data from the event camera and the frames from a standard low FPS imaging system to create high frame rate video sequences.

1.2 Scope of the Thesis

We first analyze some of the state-of-the-art video frame interpolation algorithms. Some of these algorithms use only frames, and some use both frames and events. By the results of these analyses, we present our method that will use both the events and the frames effectively, namely *Event Attention Video Frame Interpolation*. Finally,

we present the conducted experiments and the comparison of our method to the state-of-the-art video frame interpolation algorithms.

1.3 Outline of the Thesis

An overview of the event-based cameras and the preliminary experiments is presented in Chapter 2. Chapter 3 and Chapter 4 explains the video frame interpolation techniques that are using only frames and frames together with events, respectively. In Chapter 5, the proposed method and conducted experiments are explained in detail. Finally, Chapter 6 concludes this thesis with a summary and future works.

CHAPTER 2

EVENT-BASED CAMERAS: FUNDAMENTALS AND PRELIMINARY EXPERIMENTS

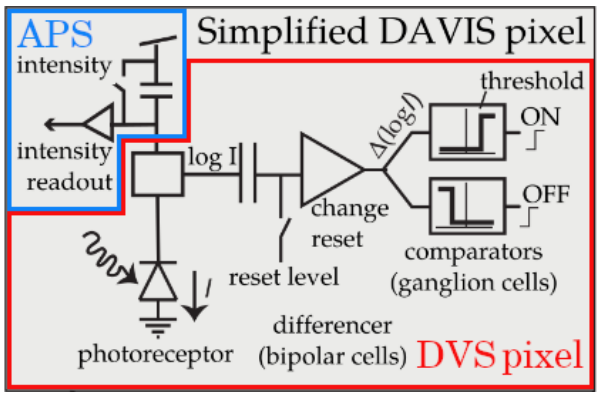
This chapter presents an overview of event-based cameras. Section 2.1 introduces the operations of event-based cameras. Section 2.2 introduces the current methods that are utilized to represent events. Section 2.3 explains the advantages and disadvantages of event-based cameras. Finally, Section 2.5 presents the existing datasets and simulators that are available for event-based vision in the literature.

2.1 Operation of Event-based Cameras

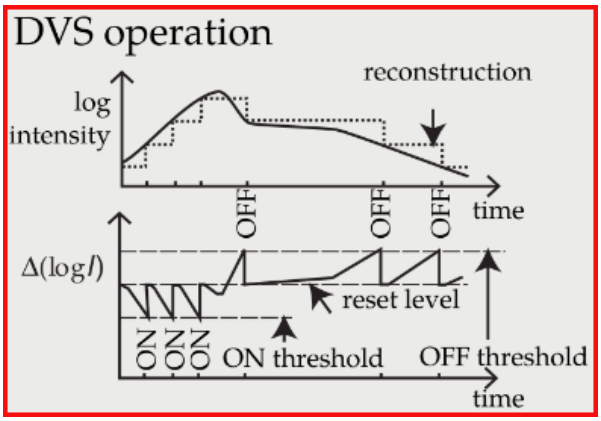
Event cameras, such as Dynamic Vision Sensor (DVS) [48], [44], [46], [47], [45], respond to illumination changes in the scene asynchronously and separately for every pixel, in contrast to conventional cameras, which capture entire pictures at a pace determined by an external clock as in Figure 1.2 (e.g. 30 fps). Figure 2.1a shows the basic circuit diagram of a general event-based camera and, Figure 2.1b shows the relation between the log intensity and event generation model of an event-based camera.

Since each event represents a change in the illumination (i.e. log intensity) of a preset magnitude at a pixel at a particular time, the output of an event camera is a configurable data-rate series of digital "events" or "spikes". The spiking behavior of the biological visual systems served as the inspiration for this mapping.

Every time a pixel delivers a signal, it learns the log intensity and constantly waits for a deviation from this learned value of a suitable magnitude. The camera transmits



(a) General circuit diagram for an event-based camera.



(b) Generation of events from the illumination changes.

Figure 2.1: General hardware structure of DAVIS event-based camera [25].

an event, which is sent from the chip by the x, y position, the time t, and the 1-bit polarity p of the change (i.e., illumination increase ("ON") or decrease ("OFF")), once the variation surpasses a threshold.

Event cameras are data-driven sensors; the output they provide is based on how much the scene is moving or changing in brightness. More events are produced per second the quicker the motion. These sensors respond rapidly to visual information since events are timestamped with microsecond precision and communicated with sub-millisecond delay.

The incident light at a pixel results from surface reflectance and scene illumination. If the amount of light is roughly constant, a change in log intensity indicates a shift in reflectance. The displacement of objects within the field of sight is mostly the

reason for these variations in reflectance. The DVS illumination change events have an invariance to scene lighting built into them due to this reason [48].

2.2 Representations of Event Data

There are several techniques of representing the events that are captured through an event-based camera. Although there are many representation techniques, most of the popular techniques use *individual events*, *event count*, and *voxel grids*.

Events are mostly obtained in the form of $(\boldsymbol{x_k}, t_k, p_k)$, where x is the pixel location of the event, p is the polarity, and t is the time stamp. Event-by-event processing techniques make use of single events, including probabilistic filters, [26, 39, 40], and spiking neural networks (SNNs) [78, 31]. Either the filter or SNN has extra information that is combined asynchronously with the received event to generate an output. This extra data could be obtained via more knowledge or built up from previous events. Following are a few instances that use the individual events approach [26], [39], [81], [95], and [68].

Event count representation creates an output by processing all of the events in a spatio-temporal region ($E \doteq \{e_k\}_{k=1}^{N_e}$). This encoding preserves the exact timing and polarity information. In order to meet the presumptions of the algorithm (such as constant movement speed for the packet), which change depending on the job, selecting the proper packet size N_e is essential. The approaches in [79], [72], [76], and [58] are some typical examples.

Each voxel in a voxel grid, which is a space-time (3D) histogram of events, corresponds to a particular pixel and time period. By not collapsing the events onto a 2D grid, this representation keeps the temporal features of the events more effectively. If the polarity is utilized, the voxel grid is an intuitive discretization of a scalar field formed on the picture plane, after the exclusion of events denoted by zero polarity (polarity $p(x, y, t)$ or brightness variation $L(x, y, t)/t)$. The polarity of each event might be distributed over its nearest voxels [75], [101], [74] or collected on a single voxel [7], [93]. The former (interpolated voxel grid) offers sub-voxel precision while both approaches quantize event timestamps.

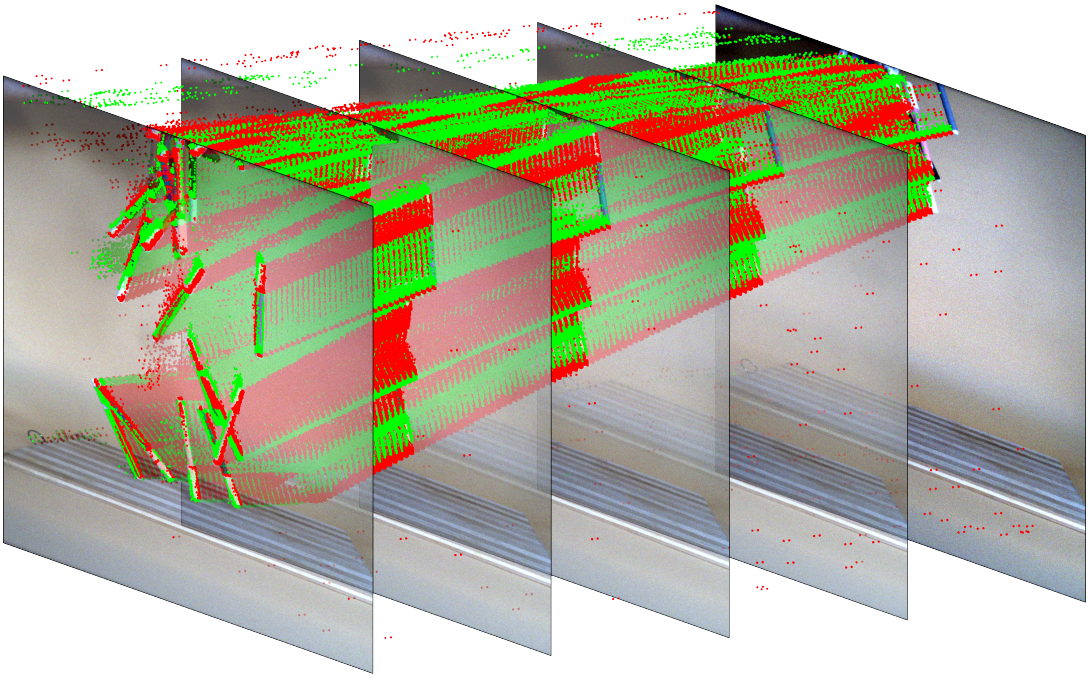


Figure 2.2: The visualization of voxel grid representation for 5 consecutive frames with. The time between each consecutive frames is divided into 64 grids for this example.

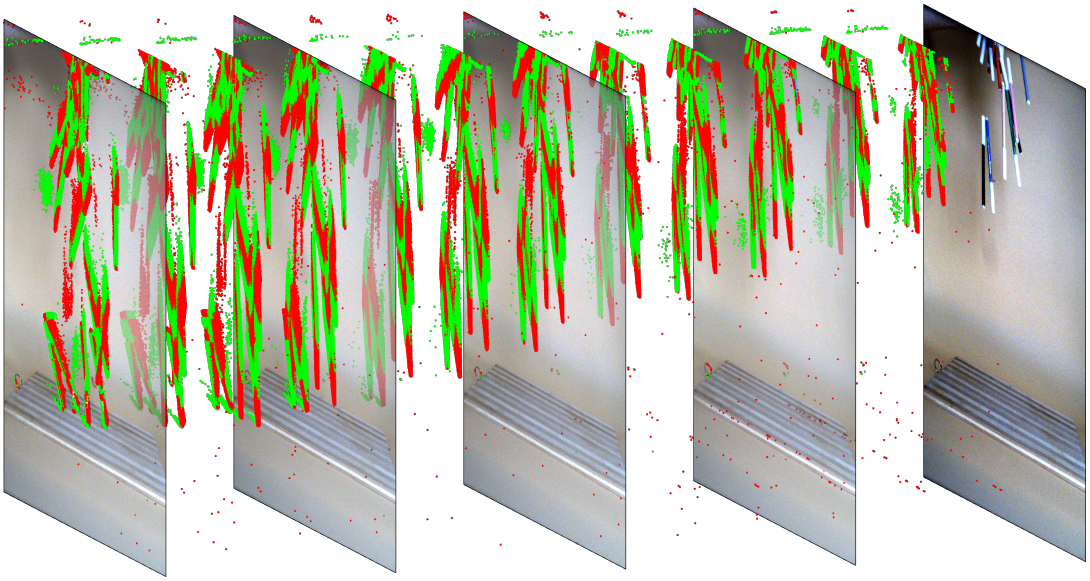


Figure 2.3: The visualization of voxel grid representation for 5 consecutive frames with. The time between each consecutive frames is divided into 6 grids for this example.

Table 2.1: A comparison of several event representation techniques for event-based camera data utilized in machine learning [5].

Event Representation	**Dimensions**	**Description**
Event Frame	HxW	Image of event polarities
Distance Transform	HxW	Image of spatial distance to active pixel
Graph-based	UxV	Graph of edges and vertices
Event Count	2xHxW	Image of event counts
Surface of Active Events (SAE)	2xHxW	Image of most recent timestamp
Voxel Grid	BxHxW	Voxel grid summing event polarities
Averaged Time Surfaces	2xHxW	Image of average timestamp for window
Inceptive Time Surfaces	3xHxW	Image of filtered timestamps & event count
Event Spike Tensor (EST)	2xBxHxW	4D grid of convolutions
TORE Volumes	2xKxHxW	4D grid of last K timestamps

Table 2.1 various numerous published methodologies and demonstrates the rapid increase in representation complexity, where H and W are input height and width. Most of these algorithms are designed for high-level tasks, including object identification, image reconstruction, and optical flow. Large representations are commonly used in certain scenarios, where H and W are the sensor height and widths. On the other hand, smaller spatial patches of events are employed in low-level or event-level applications, such as denoising or feature tracking. H and W in this example denote the height and width of the patches centered around the event of interest, which are substantially smaller than the sensor height/width.

In Table 2.1, "Event accumulation" or "event counting" segregate events within a temporal frame, and events are counted for each polarity type to generate two pictures of size $H \times W$. "Event frames" or "polarity summation," which computes the polarity of events inside a specific time window, is a similar method. Graph-based approaches convert events inside a temporal window into a network of nodes called $U \times V$. This compact representation provides for a decrease in CPU and memory resources while maintaining the sparse nature of event cameras. Time surfaces on the Surface of Active Events (SAE) [9] keep the timing for the most recent event at each pixel point (for each polarity). This approach is not appropriate for slow-motion sequences or

low-light imaging because event timing becomes imprecise or is interrupted by noise. The fact that a single edge creates many events can be used to boost robustness. An averaged time surface [85], in which each pixel value indicates the average event time of events created within a temporal window, decreases the influence of noise appearing in isolation. Inceptive time surface [4] is a time surface and polarity summation hybrid technique. The Event Spike Tensor (EST) [28] is a polarity-separated version of the voxel grid in [101], allowing any function to be performed per voxel rather than only summing polarity information. TORE volumes are a dense event representation that encodes a history of recent events.

In this thesis, we mainly utilized the interpolated voxel grids for two main reasons. The first reason is that the event voxel representation is usable in deep neural networks as they are 3 dimensional matrices. Secondly, they store spatial and temporal information quite effectively, assuming a sufficient number of grids. In Figure 2.2, an example voxel grid is presented. In this figure, all of the events are separated to 64 grids by their time timestamps. Similarly, Figure 2.3 shows an example voxel grid representation where the grid size is selected as 6. It should be noted that both of these selections are only made for visualization purposes.

