

WESTERN SYDNEY
UNIVERSITY



Neuromorphic Perception for Greenhouse Technology Using Event-based Sensors

Sami El Arja

A thesis submitted for the degree of
Master of Philosophy
at
International Centre for Neuromorphic Systems
The MARCS Institute for Brain, Behaviour and
Development
Western Sydney University

May 2022

Supervisors:
A/Prof. Gregory Cohen
Dr. Saeed Afshar
Prof. Gu Fang
© Sami El Arja – 2022

Abstract

Event-Based Cameras (EBCs), unlike conventional cameras, feature independent pixels that asynchronously generate outputs upon detecting changes in their field of view. Short calculations are performed on each event to mimic the brain. The output is a sparse sequence of events with high temporal precision. Conventional computer vision algorithms do not leverage these properties. Thus a new paradigm has been devised.

While event cameras are very efficient in representing sparse sequences of events with high temporal precision, many approaches are challenged in applications where a large amount of spatially-temporally rich information must be processed in real-time. In reality, most tasks in everyday life take place in complex and uncontrollable environments, which require sophisticated models and intelligent reasoning. Typical hard problems in real-world scenes are detecting various non-uniform objects or navigation in an unknown and complex environment. In addition, colour perception is an essential fundamental property in distinguishing objects in natural scenes. Colour is a new aspect of event-based sensors, which work fundamentally differently from standard cameras, measuring per-pixel brightness changes per colour filter asynchronously rather than measuring "absolute" brightness at a constant rate.

This thesis explores neuromorphic event-based processing methods for high-noise and cluttered environments with imbalanced classes. A fully event-driven processing pipeline was developed for agricultural applications to perform fruits detection and classification to unlock the outstanding properties of event cameras. The nature of features in such data was explored, and methods to represent and detect features were demonstrated. A framework for detecting and classifying features was developed and evaluated on the N-MNIST and Dynamic Vision Sensor (DVS) gesture datasets. The same network was evaluated on laboratory recorded and real-world data with various internal variations for fruits detection such as overlap, variation in size and appearance. In addition, a method to handle highly imbalanced data was developed. We examined the characteristics of spatio-temporal patterns for each colour filter to help expand our understanding of this novel data and explored their applications in classification tasks where colours were more relevant features than shapes and appearances.

The results presented in this thesis demonstrate the potential and efficacy of event-based systems by demonstrating the applicability of colour event data and the viability of event-driven classification.

Acknowledgements

First and foremost, I wish to express my deepest gratitude to my supervisor A/Prof. Gregory Cohen for his support and the possibility of working in his group at the International Centre for Neuromorphic Systems. I am genuinely grateful for his numerous and resourceful ideas and suggestions and for providing me with invaluable resources and guidance, especially for his continuous support during all ups and downs of the project. Thanks for not only sharing some of his many ideas with me but giving me the means to pursue mine.

I especially wish to thank my Supervisor, Dr Saeed Afshar, who had a significant influence on my research direction, without whom this work would not have been possible. For the countless hours of coding and debugging, for sharing with me his novel work for demystifying all the complex terms and for the constant stream of ideas and inspiration. Your constructive criticisms and advice have made my research journey a rewarding experience.

Many thanks go to my Supervisor, Prof. Gu Fang, for being incredibly supportive, and for challenging me when I needed challenging, and for supporting me when I needed support, and for being there since day one. His creativity and wisdom were an inspirational part of my research journey.

Collaboration enjoys a very high priority at ICNS. This thesis would not have been possible without my colleagues' help and fruitful discussions. Therefore, I would like to extend my acknowledgement to the countless experts who have offered guidance, feedback, and suggestions along the way - Dr Alexandre Marcireau, who helped me to craft a story about my research and shared with me his philosophical scientific thoughts especially about colours. Dr Damien Joubert for his countless advice and suggestions that helped reshape my research direction with many hours of discussions. Nicholas Ralph for reviewing my thesis and providing me with useful feedback. Dr Bharath Ramesh, and Yeshwanth Bethi for all the valuable discussion and brainstorming that contributed to my work. All put up with a never-ending stream of questions and an unwavering enthusiasm. I could not have completed this work without their support.

Finally, I am profoundly grateful to my parents and to my brother Hani for their unfailing love and support. Thank you

Statement of Authentication

The work presented in this thesis is, to the best of my knowledge and belief, original except as acknowledged in the text.

I hereby declare that I have not submitted this material, either in full or in part, for a degree at this or any other institution.

.....
Sami El Arja May 10, 2022

Contents

List of Tables	iv
List of Figures	vi
List of Abbreviations	xvii
1 INTRODUCTION	1
1.1 Motivation	1
1.2 Aims	3
1.3 Contribution of this Work	3
1.4 Structure of this Thesis	3
2 LITERATURE REVIEW	5
2.1 Colour Sensing and Perception	5
2.2 Frame-based Computer Vision	8
2.2.1 Feature Extraction	8
2.2.2 Object Detection	9
2.3 Event-based Computer Vision	12
2.3.1 Event-Based Sensing	13
2.3.2