2.3 Pros and Cons of Event-based Cameras

Event cameras are superior to standard cameras for several reasons. First, pixels of event cameras work independently, and waiting for a specific exposure time to take the frame is unnecessary. In other words, every pixel sends the information as soon as it receives a change. This property makes the latency of the event-based cameras very low; i.e. every pixel can send new information within the order of milliseconds.

Secondly, event-based cameras have a high dynamic range compared to normal ones. Generally, good quality standard cameras have 60 dB dynamic range, whereas most of event-based cameras exceed 120 dB dynamic range. The reason is that the photoreceptors of event-based cameras work on a logarithmic scale, and there is not a single exposure time for all of the pixels like in standard cameras.

The third reason is that event cameras work with much lower power than standard

cameras. As event cameras only detect changes in the scene, energy is only consumed on the changing pixels. Most event cameras work in the range of 10 mW level, and some other prototype event cameras even work with less than ten μW [3], [17], [83], [96].

Additional to these advantages, event-based cameras work with microsecond resolution. In other words, events can be detected and stamped, theoretically, with a 1-MHz clock. Given the requirement of the controllers for rapid response in autonomous vehicles during emergency driving scenarios, this feature is hugely helpful in autonomous vehicles.

Finally, the event-based cameras are almost unaffected by the motion blur or ghosting effect. The motion blur issue arises when the motion of moving objects exceeds the sampling frequency of the frame-based camera, causing the perception system to fail. An event-based neuromorphic vision sensor can accurately detect dynamic motion with no motion blur.

On the other hand, event-based cameras do not provide visual information in the scene, since they only signal the changes as binary information. Moreover, if too many events are created in the scene, the system might not be able to deliver this information through its pipeline. Since the technology is new, typical imaging resolutions are relatively low, while the prices of event cameras can not compete with CMOS or CCD counterparts.

In order to, investigate these advantages of event cameras better, we have conducted some experiments that put forth these advantages. For that purpose, we have completed two main tests with an event camera (Inivation Davis346 Color) with the help of some objects, a drone that can fly indoors, a drone that can fly outdoors at high speeds, and a high-quality surveillance camera (Sony IMX385-HiSilicon 3519Av100) to compete against the event-based camera. Figure 2.4, shows the cameras that are utilized in these tests.

(a) An image of Sony IMX385.

(b) An image of Inivation DAVIS346.

Figure 2.4: Cameras used in the preliminary experiments.

2.4 Preliminary Experiments by an Event-based Camera

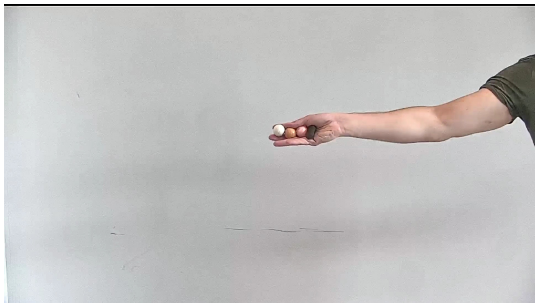
Since event-based cameras are very rare, in order to present our experiences, some simulations using such cameras will be given. The aim is to compare a standard event-based camera with a conventional RGB camera with object detection capability. For this purpose, the conventional RGB camera is utilized with a popular foreground/background detector, namely Mixture of Gaussians (MoG) [77].

2.4.1 Tests With Small Fast-Moving Objects

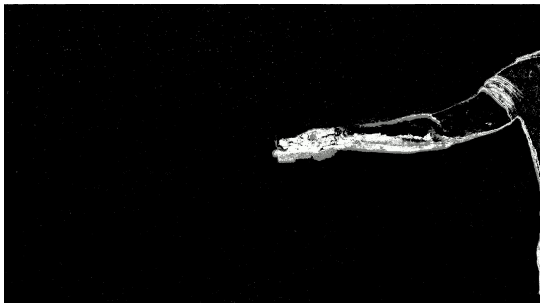
In order to understand the effect of the small-sized, fast-moving objects in standard and event-based cameras, we conducted tests with different contrast with respect to the background. During these tests, we used several different colored balls that are difficult to notice on a white background. In order to make the situation harder, the balls are thrown at high speeds in front of both cameras. We placed the event-based and state-of-the-art surveillance cameras toward the white wall and threw the balls in the order from white to dark. Figure 2.5a shows the balls in the order (left to right) they are thrown. Since the balls are moving quite fast, it is almost impossible for someone to detect them in the RGB videos. Therefore comparing the two cameras qualitatively, we have used one of the most popular background subtraction algorithms, namely Gaussian Mixture-based Background/Foreground Segmentation

Algorithm (MoG2) [87], [71]. Figure 2.5b shows the output of MoG2 algorithm for Figure 2.5a. Figure 2.5c shows the associated output of the event based camera.

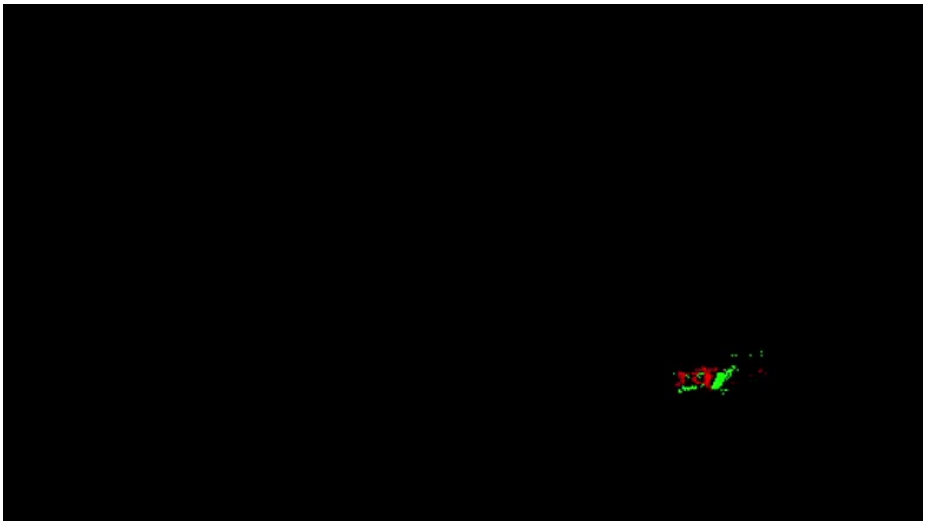
It should be noted that the output of the event-based cameras is rendered at 30FPS for visualization purposes. In other words, all the events for one-thirtieth of a second are collected and displayed them in a 2-dimensional image for interpretation. It should be noted this approach is performed for the events to be interpretable, as the event-based cameras provide events with microsecond precision.



(a) RGB Image.



(b) MoG2 algorithm output.

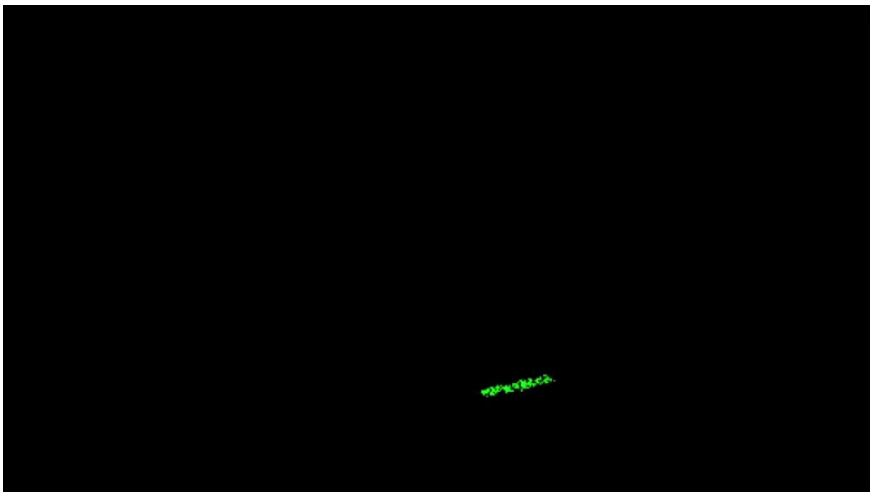


(c) Event-based camera output of Sony IMX385.

Figure 2.5: Test set up for small fast-moving objects.

Due to the similar color of the balls with respect to the background, it is challenging to detect them, especially when they are moving at high speeds. The ghosting effect in RGB images makes it almost impossible for the surveillance camera to detect a decent visual capture of the balls. On the other hand, low-latency event-based cameras do not

suffer from the ghosting effect much and create a clearer picture of moving objects. Figure 2.6b shows the output of the MoG2 algorithm that has used the RGB data taken from the surveillance camera, whereas Figure 2.6a shows the output of the event-based camera. One can easily see that the event-based camera performs much better than the surveillance camera with this setup.



(a) Event-based camera output.

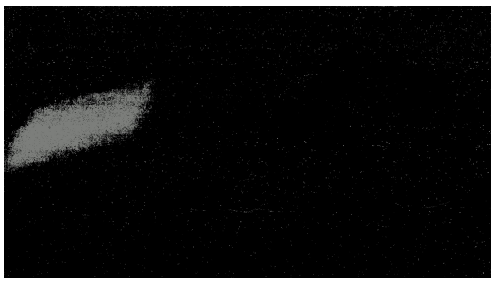
(b) MoG2 algorithm output.

Figure 2.6: An typical output for fast-small moving object scenario by 2.6a event-based camera and 2.6b RGB camera with MoG2 algorithm.

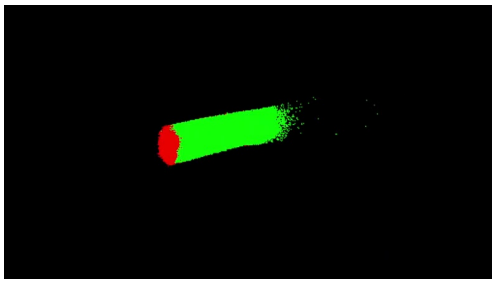
On the other hand, to understand the effect of moving larger objects in the scene, we threw more oversized objects to compare the outputs of the two cameras. As

the object sizes get larger, they become more visible to both cameras. However, the RGB camera is still affected by the ghosting effect. Figure 2.7a demonstrates the objects that are used in this test. We threw not only the tiny balls that were used in the previous test but also a larger sphere-shaped object. Figure 2.7b and Figure 2.7c show the output of the MoG2 algorithm fed with the image from the surveillance camera and the output of the event-based camera, respectively. The test shows that, as the object size gets larger, the RGB images that the surveillance camera grabs perform better. However, the RGB camera can still detect the object in only two frames, and the captured object silhouette is much worse than the one captured by the event-based camera.

(a) The illustration of the objects that are thrown in this test.



(b) MoG2 algorithm output.



(c) Event-based camera output.

Figure 2.7: An typical output for fast-larger moving object scenario by 2.7c event-based camera and 2.7b RGB camera with MoG2 algorithm.

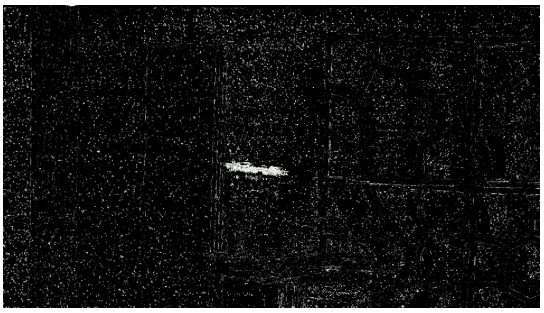
2.4.2 Tests with Flying Drones

With the fast advancement of unmanned vehicles and the technology needed to build them, the number of drones developed for military, commercial, or recreational reasons grows exponentially with each passing day. When cameras or weaponry are added to the drones, this arrangement offers serious privacy and security risks. As a result, recognizing the position and characteristics of drones, such as speed and direction, before an unpleasant event has become critical [41]. Addressing this issue, we have also wanted to observe the performance of event-based cameras in drone detection compared to a high-performance surveillance camera.

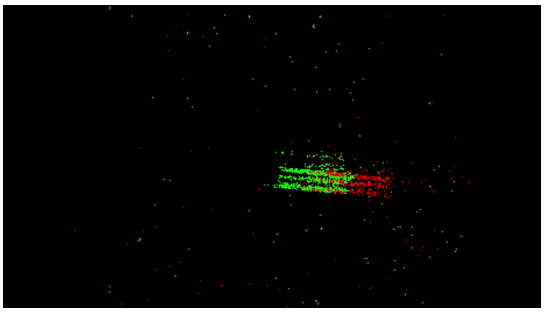
We have used two different drones that are developed for different purposes. One of the drones (Diatone 2017 GT200S FPV Racing Drone) is mainly designed for outdoor applications. This drone can reach speeds of 100 kilometers per hour and maneuver quite rapidly. The other drone (Kingkong 90GT 90mm Brushless Mini FPV Racing Drone), which is relatively smaller than the former one, is used mainly for indoor applications and has better maneuvering ability. Still, both of the drones, when they move close to the cameras, are considerably hard to detect, and they create a similar scenario with the ball tests.

The main differences between the event-based and standard cameras are speed and high dynamic range. The best way to better illustrate these differences is when trying to catch a fast-moving object at night. For this purpose, in the first of our drone tests, we arranged the lightning conditions to simulate night-time conditions and made the drone pass very quickly in front of the cameras with the help of a professional drone operator. Although the surveillance camera is one of the state-of-the-art cameras to adjust its exposure, Figure 2.8a shows that the captured image of the drone is poor. Figure 2.8b, which shows the output of the MoG2 algorithm, indicates that the captured image of the drone is observed as a small line in the image, and it would be almost impossible to interpret by any algorithm. Figure 2.8c displays the output of the event-based camera. It is easily seen that the output of the event-based camera creates a more clear picture of the indoor drone. The propellers can be distinguished as different lines, and the drone can be easily separated from the background.

(a) The captured RGB image.



(b) MoG2 algorithm output.



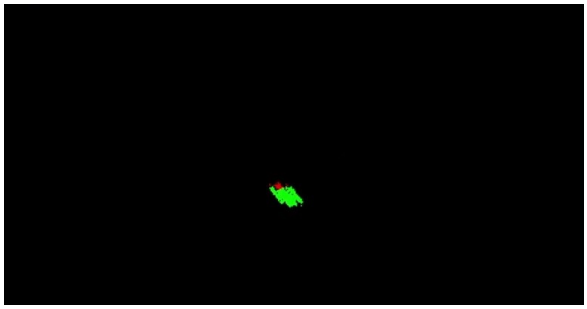
(c) Event-based camera output.

Figure 2.8: A typical indoor drone test scenario.

Figure 2.9 shows the output images related to one of the outdoor drone tests. As it can be observed from the images, as the drones pass away from the cameras, the drones become more visible to both of the cameras. Figure 2.9a and Figure 2.9b display the RGB and MoG2 output results. The drone can be separated from its surroundings quickly in these images. Figure 2.9c illustrates the output of the event-based camera. Similar to RGB and MoG2 results, the drone is clearly visible. In fact, the main difference between the surveillance camera and the event-based camera is evident in the videos of these tests. Within the videos, the movement of the drones is much more traceable for event-based cameras if the events are rendered at a slower frame rate.

(a) The captured RGB image.

(b) MoG2 algorithm output.



(c) Event-based camera output.

Figure 2.9: A typical outdoor drone test scenario.

2.4.3 Discussion

Neuromorphic systems and event-based cameras are quite promising for any vision-based real-life problem.

The performed tests to understand the advantages of event-based cameras over standard cameras in a better way indicate that the contrast of the object with respect to the background, the object size, and their speed are three critical factors when capturing a frame and detecting an object. Comparing the results of the indoor and outdoor drone

tests, we have seen that these two factors are even more critical when challenging lighting conditions. The smaller drone used for indoor tests was unrecognizable in the standard camera due to its speed and size, as it was only observable in a small number of frames. However, for the event-based camera, the drone has resulted in a significant number of events, and the silhouette of the drone was much more visible compared to the standard surveillance camera. Similarly, the ball tests for the white wall-white ball test have shown that for better interpretation in time, the frame rate of the visual RGB camera must be increased with high quality. Therefore, we have decided to examine the video frame interpolation (VFI) by using both the event information and the images obtained from a standard camera.

It should be reminded event-based cameras have some disadvantages as well as good properties. In other words, the event-based camera can not provide the color of the objects that appear in the scene. This deficiency makes the problem of constructing original frames from only the events an ill-posed problem. Hence, a novel approach should be required to intelligently fuse RGB camera outputs with that of event cameras.

2.5 Event-based Datasets and Event Camera Simulators

The number of event-based vision simulators and datasets is rapidly increasing. They may be broadly divided into three groups: ones that focus on motion estimation or image reconstruction (regression) tasks, those that target recognition (classification) tasks, and those that target end-to-end human-labeled data, such as driving [33]. Datasets for object tracking, segmentation, optical flow, SLAM, and other techniques are included in the first group. Datasets for recognizing objects and actions make up the second group.

There are three datasets for optical flow: [100], [80], and [8]. Related to the difficulty of obtaining ground-truth optical flow, [80] considers only flow in solely rotational motion captured with an IMU, and hence the dataset lacks flow due to translational (parallax) motion. [100], [8] datasets give optical flow as the motion field created on the picture plane by camera motion and scenario depth (measured with a range

sensor, such as an RGB-D camera, a stereo pair, or a LiDAR). Naturally, noise and errors in the alignment and calibration of the several associated sensors might affect ground truth optical flow.

(a) An example of a fire image in BS-ERGB dataset.

(b) An example of a watermelon image in BS-ERGB dataset.

(c) An example of an aquarium in BS-ERGB dataset.

Figure 2.10: Example images of BS-ERGB dataset. [89]

The datasets provided in [94], [59], [8], and [23] are mostly utilized pose estimation and SLAM. The most widely used dataset is presented in [59], and it has been utilized to evaluate visual odometry and visual-inertial odometry methods [92], [76], [58], etc.

This dataset is also often used to test edge detectors and feature trackers.

The recognition datasets are generally smaller in size with respect to standard computer vision datasets. They are made up of deck cards (4 classes), faces (7 classes), handwritten numerals (36 classes), movements (rocks, papers, scissors) in dynamic settings, vehicles, and other objects. Using saccade-like movements, neuromorphic, versions of prominent frame-based computer vision datasets such as MNIST and Caltech101 have been produced. In real-world circumstances, newer datasets [3], [85], [57], are collected (not generated from frame-based data). These datasets have been used to test event-based recognition algorithms.

On the other hand, the DVS emulators in [38], [27], and [59] are based on the operating principle of an ideal DVS pixel. The simulator creates the necessary sequence of events, brightness images, and depth maps given a virtual 3D scene and the trajectory of a moving DAVIS inside it. Since no extensive analysis of the noise and dynamic impacts of existing event cameras has been conducted, the noise models employed are currently simple and crude.

During the tests, we have used datasets with event-based information obtained from an event-based camera rather than utilizing simulators to generate events. This is because the events generated by simulators do not necessarily generate proper events with crucial information about the scene dynamics. Apart from the dataset that we have collected by the drones and the balls, we have used BS-ERGB [89], HS-ERGB[90] datasets. Figure 2.10 shows some examples of BS-ERGB dataset.

CHAPTER 3

VIDEO FRAME INTERPOLATION USING DEEP LEARNING METHODS

3.1 Introduction

High-resolution videos with high frame rates often demand the use of professional high-speed cameras, which are unaffordable to consumers. Modern mobile device manufacturers have attempted to include more cheap sensors with similar functionality into their systems, but they still suffer from the massive memory needs and power dissipation associated with these sensors.

Video Frame Interpolation (VFI) overcomes this issue by transforming videos with moderate frame rates to high frame rate videos in post-processing. In principle, any number of additional frames can be created between two key-frames in the input video. Figure 3.1 shows an example of video frame interpolation to a sequence of rider images.

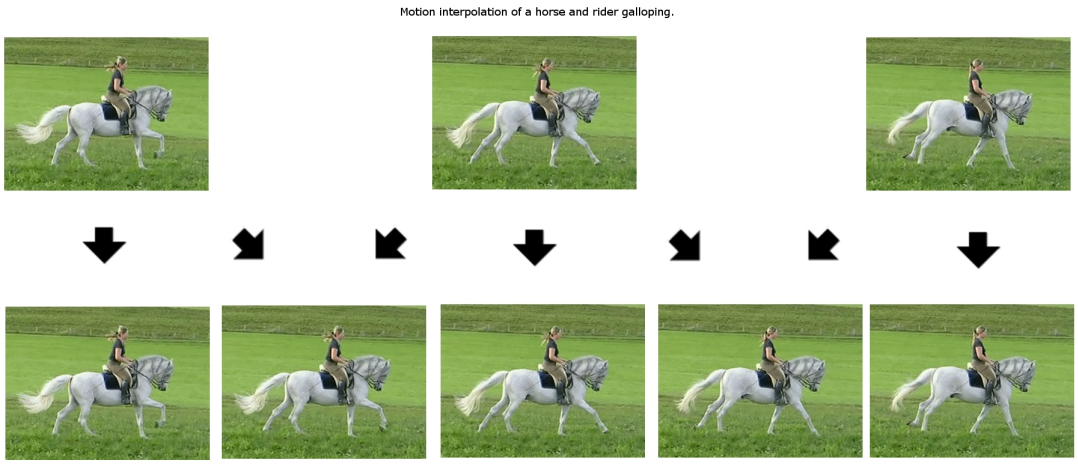


Figure 3.1: An example to video frame interpolation [29].

This chapter presents an overview of the video frame interpolation methods based on deep learning techniques. These techniques utilize only keyframes to generate intermediate frames. Traditional video frame interpolation techniques are introduced in Section 3.3. Section 3.4 gives an outline of learning-based video frame interpolation methods, where section 3.4.1 represents the algorithms that use one camera and multiple key-frames and, Section 3.4.2 introduces algorithms that utilize multiple cameras. Finally, Section 3.5 presents the state of the video frame interpolation algorithm that we have used to develop our method.

3.2 Video Frame Interpolation

In applications such as consumer electronics, surveillance, video recovery, and video post-processing, video frame interpolation is a crucial field of computer vision research. By generating intermediate frames between successive input frames, VFI seeks to raise the frame rate of a video sequence. This goal acquires exceptionally smooth, precise motion to make animation fluent enough and minimize display motion blur. Large-scale diversified video footage may be used to uncover information using advanced deep learning algorithms. These methods reveal intermediate motion information and provide fresh possibilities for advancing video interpolation technology.

3.2.1 Video Frame Interpolation Metrics

The main VFI metrics that are present in the literature are Peak Signal to Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM).

3.2.1.1 Peak Signal to Noise Ratio

Peak signal-to-noise ratio (PSNR) is an engineering term describing the ratio of a signal's greatest strength to the power of corrupting noise which degrades the representational accuracy of the signal. The mean squared error provides the most precise definition of PSNR (MSE). Given a noisy approximate K of a noise-free m by n

polychromatic picture I, MSE is defined with 3.1.

$$MSE = \frac{1}{m \times n \times C} \sum_{k=0}^{C-1} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(k,i,j) - K(k,i,j)]^2 \tag{3.1}$$

Using this definition of MSE, the PSNR can be defined in dB as 3.2.

$$PSNR = 10 \times \log_{10}(\frac{MAX_I^2}{MSE}) \tag{3.2}$$

$$PSNR = 20 \times \log_{10}(\frac{MAX_I}{\sqrt{MSE}}) \tag{3.3}$$

$$PSNR = 20 \times \log_{10}(MAX_I) - 10 \times \log_{10}(MSE) \tag{3.4}$$

In equations 3.2, 3.3 and 3.4, MAX_I corresponds to the maximum possible value of the image. Generally, 8 bits are used per pixel where MAX_I becomes 255.

3.2.1.2 Structural Similarity Index Measure

The perceived quality of digital photos and videos may be predicted using the structural similarity index measure (SSIM). SSIM is a tool for calculating how similar two photos are to one another. The SSIM index is a complete reference metric, meaning that the initial uncompressed or distortion-free picture serves as the baseline for the measurement or prediction of image quality.

The SSIM index is calculated on various windows of an image. The distance between two windows with common sizes $N \times N$ is expressed as equation 3.5.

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{3.5}$$

where:

μ_x : the average of x

μ_y : the average of y

σ_x^2 : the variance of x

σ_y^2 : the variance of y

σ_{xy} : the covariance of x and y

c_1 and c_2 : two variables to prevent division by 0

3.3 Traditional Video Frame Interpolation Techniques

Most methods for motion-compensated frame interpolation in the literature employ block-wise processing, where all of the pixels in a block are presumptively moving in the same direction. As a result, all processing operations, such as motion estimation, motion compensation, and occlusion handling, are carried out block-by-block. Since block size has uncertain effects on motion prediction, a frequent strategy is to start with a large block size and gradually reduce it. There are several different motion estimating techniques, including complete search, pattern-based search, and candidate-based prediction.

Block-wise true predict-and-update type motion estimation methods and conventional search type motion estimation are two categories into which the block-wise motion estimation techniques in the literature can be categorized. The full search method offers the best performance compared to all other traditional search-type motion estimation techniques. Therefore, typical motion estimation algorithms aim to minimize performance degradation while reducing computational complexity.

Accurate motion vector estimation is necessary to achieve acceptable visual quality at the interpolated frames. Since this requirement is not feasible in real-world settings, motion compensation algorithms often seek to mask the visual errors brought on by inaccurate motion vectors.

3.4 Learning Based Video Frame Interpolation Methods

Learning based VFI techniques can be categorized into two: The first category is the Frame-Based Interpolation Methods which solely depend on images taken by one camera that takes images at a fixed frame rate. The second category, Multi Camera Interpolation Methods, uses multiple cameras with different temporal characteristics.

3.4.1 Frame-Based Interpolation Methods

The frame-based interpolation techniques that use training data can be divided into two main classes, which are warping-based and kernel-based techniques.

3.4.1.1 Warping Based Methods

Warping based methods [60, 36, 99, 61, 69] construct intermediate frames between two successive keyframes by combining optical flow estimates [34, 54, 88] and image warping [35].

These approaches, in particular, calculate optical flow and warp the input keyframe(s) to the desired frame under the assumptions of linear motion and brightness constancy across frames, while using ideas such as contextual information [60], visibility maps [36], spatial transformer networks [99], forward warping [61], or dynamic blending filters [69] to enhance the results. While the majority of these methods assume linear motion, several recent studies consider quadratic [98] or cubic motion [14]. Although these approaches can handle non-linear movements, they are still constrained by their order and cannot handle arbitrary motion. To some extent, the motion models remain quite simple. Figure 3.2 shows an example of an optical flow construction. The timings that are shown at the top of the images stand for the particular algorithm's [34] execution time, and they change according to the image sizes.

Figure 3.2: The optical flow constructed by [34].

3.4.1.2 Kernel Based Methods

Kernel based methods [62, 63] bypass the warping-based techniques' explicit motion estimate and warping steps. They represent VFI instead as local convolution over the input keyframes, with the convolutional kernel calculated from the keyframes. This method is more resistant to motion blur and variations in lighting. Alternatively, phase-based methods [56] models VFI as a phase shift estimation issue, with a neural network decoder explicitly estimating the intermediate frame's phase decomposition. However, while these approaches may theoretically simulate any motion, due to the locality of the convolution kernels, they do not scale to significant movements in practice. Figure 3.3 shows an example kernel based method architecture [62].

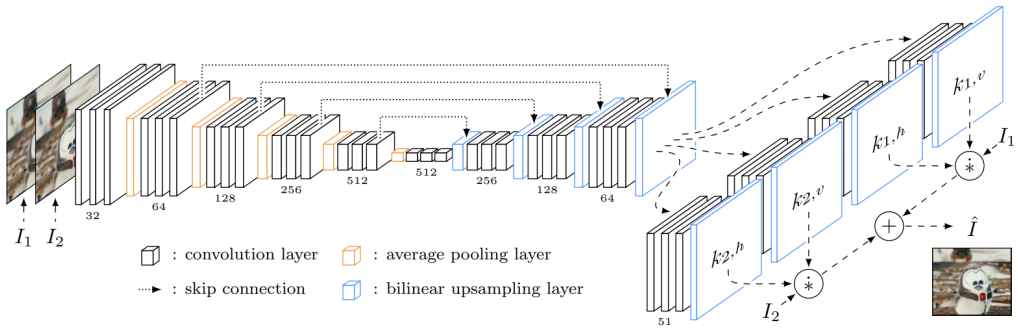


Figure 3.3: An example kernel based algorithm [62]

3.4.2 Multi-Camera Interpolation Methods

Multi-camera interpolation methods aim to solve the large motion problem by aggregating inputs from many frame-based cameras with varying spatio-temporal trade-offs. The authors of [1], for example, merged low-resolution video with high-resolution still photos, whereas [65] fused a low-resolution high-frame-rate video with a high-

resolution low-frame-rate video. Both systems are capable of recovering the missing visual information required to reconstruct accurate object movements, but at the expense of a bigger form factor, higher power consumption, and a larger memory footprint.

3.5 Video Frame Interpolation Transformer [84]

A quite promising kernel-based video frame interpolation technique, Video Frame Interpolation Transformer [84] is an algorithm published by Zhihao Shi et al. This method mainly utilizes vision transformers (Please see Appendix A for common architectures of vision transformers).

In contrast to traditional Transformers [12, 24, 13], where the fundamental modules are mostly derived from the original NLP models [91], the suggested VFIT has three distinct architectures to create photo-realistic and temporally-coherent frames.

To start with, the original Transformer [91] is built on a self-attention layer that interacts with the pixels globally. Due to the quadratic complexity of this global operation in terms of the number of variables, simply applying it to video frame interpolation results in enormously high memory and computing costs due to the high-dimensional structure of videos.

Several approaches [13, 11] work around this issue by splitting the feature maps into patches and using each patch as a new element in self-attention. This technique, however, cannot explain fine-grained relationships between pixels inside each patch, which are required for synthesizing realistic features. Furthermore, it may introduce edge artifacts around patch boundaries.

Video frame interpolation, on the other hand, incorporates Swin's [52] local attention technique into VFIT to handle the complexity issue while keeping the capacity to simulate long-range dependencies via its shift-window approach.

The Video Frame Interpolation Transformer illustrates how, with proper refinement and adaption, the local attention mechanism originally intended for image identification can effectively increase video interpolation performance with fewer parameters.

Second, the original local attention mechanism [52] is only appropriate for image input and cannot be simply applied to the video interpolation task that involves an extra temporal dimension.

To overcome this issue, video frame interpolation generalizes the idea of local attention to the spatial-temporal domain, resulting in the video-compatible Spatial-Temporal Swin attention layer (STS).

However, when employing large window dimensions, this simple addition might cause memory concerns. In order to make the model more memory efficient, video frame interpolation created a space-time separable form of STS [52, 53] known as Sep-STS [84] by factoring in the spatial-temporal self-attention.

3.5.1 Architecture for Video Frame Interpolation Transformer [84]

The network that is proposed by the Video Frame Interpolation Transformer [84] can be divided into two for our purposes. The first part is the encoder-decoder structure that is used to extract spatio-temporal information from the keyframes. This part includes Separable Spatial-Temporal Swin (SepSTS) attention layers and convolutional and deconvolutional layers. The second layer takes the features generated by the first layer and utilizes frame synthesis blocks (SynBlocks) to create intermediate frames. The overall structure of the network is presented in Figure 3.4.

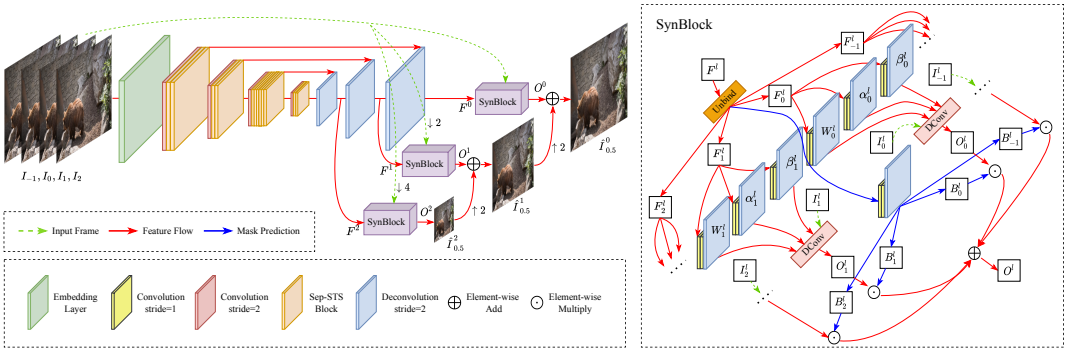


Figure 3.4: Overview of the proposed Video Frame Interpolation Transformer [84].

As shown in Figure 3.4, VFIT takes four images as its input and gives an output image corresponding to a middle image of these four images. The first critical operation that

those images go through is the encoder structure. Rather than the 3D convolutional layers, SepSTS blocks the fundamental part of the encoder block. These blocks are designed to utilize attention algorithms to extract spatial and temporal information in a memory-saving manner.

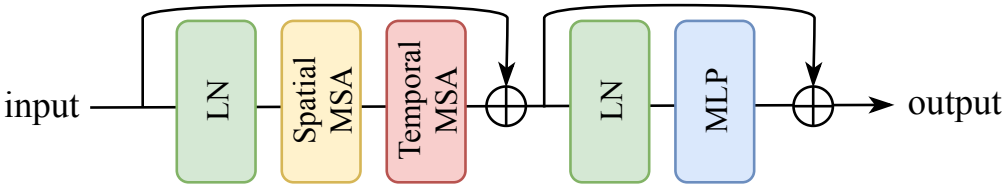


Figure 3.5: Separable Spatial-Temporal Swin (SepSTS) attention layers by [84].

The first step of a SepSTS block is layer normalization (LN). Then, there is a spatial and temporal multi-head self-attention blocks (Please see appendix A). Finally, the input is residually connected to the output of MSA blocks. Afterward, the output is given to another LN and a multi-layer perception (MLP) in a residual manner to obtain the output of SepSTS. It should be emphasized that MSA blocks are built differently from standard attention techniques in order to improve memory efficiency.

After the encoder, there are three deconvolutional layers that generate the feature maps that will be used by the frame synthesis blocks (SynBlocks). These blocks are the essential blocks in VFIT for our proposed method, and they are a multi-frame extension to AdaCoF [42].

Given an input feature map $F^l \in \mathbb{R}^{C\times T\times H\times W}$, the SynBlock predicts a frame that is a down-sampled version of the input by l times. To aggregate the information from the source frames, SynBlocks estimates a collection of generalized deformable kernels [97].

First F^l is unbinded at the temporal dimension to get T separate feature maps for all input frames, denoted as $F^l_{\{-(\lfloor \frac{T}{2} \rfloor -1),\cdots,\lceil \frac{T}{2} \rceil\}}$, and for each frame t, $F^l_t \in \mathbb{R}^{C\times H\times W}$. Then these are fed into three small convolutional CNN's to obtain per-pixel deformable kernels. The output of these three convolutional CNN's are kernel weights $W^l_t \in \mathbb{R}^{K\times H\times W}$, horizontal offsets $\alpha^l_t \in \mathbb{R}^{K\times H\times W}$, and vertical offsets $\beta^l_t \in \mathbb{R}^{K\times H\times W}$, where K is the number of sampling locations of each kernel. With these parameters, the output of the deformable convolution is constructed as 3.6.

$$O_t^l(x,y) = \sum_{k=1}^{K} W_t^l(k,x,y) I_t^l(x + \alpha_t^l(k,x,y), y + \beta_t^l(k,x,y)) \tag{3.6}$$

In equation 3.6, adaptive weights W are selected to aggregate neighboring pixels around (x, y), similar to [97].

Finally, the output at scale l is generated by blending O_t^l of all frames with learned masks. First, the feature maps $F^l_{\{-(\lfloor \frac{T}{2} \rfloor - 1), \cdots, \lceil \frac{T}{2} \rceil\}}$ are concatenated at the channel dimension and send to a small 2D CNN to produce T blending masks $B^l_{\{-(\lfloor \frac{T}{2} \rfloor - 1), \cdots, \lceil \frac{T}{2} \rceil\}}$. At the last layer of the CNN, a softmax function is used to normalize the masks. Finally, the output of the SynBlock f^l_{syn} is generated by:

$$O^l = \sum_t B_t^l \cdot O_t^l. \tag{3.7}$$

Considering the ideas of Video Frame Interpolation Transformer, we have shaped our original network, which will also use events.

Based on the experimental results presented in [84] and based on some of our preliminary tests, it was observed that VFIT algorithm generates quite remarkable interpolated frames with its proposed architecture. The deformable convolution approach, which is explained in 3.6 is argued to be a critical factor in generating high-quality intermediate frames. Since α_t^l, β_t^l and W_t^l parameters are estimated to determine sub-pixel displacements for the interpolated frame, one can propose that those sub-pixel displacements can be better estimated using event-based information.

CHAPTER 4

VIDEO FRAME INTERPOLATION USING EVENT-BASED INFORMATION

4.1 Introduction

This chapter presents an overview of the video frame interpolation methods that utilize event-based information. Although the number of techniques is quite limited for this purpose, these techniques mostly utilize keyframes and events generated by an event-based camera to create intermediate frames. Section 4.2 gives an overview of event-based algorithms, whereas Section 4.2.1 presents the only events approaches, and Section 4.2.2 explains the algorithms that use both the events and the frames. Finally, Section 4.3 explains one of the latest algorithms that use both the frames and the events, TimeLens [90], which we have studied and shaped our architecture accordingly.

4.2 Event-based Video Frame Interpolation

Event-based [48, 10] cameras are unique sensors that only report per-pixel intensity changes rather than full-image intensity changes, and they perform these goals with excellent temporal resolution and low latency on the scale of microseconds. The output is an asynchronous stream of binary "events," which may be thought of as a compressed version of the actual visual signal. These characteristics make them suitable for VFI in highly dynamic scenarios (e.g., high-speed non-linear motion or challenging illumination).

4.2.1 Event-only VFI Approaches

Approaches using only events use GANs [93], RNNs [74, 75, 82] or even self-supervised CNNs [67] to reconstruct high frame rate videos directly from a stream of incoming events and can be viewed as a proxy for the VFI problem. However, because integrating intensity gradients into an intensity frame is an ill-posed task, the global contrast of the interpolated frames is typically miscalculated. Furthermore, because the intensity edges of event cameras are only visible while they move, the interpolation results are similarly motion-dependent.

4.2.2 Joint Event and Frame Based VFI Approaches

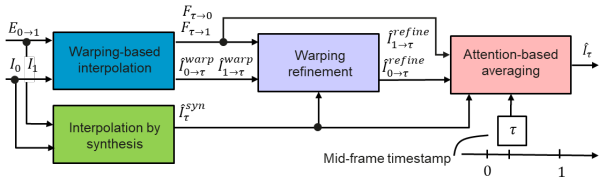
Some algorithms [66, 93] employ both streams of information since certain event cameras, such as the Dynamic and Active Vision Sensor (DAVIS) [10], may simultaneously output the event stream and intensity images – the latter at low frame rates and prone to the same difficulties as frame-based cameras (e.g. motion blur). These studies often address VFI in connection with deblurring, denoising, super-resolution, or other essential tasks. They generate intermediate frames by collecting and applying temporal brightness changes, denoted by events, from the input keyframes. While these approaches can manage illumination variations and non-linear motion, they nevertheless perform poorly when compared to frame-based systems, because not all brightness changes are reliably logged as events due to the inherent instability of the contrast threshold and sensor noise.

4.3 Time Lens [90]: Event-based Video Frame Interpolation

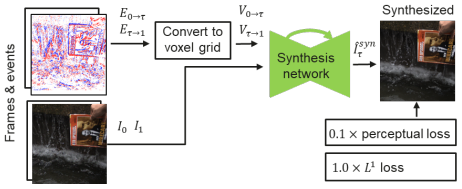
TimeLens [90] is a recent algorithm that uses both the events and the frames to interpolate images in between keyframes. There is also a successor of TimeLens, namely TimeLens++ [89].

Kernel-based and warping-based methods are described in Sections 3.4.1.2 and 3.4.1.1. TimeLens aims to combine both of these methods to create a structure that can use them together. Therefore, TimeLens have three substructures to construct intermedi-

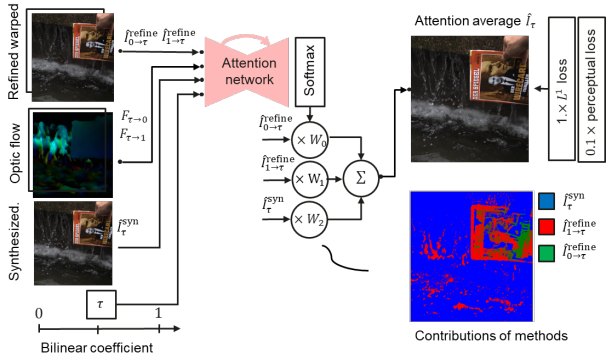
ate images and another structure that combines the results. The workflow of the algorithm is presented in Figure 4.1a. The main parts of the algorithm are the synthesis block (Figure 4.1b), warping block (Figure 4.1d), warping refinement block (Figure 4.1e), and the block that combines the results of these three blocks, the attention-based averaging block (Figure 4.1c).



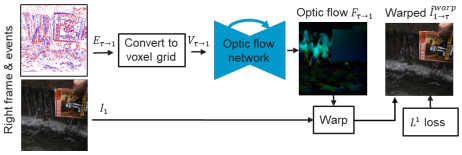
(a) Overview of the proposed method.



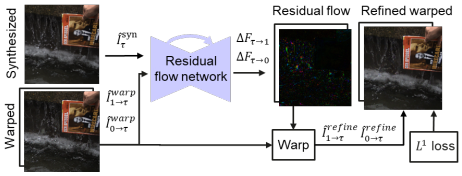
(b) Interpolation by synthesis module.



(c) Attention-based averaging module.



(d) Warping-based interpolation module.



(e) Warping refinement module.

Figure 4.1: The overall workflow of the TimeLens [90]. The figures show the loss function that has used to train each module. Similar modules are in the same color across the figures.

Given the left I_0 and right I_1 RGB keyframes and events sequences $E_{0\to\tau}$ and $E_{\tau\to1}$, interpolation by synthesis block, directly regresses a new frame $\hat{I}^{\text{syn}}$ as shown in Figure 4.1b. On the other hand, warping-based interpolation block, shown in Figure 4.1d, estimates the optical flow $F_{\tau\to0}$ and $F_{\tau\to1}$ between the new frame $\hat{I}_\tau$ and boundary keyframes I_0 and I_1 using events $E_{\tau\to0}$ and $E_{\tau\to1}$ respectively. Then the computed optical flows are used to warp the boundary keyframes in timestep τ using differentiable interpolation [35], which in turn produces two new frame estimates $\hat{I}^{\text{warp}}_{0\to\tau}$ and $\hat{I}^{\text{warp}}_{1\to\tau}$. Warping refinement module computes refined interpolated frames, $\hat{I}^{\text{refine}}_{0\to\tau}$ and

$\hat{I}^{\text{refine}}_{1\to\tau}$, by estimating residual optical flow, $\Delta F_{\tau\to 0}$ and $\Delta F_{\tau\to 1}$ respectively, between the warping-based interpolation results, $\hat{I}^{\text{warp}}_{0\to\tau}$ and $\hat{I}^{\text{warp}}_{1\to\tau}$, and the synthesis result $\hat{I}^{\text{syn}}_{\tau}$. It then proceeds by warping $\hat{I}^{\text{warp}}_{0\to\tau}$ and $\hat{I}^{\text{warp}}_{1\to\tau}$ for a second time using the estimated residual optical flow, as shown in Fig. 4.1e. Finally, the attention averaging module, shown in Fig. 4.1c, blends in a pixel-wise manner the results of synthesis $\hat{I}^{\text{syn}}_{\tau}$ and warping-based interpolation $\hat{I}^{\text{refine}}_{0\to\tau}$ and $\hat{I}^{\text{refine}}_{1\to\tau}$ to achieve final interpolation result $\hat{I}_{\tau}$.

4.3.1 Performed Ablation Studies for TimeLens Algorithm

Apart from the results given in TimeLens, we have conducted another ablation study to understand the effect of each individual block. Table 4.1 shows the original paper's ablation study conducted on the Vimeo90k dataset. This dataset does not originally contain event information. Thus, TimeLens have constructed events using an event simulator to obtain synthetic events. This approach might not always be the best approach to train/test a network based on mostly event information.

Table 4.1: Ablation study presented in TimeLens [90].

Module	**PSNR**	**SSIM**
Warping interpolation	26.68	0.926
Interpolation by synthesis	34.10	0.964
Warping refinement	33.02	0.963
TimeLens	35.83	0.976

In order to confirm the results of Table 4.1, we have conducted another ablation study with the BS-ERGB dataset, which has actual event information from a real event-based camera.

Table 4.2 shows the results of our ablation study. (Note that the last row of the table is taken from [89]). The results we obtained are close to the results presented in TimeLens++, and the effect of synthesis and warping refinement blocks are similar to the ablation presented in TimeLens. From these tables, we argue that the event information is not very well utilized as the warping and synthesis blocks have almost similar effects on the output.

Table 4.2: Our ablation study conducted on BS-ERGB dataset with image sizes 256×256.

Module	**PSNR**	**SSIM**
Interpolation by synthesis	28.04	0.9223
Warping refinement	28.24	0.9195
TimeLens	28.63	0.9223
TimeLens [89]	28.56	-

Another deduction we have made from TimeLens and TimeLens++ is the influence of image size on algorithm performance. The BS-ERGB dataset is presented in the work TimeLens++. Although the TimeLens++ codes are not provided, the TimeLens inference codes can be accessed on the TimeLens official website. The PSNR score of TimeLens on the BS-ERGB dataset in TimeLens++ is $28,56db$. However, when we run the same code for a full-sized 970×625 image, the score drops to $23,97dB$. Knowing this fact, we have conducted our experiments both with 256×256 images and full-sized images.

We have reached several conclusions by examining the structure and the experiments presented by TimeLens. The first observation is the usage of event voxel representation in a deep neural network. TimeLens uses event voxels efficiently to be fed into warping and synthesis blocks. Secondly, TimeLens chose the number of grids in the voxel grids as 5. Considering the performance of TimeLens with the other state-of-the-art VFI algorithms, we see that TimeLens does not improve accuracies too much. We have concluded that this inefficiency might be related to the number of grids in voxels and should be investigated. Finally, the ablation study of TimeLens has shown that the warping and the synthesis-based approaches perform similarly and may not be the best way to utilize the event information for the VFI task.

CHAPTER 5

PROPOSED METHOD

5.1 Motivation

Previous works have shown that using event-based information can improve video frame interpolation performance. The two crucial important factors for the effectiveness of event-based cameras on video frame interpolation are the following:

- **Low Latency of Event-based Cameras:** The event-based cameras have very low latency; that is, each pixel might transmit new information in order of milliseconds. As event-based cameras receive information quite fast, event cameras can also fill the gap between the keyframes. Ordinarily, for the standard, the information between the keyframes of a video frame interpolation task can only be retrieved by some assumptions about the movement of the object. Event-based cameras, with their low latency, give valuable information on the movements of objects.

- **High Dynamic Range of Event-based Cameras:** When compared to standard cameras, event-based cameras have a higher dynamic range. In general, high-quality traditional cameras have a dynamic range of 60 dB; however, most event-based cameras have a dynamic range of 120 dB. Therefore, event-based cameras also supply information for the invisible places to standard cameras, which helps to interpolate frames even in extreme lighting conditions.

5.2 Key Observations

In order to benefit from the superior attributes of event-based cameras, we aim to extract information from the sequential event-based information and combine it with standard images. The possible advantages of using event-based cameras on video-frame interpolation can be listed as follows:

- With their high-speed sensors, event cameras provide information about the deadtime between keyframes that will reveal the motion of objects. Combined with standard camera outputs, they can significantly increase video frame interpolation performance significantly.
- Under challenging lighting conditions, the movements of the objects will be more apparent to event-based cameras as they have a high dynamic range. With the help of ordinary frame information, interpolated images can have even better qualities than standard images.

5.3 Proposed Architecture

Based on the presented literature in the previous chapters, the following critical conclusions are deducted after examining the state-of-the-art video frame interpolation algorithms, TimeLens [90] and Video Frame Interpolation Transformer [84].

- TimeLens uses event information as voxel grids, which enables events to be fed into a standard neural network. Heuristically, we observe from Figure 2.2 that using a voxel grid successfully extracts information from events. This image shows the events between the 4 time slots divided into a 256-segment voxel grid. However, TimeLens renders voxel grids only five segments long. Figure 2.3 shows a similar rendering example. Using small sized voxel grids, causes the precious timestamp information in the events to be lost.
- TimeLens aims to use event information with synthesis and warping-based algorithms. In the ablation study, we observed that the results of the two techniques are similar to each other, and their effects on the results of the original

algorithm are close. This shows that the information inside the events yields similar results when extracted with both methods.

- TimeLens gives the results on the Vimeo90k [99] dataset by two different methods. The former was trained with completely synthetic events, and the latter network was trained with synthetic events and then fine-tuned with actual events and frames collected with DAVIS. The output PSNR values are $30.57dB$ and $32.49dB$, respectively. The difference between the performances shows that it is essential to use actual event-based camera outputs to train a network that utilizes events.

- The results of TimeLens, which uses both events and keyframes, and Video Frame Interpolation Transformer, which only uses keyframes, on the Vimeo90K dataset are controversial. TimeLens, which has information about the deadtime between keyframes, has a PSNR of $32.49dB$. In contrast, Video Frame Interpolation Transformer has a PSNR of $36.96dB$. This result indicates that the event information can be utilized more efficiently to construct intermediate frames.

- When we examined the Video Frame Interpolation Transformer, we understood that the purpose of using the encoder-decoder structure is to predict the movement between two key-frames. We questioned whether this network is quite important when we have events.

- In the ablation study given by Video Frame Interpolation Transformer, we saw that the performance decreased from $36.02dB$ to $35.82dB$ when the SepSTS blocks currently used in the Encoder-Decoder structure were removed and replaced with Resblock [32]. We deduce from these experiments that the main reason for the high performance of Video Frame Interpolation Transformer is not the encoder-decoder structure used at the beginning but the SynBlocks [97] used in the last part.

By the help of these observations, we designed our algorithm, *Event Attention Video Frame Interpolation*. The general structure of our algorithm can be seen in Figure 5.1. We input 4 images, I_{-2}, I_{-1}, I_1, I_2, that are captured by a standard camera. Additionally, we take the actual events related to these images' four time periods and

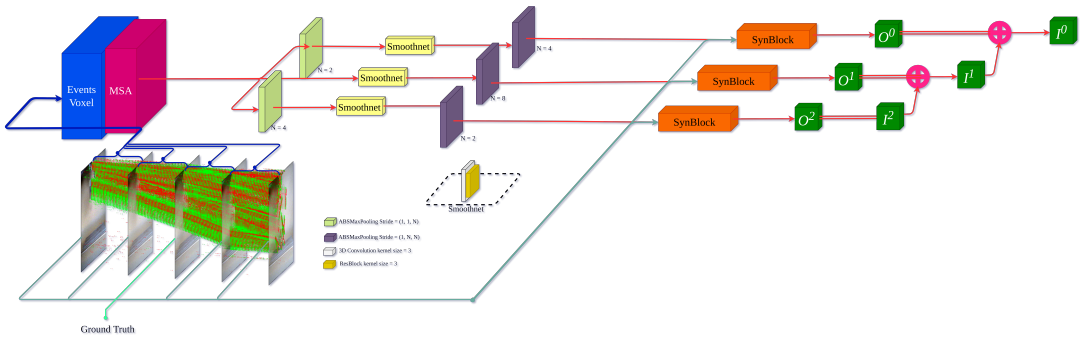


Figure 5.1: Events given to MSA and max pooling blocks.

construct voxel grids of length 256. The first event interval $E_{-2\to-1}$ and second event interval $E_{-1\to0}$ are directly taken, but the third $E_{0\to1}$,and fourth $E_{1\to2}$ event intervals are reversed in time and polarity. Note that the aim is to construct the intermediate frame I_0.

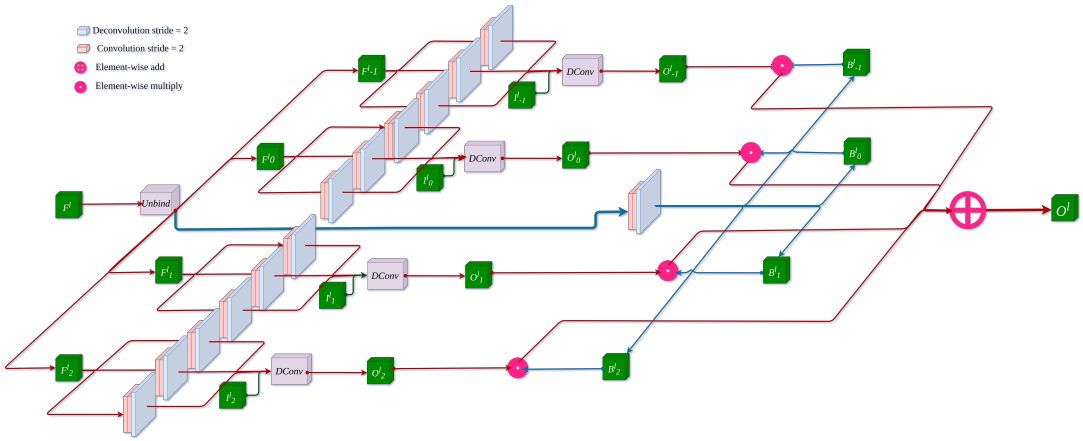


Figure 5.2: SynBlock architecture.

The proposed algorithm can be divided into two stages: The first stage consists of two pooling layers and SmoothNet blocks. We first gave the event voxel representation into a multi-head self-attention block with 16 heads. Then we used a max pooling operation to extract all of the information from the voxel grid representations temporal domain. We have decreased the temporal domain length by 2 and 4 and fed the downscaled and the original voxel representations into SmoothNets. SmoothNets consists of three 3D convolutional layers. These blocks are used for the spatial smoothness of event feature vectors before the second (spatial) pooling operation. For the spatial pooling operation, we have directly used decimation blocks. The downscaling pa-

rameters of the first and the second pooling operations are selected so that the result gives 8 when multiplied.

The output of the second pooling layer is given to the second stage of our algorithm. This stage fuses the events with the frames that are taken from the standard camera by using Synthesis Blocks (SynBlocks) adapted from [84]. There are three SynBlock in the second stage of our network. The operations of these SynBlocks are identical, but they work on images with different spatial sizes.

Each SynBlock outputs an intermediate frame downscaled with $l \in 0, 1, 2$. The outputs of these three SynBlocks are combined to create the final output of our algorithm, which is presented in equation 5.1 where f_{up} stands for bilinear upscaling, and O^l is an output of a SynBlock. This downscaling/upscaling structure, as shown in Figure 5.1, helps the network learn about fast-moving objects better.

$$\hat{I}_0^l = f_{up}(\hat{I}_0) + O^l \tag{5.1}$$

Now let us consider the inside of SynBlocks in this general structure for which we have followed [84]. The input part of the SynBlocks starts with the unbinding operation. In other words, the feature vectors F^l are divided into four in the temporal dimension. Each of these four temporal parts passes three parallel convolutional blocks generating four sisters of convolutional kernels. Each convolutional block that is constructing these convolutional kernels is created by 2D convolutions operating over event feature maps that produce deformable kernels for images. In other words, these three kernels are weights $W_t^l \in R^{KxHxW}$, horizontal offsets, $\alpha_t^l \in R^{KxHxW}$,and vertical offsets $\beta_t^l \in R^{KxHxW}$, the K stands for the number of sampling locations of every kernel (An example sampling can $(\hat{u}_i, \hat{v}_i) = \{(-1,-1), ..., (0,0), ..., (1,1)\}$). Using these predicted kernels, the result of the deformable [97] convolution is reached as given in equation 5.2.

$$O_t^l(x,y) = \sum_{n=1}^{K} W_t^l(n,x,y) I_t^l(x + \alpha_t^l(n,x,y), y + \beta_t^l(n,x,y)) \tag{5.2}$$

Thus, RGB frames are fused with the feature vectors coming from convolutional

blocks by deformable convolution blocks introduced by [19] extended by [42] with shareable weights. At the last phase of a SynBlock, O_t^l is blended with learned masks. The masks are acquired by using standard convolutional neural networks on the feature map, which is the concatenation of the SynBlock input. The result of a SynBlock, which is represented in Figure 5.2, is finally equated as given in equation 5.3.

$$O^l = \sum_t B_t^l \bullet O_t^l \tag{5.3}$$

5.3.1 Training Details

While training our network, we have used Adamax optimizer with $beta_1 = 0.9$ and $beta_2 = 0.999$, L_1 loss and 6 batch size. We started the learning rate as 0.0008 and halved it every eight epochs. We have used 2 RTX2080Ti GPUs to train our network, for which one epoch took about 3 hours. For every network that we have proposed, we have trained it for 24 epochs at least. The final PSNR and SSIM scores are obtained after 26 epochs.

5.4 Experimental Results

To compare the results of our proposed method against the state-of-the-art video frame interpolation techniques, we have conducted experiments on the BS-ERGB dataset. We selected this dataset mainly because it contains events from an actual event-based camera and images from a standard camera with relatively high resolutions. Additionally, compared with other datasets like Vimeo90K, the BS-ERGB dataset contains challenging image sequences that can cause algorithms using only frames to struggle.

While performing our experiments, we have seen that changing the image sizes can affect the PSNR values significantly. Therefore, we separated the results for full-sized images and images downscaled to 256×256.

5.4.1 Test Results

Our overall test results are summarized in Tables 5.1 and 5.2 for downscaled and full-sized images, respectively. Notice that the performance of TimeLens++ is left empty in Table 5.2. This is because the source or the inference code of this algorithm is not shared.

Table 5.1: Comparison of the proposed method in BS-ERGB dataset in low resolution.

Method	**Input**	**Number of Parameters (M)**	**PSNR (dB)**	**SSIM**
FLAVRFV [37]	Frames	42.4	31.72	0.9469
DAIN [6]	Frames	24.0	21.40	TBA
VFIT [84]	Frames	29.0	32.08	0.9449
Timelens [90]	Frames and Events	72.2	28.36	TBA
Timelens++ [89]	Frames and Events	53.9	28.56	-
Proposed	Frames and Events	**1.8**	**32.23**	**0.9581**

Table 5.2: Comparison of the proposed method in BS-ERGB dataset in full scale.

Method	**Input**	**Number of Parameters (M)**	**PSNR (dB)**	**SSIM**
FLAVRFV [37]	Frames	42.4	27.642	0.8729
DAIN [6]	Frames	24.0	TBA	TBA
VFIT [84]	Frames	29.0	28.00	0.8767
Timelens [90]	Frames and Events	72.2	23.97	TBA
Timelens++ [89]	Frames and Events	53.9	-	-
Proposed	Frames and Events	**1.8**	**29.04**	**0.8771**

Figures 5.3 and 5.4 show the parameters versus PSNR graphs. The former shows the results for down-scaled images, and the latter shows the output for full sized images.

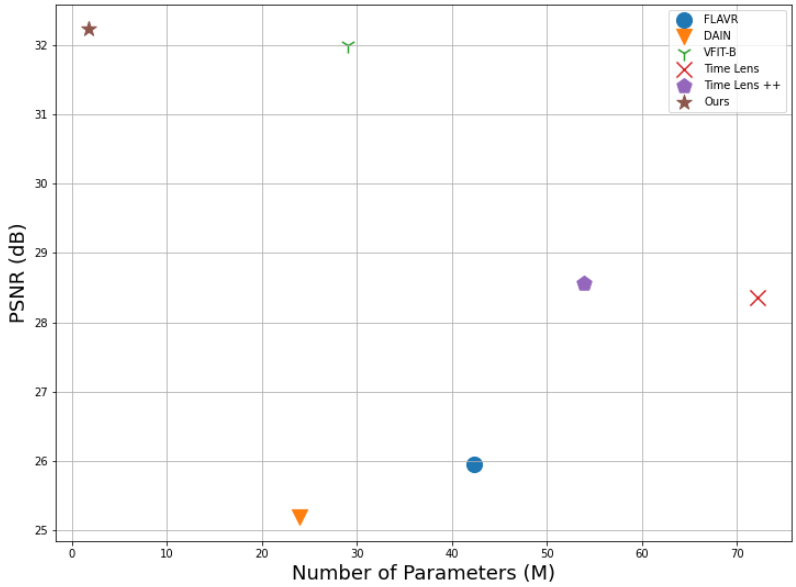


Figure 5.3: Number of parameters versus PSNR graph for down scaled images.

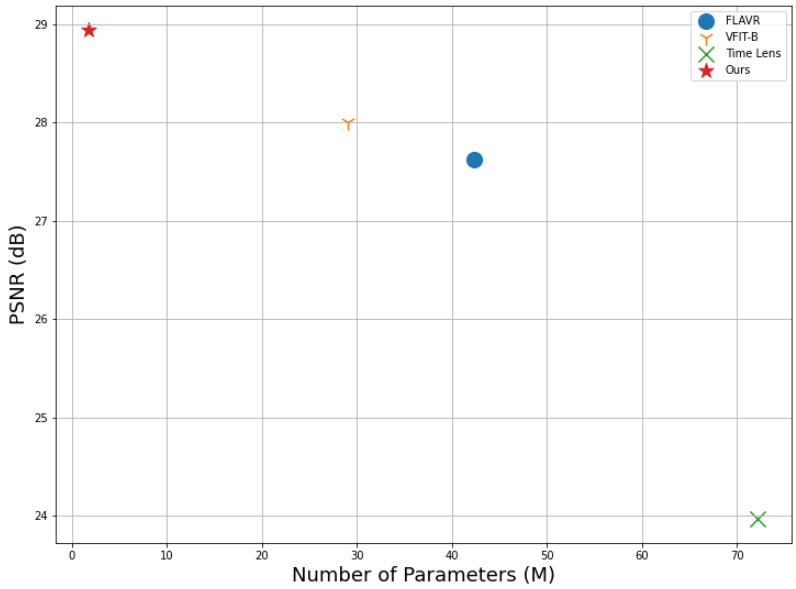


Figure 5.4: Number of parameters versus PSNR graph for full scaled images.

5.4.2 Ablation Studies

We have conducted a series of ablation studies to legitimize the proposed strategy. First, we discarded the multi-head self-attention mechanism at the input stage of the network. Then we have made use of Sep-STS layers proposed in [84] in the input stage of the network. Finally, we have analyzed the effect of voxel-grid size on the performance of the proposed algorithm.

As mentioned, event data already provides a piece of information about how the objects move in the scene. We have trained a network with no attention mechanism (MSA) in the input stage. The sizes of every other block in the network were kept the same, and the network was trained. The quantitative results are given in Table 5.3.

Table 5.3: Ablation study for multi-head self-attention.

Method	**PSNR (dB)**	**SSIM**
Without MSA	31.64	0.9496
Proposed	**32.23**	**0.9581**

Sep-STS blocks is first introduced in [84] for encoder part of the network from [42]. Sep-STS blocks are discrete transformer structures that partition spatial and transient aspects for memory optimization purposes. We have changed the voxel grid size and fed the voxel grids to the encoder and images to SynBlocks directly. Then we tested the network, which can be seen in Figure 5.5. The qualitative results can be seen in Table 5.4.

Table 5.4: Ablation study for the first stage of our proposed network.

Method	**PSNR (dB)**	**SSIM**
Sep-STS Encoder	31.19	0.9436
Proposed	**32.23**	**0.9581**

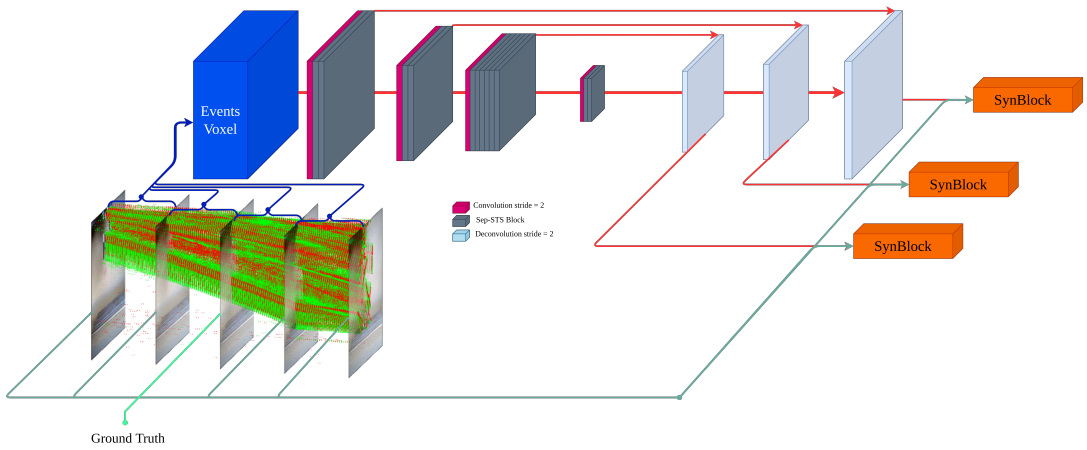


Figure 5.5: Events given to SepSTS encoder block.

5.4.3 Discussion

Our experiments show that the proposed method is superior to state-of-the-art video frame interpolation algorithms. For images with size 256×256 our method increases PSNR by $0.15dB$ and SSIM by 0.0132. For the full-scaled images of size 970×625, it increases PSNR by $1.04dB$ and SSIM by 0.004. Additionally, our network is very lightweight when compared with state-of-the-art methods. Video Frame Interpolation Transformer has the closest scores to our proposed method and has 29M parameters. On the other hand, our network has only 1.8M parameters.

VFI results that are related to our preliminary experiments, which are conducted using the available event-based camera, indicate that the timestamps between the events and images should be as precise as possible. In other words, synchronization of timestamps between two cameras by means of an imprecise, approximate approach yields interpolated images with artifacts. In order to synchronize the timestamps of these two cameras, a hardware implementation should be preferred.

We have investigated the performance of voxel grid size on the VFI performance. For this purpose, we have conducted experiments with voxel-grid sizes of $16, 32, 64, 128$ and 0.0008 learning rate. The performance results of these experiments yield $32.44, 32.05, 32.39, and 32.23$. After obtaining a higher performance with a 16 voxel grid size, we have decided to increase the learning rate to 0.0016 to see the effect of hyperparameters on the performance. The performance for voxel-grid size 128 has increased to 32.56 from 32.23.

Therefore, our experiments with different-sized voxel grids have shown that using a larger grid size increases the VFI performance if proper hyperparameter tuning is performed.

Considering our ablation study, the MSA layer that is used as the first layer in our network increases the PSRN and SSIM performance by 0.59 and 0.0085, respectively. This is because the temporal attention mechanism helps the network to focus on critical temporal dimensions of our voxel grids.

The ablation study on our max pooling and attention layers show that the SepSTS blocks used to extract temporal and spatial information do not work as well as our proposed method. Our technique boosts PSNR by 1.04 and SSIM by 0.0145.

Figures 5.6, 5.7, and 5.8 compare the qualitative results of TimeLens, Video Frame Interpolation Transformer and our method. Our method is superior to the others in scenarios where there are objects that have nonlinear and fast trajectories.

(a) Ground truth image.

(b) Our algorithms output.

(c) VFIT output.

(d) TimeLens Output.

Figure 5.6: Comparison of 5.6a ground truth image, 5.6d TimeLens [90], 5.6c Video Frame Interpolation Transformer [84] and 5.6b our result for a ball sequence.

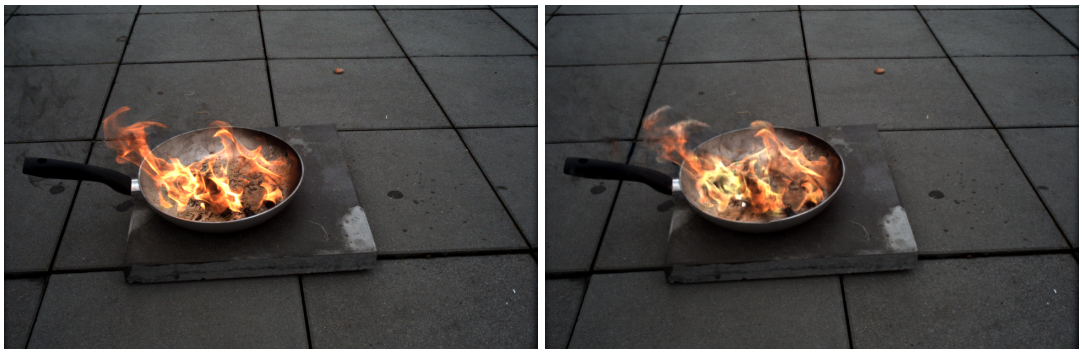
(a) Ground truth image.

(b) Our algorithms output.

(c) VFIT output.

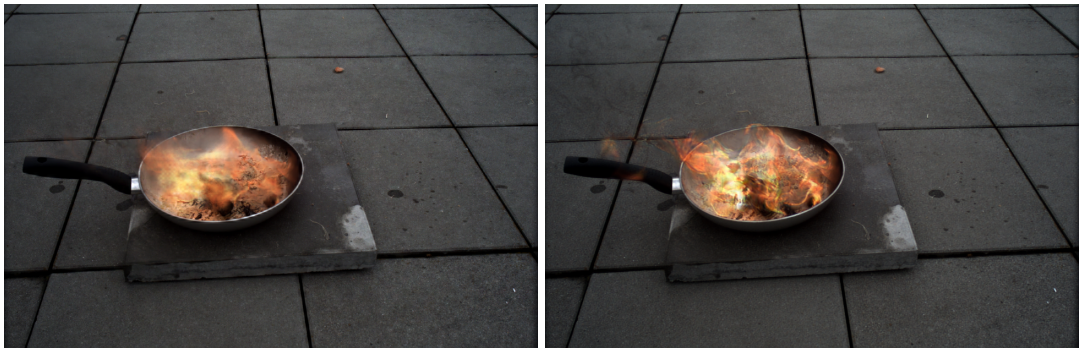
(d) TimeLens Output.

Figure 5.7: Comparison of 5.7a ground truth image, 5.7d TimeLens [90], 5.7c Video Frame Interpolation Transformer [84] and 5.7b our result for an elastic band sequence.



(a) Ground truth image.

(b) Our algorithms output.



(c) VFIT output.

(d) TimeLens Output.

Figure 5.8: Comparison of 5.8a ground truth image, 5.8d TimeLens [90], 5.8c Video Frame Interpolation Transformer [84] and 5.8b our result for a fire sequence.

CHAPTER 6

CONCLUSIONS & FUTURE WORKS

6.1 Conclusions

In this thesis, we aimed to combine events from an event-based camera and images from a standard imaging device for video frame interpolation. To be able to extract as much information as possible from the events, we have searched for different event representations and their performance on neural network architectures. Based on the experimental results, event voxel representation is found the be the most suitable method to represent events when the grid size is chosen sufficiently enough. Event voxels are eligible to be utilized in deep neural networks and are easy to construct.

We examined various state-of-the-art video frame interpolation algorithms and based our method on them. One of the most informative papers in the literature was Video Frame Interpolation Transformer, which proved to be one of the best algorithms for VFI. It constructs intermediate frames using only keyframes. VFIT utilizes an encoder-decoder architecture that uses SepSTS blocks to obtain spatial and temporal information about the intermediate frames. We investigated this encoder-decoder architecture's necessity in the presence of events and found that it is unnecessary. Additionally, VFIT uses SynBlocks to fuse information obtained from the encoder-decoder block and the keyframes. We observed the eligibility of this algorithm to combine event-based information and the keyframes.

We also examined TimeLens, which combines the information from event-based and standard cameras. TimeLens uses both warping and synthesis-based approaches for VFI. Comparing the results of TimeLens and VFIT, we have seen that the event information could be used more efficiently by means of a different methodology. More-

over, considering the training details and the ablation studies of TimeLens, we have observed that the usage of a dataset which contains actual event-based information is quite crucial for the performance of the network. Therefore, we have used the BS-ERGB dataset, which contains relatively high-resolution real event information.

With the help of our literature observations, we have proposed Event Attention Video Frame Interpolation (EA-VFI), which effectively combines events and keyframes. We have utilized multi-head self-attention layers to extract temporal information from the event voxels and SynBlocks to fuse event-based information and keyframes. Experimental results have shown that the proposed method is superior to state-of-the-art video frame interpolation techniques in PSNR and SSIM.

To sum up, we have utilized event information for video frame interpolation, analyzed state-of-the-art VFI methods, and proposed a network that works outstanding in challenging scenarios.

6.2 Future Works

Looking at the results of our proposed method, we see that fast-moving objects sometimes lose color information. This result is expected as the event-based cameras do not give information on the RGB scale colour change but only provide information on the change of the scene. This problem may be handled with another fusion algorithm that will be used before the SynBlocks. Furthermore, by predicting kernels tied to different time steps, our algorithm can be easily extended to multi-frame interpolation or even arbitrary-time interpolation by taking time as an extra input.

APPENDIX A

ARCHITECTURE UTILIZED IN VIDEO FRAME INTERPOLATION

A.1 Vision Transformers

Transformer [91], an attention-based encoder-decoder, has already redefined natural language processing (NLP). Inspired by such significant achievements, some pioneering work has recently been done in the computer vision (CV) field on employing Transformer-liked architectures, which have demonstrated their effectiveness on fundamental CV tasks (classification, detection, and segmentation). Visual Transformers have shown substantial performance increases over numerous benchmarks due to their competitive modeling capabilities when compared to current Convolution Neural Networks (CNNs). Although this idea is not very commonly used in video frame interpolation, it can also be utilized for this task.

A.1.1 Multi-Head Self/Cross Attention

The most critical structure of an attention mechanism is the (multi-head) self/cross attention mechanism.

A single-head attention mechanism takes two input vectors, $X \in \mathbb{R}^{n_x \times d_x}$, $Y \in \mathbb{R}^{n_y \times d_y}$. These two vectors are then put through three linear layers to obtain three fundamental vectors, key (K), query (Q), and value (V). This is achieved by multiplying the vectors X and Y with three matrices $W^Q \in \mathbb{R}^{d_x \times d^k}$, $W^K \in \mathbb{R}^{d_y \times d^k}$, and $W^V \in \mathbb{R}^{d_y \times d^v}$ as given in equation A.1. Then, the vectors key and query dot product are found and scaled by a predetermined number. Then, this number is given to a softmax function, and it is multiplied by the vector V which can be found in equation A.2.

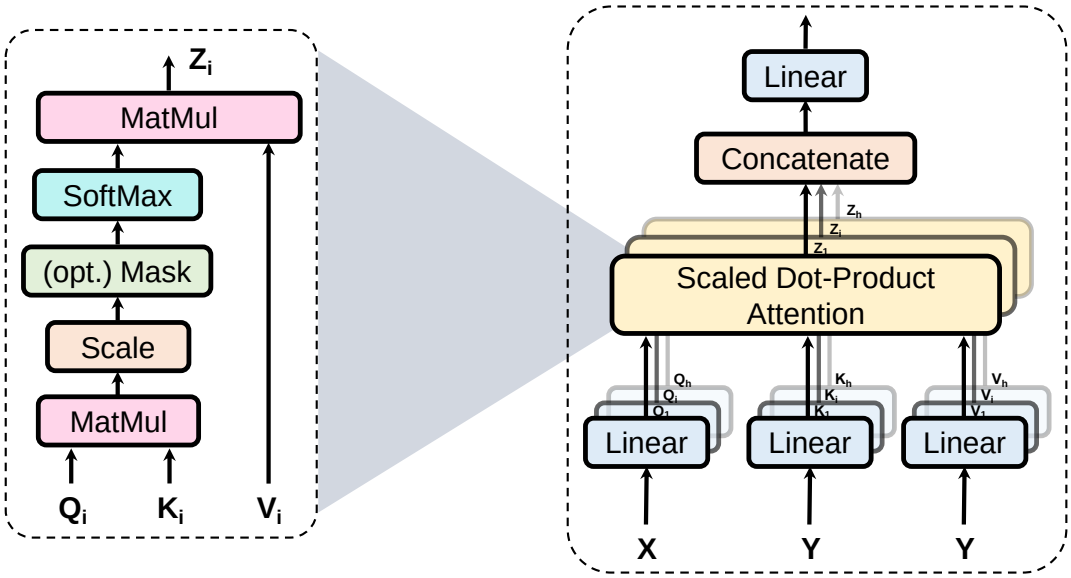


Figure A.1: The structure of the attention mechanism. Left: Scaled Dot-Product Attention. Right: Multi-Head Attention Mechanism [51].

$$Q = XW^Q, \quad K = YW^K, \quad V = YW^V, \tag{A.1}$$

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{A.2}$$

The equations A.1 and A.2 is actually stands for cross attention which aims to find correlation between two different vectors. These equations can be turned into self-attention by simply taking $Y = X$. In this case, the attention mechanism will focus more on the elements that represent the input arrays. This attention mechanism can simply be extended to multi head attention by using multiple $W^Q \in \mathbb{R}^{d_x \times d^k}$, $W^K \in \mathbb{R}^{d_y \times d^k}$, and $W^V \in \mathbb{R}^{d_y \times d^v}$ matrices. This type of attention mechanisms are presented in Figure A.1 and the formulation can be found in equation A.3.

$$\begin{aligned} &Q_i = XW^{Q_i},\ K_i = XW^{K_i},\ V_i = XW^{V_i}, \\ &Z_i = \text{Attention}(Q_i, K_i, V_i),\ i = 1 \ldots h, \\ &\text{MultiHead}(Q, K, V) = \text{Concat}(Z_1, Z_2, ..., Z_h)W^O, \end{aligned} \tag{A.3}$$

In equation A.3, where h is the head number, $W^O \in \mathbb{R}^{hd_v \times d_{model}}$ denotes the out-

put projected matrix, Z_i denotes the output vector of each head, $W^{Q_i} \in \mathbb{R}^{d_{model} \times d_k}$, $W^{K_i} \in \mathbb{R}^{d_{model} \times d_k}$, and $W^{V_i} \in \mathbb{R}^{d_{model} \times d_v}$ are three different groups of matrices. With the help of standard backpropagation algorithms, the weights are adjusted for the specific task.

REFERENCES

[1] Enhancing and experiencing spacetime resolution with videos and stills. In *ICCP*, pages 1–9. IEEE, 2009.

[2] S. American. The silicon retina. *Scientific American*, 264(5):76, 83, May 1991.

[3] A. Amir, B. Taba, D. Berg, T. Melano, J. McKinstry, C. Di Nolfo, T. Nayak, A. Andreopoulos, G. Garreau, M. Mendoza, et al. A low power, fully event-based gesture recognition system. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7243–7252, 2017.

[4] R. Baldwin, M. Almatrafi, J. R. Kaufman, V. Asari, and K. Hirakawa. Inceptive event time-surfaces for object classification using neuromorphic cameras. In *International conference on image analysis and recognition*, pages 395–403. Springer, 2019.

[5] R. Baldwin, R. Liu, M. M. Almatrafi, V. K. Asari, and K. Hirakawa. Time-ordered recent event (tore) volumes for event cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[6] W. Bao, W.-S. Lai, C. Ma, X. Zhang, Z. Gao, and M.-H. Yang. Depth-aware video frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3703–3712, 2019.

[7] P. Bardow, A. J. Davison, and S. Leutenegger. Simultaneous optical flow and intensity estimation from an event camera. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 884–892, 2016.

[8] F. Barranco, C. Fermuller, Y. Aloimonos, and T. Delbruck. A dataset for visual navigation with neuromorphic methods. *Frontiers in neuroscience*, 10:49, 2016.

[9] R. Benosman, C. Clercq, X. Lagorce, S.-H. Ieng, and C. Bartolozzi. Event-based visual flow. *IEEE transactions on neural networks and learning systems*, 25(2):407–417, 2013.

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

[11] J. Cao, Y. Li, K. Zhang, and L. Van Gool. Video super-resolution transformer. *arXiv preprint arXiv:2106.06847*, 2021.

[12] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko. End-to-end object detection with transformers. In *Proceedings of the European Conference on Computer Vision*, 2020.

[13] H. Chen, Y. Wang, T. Guo, C. Xu, Y. Deng, Z. Liu, S. Ma, C. Xu, C. Xu, and W. Gao. Pre-trained image processing transformer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

[14] Z. Chi, R. M. Nasiri, Z. Liu, J. Lu, J. Tang, and K. N. Plataniotis. All at once: Temporally adaptive multi-frame interpolation with advanced motion modeling. *arXiv preprint arXiv:2007.11762*, 2020.

[15] T.-J. Chin, S. Bagchi, A. Eriksson, and A. Van Schaik. Star tracking using an event camera. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

[16] G. Cohen, S. Afshar, B. Morreale, T. Bessell, A. Wabnitz, M. Rutten, and A. van Schaik. Event-based sensing for space situational awareness. *The Journal of the Astronautical Sciences*, 66(2):125–141, 2019.

[17] J. Conradt, M. Cook, R. Berner, P. Lichtsteiner, R. J. Douglas, and T. Delbruck. A pencil balancing robot using a pair of aer dynamic vision sensors. In *2009 IEEE International Symposium on Circuits and Systems*, pages 781–784. IEEE, 2009.

[18] M. Cook, L. Gugelmann, F. Jug, C. Krautz, and A. Steger. Interacting maps for fast visual interpretation. In *The 2011 International Joint Conference on Neural Networks*, pages 770–776. IEEE, 2011.

[19] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. Deformable convolutional networks, 2017.

[20] T. Delbruck. Neuromorphic vision sensing and processing. In *2016 46Th european solid-state device research conference (ESSDERC)*, pages 7–14. IEEE, 2016.

[21] T. Delbruck and M. Lang. Robotic goalie with 3 ms reaction time at 4% cpu load using event-based dynamic vision sensor. *Frontiers in neuroscience*, 7:223, 2013.

[22] T. Delbruck and P. Lichtsteiner. Fast sensory motor control based on event-based hybrid neuromorphic-procedural system. In *2007 IEEE international symposium on circuits and systems*, pages 845–848. IEEE, 2007.

[23] J. Delmerico, T. Cieslewski, H. Rebecq, M. Faessler, and D. Scaramuzza. Are we ready for autonomous drone racing? the uzh-fpv drone racing dataset. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6713–6719. IEEE, 2019.

[24] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proceedings of the International Conference on Learning Representations*, 2020.

[25] G. Gallego, T. Delbrück, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. J. Davison, J. Conradt, K. Daniilidis, et al. Event-based vision: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(1):154–180, 2020.

[26] G. Gallego, J. E. Lund, E. Mueggler, H. Rebecq, T. Delbruck, and D. Scaramuzza. Event-based, 6-dof camera tracking from photometric depth maps. *IEEE transactions on pattern analysis and machine intelligence*, 40(10):2402–2412, 2017.

[27] G. P. García, P. Camilleri, Q. Liu, and S. Furber. pydvs: An extensible, real-time dynamic vision sensor emulator using off-the-shelf hardware. In *2016*

IEEE Symposium Series on Computational Intelligence (SSCI), pages 1–7. IEEE, 2016.

[28] D. Gehrig, A. Loquercio, K. G. Derpanis, and D. Scaramuzza. End-to-end learning of representations for asynchronous event-based data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5633–5643, 2019.

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

[30] A. Glover and C. Bartolozzi. Event-driven ball detection and gaze fixation in clutter. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2203–2208. IEEE, 2016.

[31] G. Haessig, F. Galluppi, X. Lagorce, and R. Benosman. Neuromorphic networks on the spinnaker platform. In *2019 IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, pages 86–91. IEEE, 2019.

[32] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

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

[34] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *CVPR*, pages 2462–2470, 2017.

[35] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In *NIPS*, pages 2017–2025, 2015.

[36] H. Jiang, D. Sun, V. Jampani, M.-H. Yang, E. Learned-Miller, and J. Kautz. Super slomo: High quality estimation of multiple intermediate frames for video interpolation. In *CVPR*, pages 9000–9008, 2018.

[37] T. Kalluri, D. Pathak, M. Chandraker, and D. Tran. Flavr: Flow-agnostic video representations for fast frame interpolation. *arXiv preprint arXiv:2012.08512*, 2020.

[38] M. L. Katz, K. Nikolic, and T. Delbruck. Live demonstration: Behavioural emulation of event-based vision sensors. In *2012 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 736–740. IEEE, 2012.

[39] H. Kim, A. Handa, R. Benosman, S.-H. Ieng, and A. J. Davison. Simultaneous mosaicing and tracking with an event camera. *J. Solid State Circ*, 43:566–576, 2008.

[40] H. Kim, S. Leutenegger, and A. J. Davison. Real-time 3d reconstruction and 6-dof tracking with an event camera. In *European conference on computer vision*, pages 349–364. Springer, 2016.

[41] P. Kirkland, G. Di Caterina, J. Soraghan, and G. Matich. Neuromorphic technologies for defence and security. In *Emerging Imaging and Sensing Technologies for Security and Defence V; and Advanced Manufacturing Technologies for Micro-and Nanosystems in Security and Defence III*, volume 11540, pages 113–130. SPIE, 2020.

[42] H. Lee, T. Kim, T.-y. Chung, D. Pak, Y. Ban, and S. Lee. Adacof: Adaptive collaboration of flows for video frame interpolation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[43] J. H. Lee, T. Delbruck, M. Pfeiffer, P. K. Park, C.-W. Shin, H. Ryu, and B. C. Kang. 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.

[44] P. Lichtsteiner. 64x64 event-driven logarithmic temporal derivative silicon retina. In *Program 2003 IEEE Workshop on CCD and AIS*, 2003.

[45] P. Lichtsteiner. *An AER temporal contrast vision sensor*. PhD thesis, ETH Zurich, 2006.

[46] P. Lichtsteiner and T. Delbruck. A 64x64 aer logarithmic temporal derivative silicon retina. In *Research in Microelectronics and Electronics, 2005 PhD*, volume 2, pages 202–205. IEEE, 2005.

[47] P. Lichtsteiner, C. Posch, and T. Delbruck. A 128 x 128 120db 30mw asynchronous vision sensor that responds to relative intensity change. In *2006 IEEE International Solid State Circuits Conference-Digest of Technical Papers*, pages 2060–2069. IEEE, 2006.

[48] P. Lichtsteiner, C. Posch, and T. Delbruck. A 128× 128 120 db 15 µs latency asynchronous temporal contrast vision sensor. *Solid-State Circuits, IEEE Journal of*, 43:566 – 576, 03 2008.

[49] M. Litzenberger, B. Kohn, A. N. Belbachir, N. Donath, G. Gritsch, H. Garn, C. Posch, and S. Schraml. Estimation of vehicle speed based on asynchronous data from a silicon retina optical sensor. In *2006 IEEE intelligent transportation systems conference*, pages 653–658. IEEE, 2006.

[50] S.-C. Liu, B. Rueckauer, E. Ceolini, A. Huber, and T. Delbruck. Event-driven sensing for efficient perception: Vision and audition algorithms. *IEEE Signal Processing Magazine*, 36(6):29–37, 2019.

[51] Y. Liu, Y. Zhang, Y. Wang, F. Hou, J. Yuan, J. Tian, Y. Zhang, Z. Shi, J. Fan, and Z. He. A survey of visual transformers. *arXiv preprint arXiv:2111.06091*, 2021.

[52] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE International Conference on Computer Vision*, 2021.

[53] Z. Liu, J. Ning, Y. Cao, Y. Wei, Z. Zhang, S. Lin, and H. Hu. Video swin transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3202–3211, 2022.

[54] Z. Liu, R. A. Yeh, X. Tang, Y. Liu, and A. Agarwala. Video frame synthesis using deep voxel flow. In *ICCV*, pages 4463–4471, 2017.

[55] A. Mehonic and A. J. Kenyon. Brain-inspired computing needs a master plan. *Nature*, 604(7905):255–260, 2022.

[56] S. Meyer, A. Djelouah, B. McWilliams, A. Sorkine-Hornung, M. Gross, and C. Schroers. Phasenet for video frame interpolation. In *CVPR*, 2018.

[57] S. Miao, G. Chen, X. Ning, Y. Zi, K. Ren, Z. Bing, and A. Knoll. Neuromorphic vision datasets for pedestrian detection, action recognition, and fall detection. *Frontiers in neurorobotics*, 13:38, 2019.

[58] E. Mueggler, G. Gallego, H. Rebecq, and D. Scaramuzza. Continuous-time visual-inertial odometry for event cameras. *IEEE Transactions on Robotics*, 34(6):1425–1440, 2018.

[59] E. Mueggler, H. Rebecq, G. Gallego, T. Delbruck, and D. Scaramuzza. The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and slam. *The International Journal of Robotics Research*, 36(2):142–149, 2017.

[60] S. Niklaus and F. Liu. Context-aware synthesis for video frame interpolation. In *CVPR*, pages 1701–1710, 2018.

[61] S. Niklaus and F. Liu. Softmax splatting for video frame interpolation. In *CVPR*, pages 5437–5446, 2020.

[62] S. Niklaus, L. Mai, and F. Liu. Video frame interpolation via adaptive convolution. In *CVPR*, 2017.

[63] S. Niklaus, L. Mai, and F. Liu. Video frame interpolation via adaptive separable convolution. In *ICCV*, 2017.

[64] G. Orchard, C. Meyer, R. Etienne-Cummings, C. Posch, N. Thakor, and R. Benosman. Hfirst: A temporal approach to object recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37(10):2028–2040, 2015.

[65] A. Paliwal and N. K. Kalantari. Deep slow motion video reconstruction with hybrid imaging system. *PAMI*, 2020.

[66] L. Pan, C. Scheerlinck, X. Yu, R. Hartley, M. Liu, and Y. Dai. Bringing a blurry frame alive at high frame-rate with an event camera. In *Proceedings of*

the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 6820–6829, 2019.

[67] F. Paredes-Vallés and G. C. de Croon. Back to event basics: Self-supervised learning of image reconstruction for event cameras via photometric constancy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3446–3455, 2021.

[68] F. Paredes-Vallés, K. Y. Scheper, and G. C. De Croon. Unsupervised learning of a hierarchical spiking neural network for optical flow estimation: From events to global motion perception. *IEEE transactions on pattern analysis and machine intelligence*, 42(8):2051–2064, 2019.

[69] J. Park, K. Ko, C. Lee, and C.-S. Kim. Bmbc: Bilateral motion estimation with bilateral cost volume for video interpolation. *ECCV*, 2020.

[70] C. Posch, D. Matolin, and R. Wohlgenannt. A qvga 143 db dynamic range frame-free pwm image sensor with lossless pixel-level video compression and time-domain cds. *IEEE Journal of Solid-State Circuits*, 46(1):259–275, 2010.

[71] P. W. Power and J. A. Schoonees. Understanding background mixture models for foreground segmentation. In *Proceedings image and vision computing New Zealand*, volume 2002, pages 10–11, 2002.

[72] H. Rebecq, G. Gallego, E. Mueggler, and D. Scaramuzza. Emvs: Event-based multi-view stereo—3d reconstruction with an event camera in real-time. *International Journal of Computer Vision*, 126(12):1394–1414, 2018.

[73] H. Rebecq, T. Horstschäfer, G. Gallego, and D. Scaramuzza. Evo: A geometric approach to event-based 6-dof parallel tracking and mapping in real time. *IEEE Robotics and Automation Letters*, 2(2):593–600, 2016.

[74] H. Rebecq, R. Ranftl, V. Koltun, and D. Scaramuzza. Events-to-video: Bringing modern computer vision to event cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3857–3866, 2019.

[75] H. Rebecq, R. Ranftl, V. Koltun, and D. Scaramuzza. High speed and high dynamic range video with an event camera. *IEEE transactions on pattern analysis and machine intelligence*, 43(6):1964–1980, 2019.

[76] C. Reinbacher, G. Munda, and T. Pock. Real-time panoramic tracking for event cameras. In *2017 IEEE International Conference on Computational Photography (ICCP)*, pages 1–9. IEEE, 2017.

[77] D. A. Reynolds. Gaussian mixture models. *Encyclopedia of biometrics*, 741(659-663), 2009.

[78] C. Richter, F. Röhrbein, and J. Conradt. Bio-inspired optic flow detection using neuromorphic hardware. In *BCCN conference, Göttingen*, 2014.

[79] P. Rogister, R. Benosman, S.-H. Ieng, P. Lichtsteiner, and T. Delbruck. Asynchronous event-based binocular stereo matching. *IEEE Transactions on Neural Networks and Learning Systems*, 23(2):347–353, 2011.

[80] B. Rueckauer and T. Delbruck. Evaluation of event-based algorithms for optical flow with ground-truth from inertial measurement sensor. *Frontiers in neuroscience*, 10:176, 2016.

[81] C. Scheerlinck, N. Barnes, and R. Mahony. Continuous-time intensity estimation using event cameras. In *Asian Conference on Computer Vision*, pages 308–324. Springer, 2018.

[82] C. Scheerlinck, H. Rebecq, D. Gehrig, N. Barnes, R. Mahony, and D. Scaramuzza. Fast image reconstruction with an event camera. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 156–163, 2020.

[83] R. Serrano-Gotarredona, M. Oster, P. Lichtsteiner, A. Linares-Barranco, R. Paz-Vicente, F. Gómez-Rodríguez, L. Camuñas-Mesa, R. Berner, M. Rivas-Pérez, T. Delbruck, et al. Caviar: A 45k neuron, 5m synapse, 12g connects/s aer hardware sensory–processing–learning–actuating system for high-speed visual object recognition and tracking. *IEEE Transactions on Neural networks*, 20(9):1417–1438, 2009.

[84] Z. Shi, X. Xu, X. Liu, J. Chen, and M.-H. Yang. Video frame interpolation transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17482–17491, 2022.

[85] A. Sironi, M. Brambilla, N. Bourdis, X. Lagorce, and R. Benosman. Hats: Histograms of averaged time surfaces for robust event-based object classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1731–1740, 2018.

[86] B. Son, Y. Suh, S. Kim, H. Jung, J.-S. Kim, C. Shin, K. Park, K. Lee, J. Park, J. Woo, et al. 4.1 a 640× 480 dynamic vision sensor with a 9μm pixel and 300meps address-event representation. In *2017 IEEE International Solid-State Circuits Conference (ISSCC)*, pages 66–67. IEEE, 2017.

[87] C. Stauffer and W. E. L. Grimson. Adaptive background mixture models for real-time tracking. In *Proceedings. 1999 IEEE computer society conference on computer vision and pattern recognition (Cat. No PR00149)*, volume 2, pages 246–252. IEEE, 1999.

[88] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *CVPR*, pages 8934–8943, 2018.

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

[90] S. Tulyakov, D. Gehrig, S. Georgoulis, J. Erbach, M. Gehrig, Y. Li, and D. Scaramuzza. TimeLens: Event-based video frame interpolation. *IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

[91] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

[92] A. R. Vidal, H. Rebecq, T. Horstschaefer, and D. 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.

[93] L. Wang, Y.-S. Ho, K.-J. Yoon, et al. Event-based high dynamic range image and very high frame rate video generation using conditional generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10081–10090, 2019.

[94] D. Weikersdorfer, D. B. Adrian, D. Cremers, and J. Conradt. Event-based 3d slam with a depth-augmented dynamic vision sensor. In *2014 IEEE international conference on robotics and automation (ICRA)*, pages 359–364. IEEE, 2014.

[95] D. Weikersdorfer and J. Conradt. Event-based particle filtering for robot self-localization. In *2012 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 866–870. IEEE, 2012.

[96] H. Xu, Y. Gao, F. Yu, and T. Darrell. End-to-end learning of driving models from large-scale video datasets. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2174–2182, 2017.

[97] X. Xu, M. Li, W. Sun, and M.-H. Yang. Learning spatial and spatio-temporal pixel aggregations for image and video denoising. *IEEE Transactions on Image Processing*, 2020.

[98] X. Xu, L. Siyao, W. Sun, Q. Yin, and M.-H. Yang. Quadratic video interpolation. In *NeurIPS*, pages 1647–1656, 2019.

[99] T. Xue, B. Chen, J. Wu, D. Wei, and W. T. Freeman. Video enhancement with task-oriented flow. *IJCV*, 127(8):1106–1125, 2019.

[100] A. Z. Zhu, L. Yuan, K. Chaney, and K. Daniilidis. Ev-flownet: Self-supervised optical flow estimation for event-based cameras. *arXiv preprint arXiv:1802.06898*, 2018.

[101] A. Z. Zhu, L. Yuan, K. Chaney, and K. Daniilidis. Unsupervised event-based learning of optical flow, depth, and egomotion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 989–997, 2019.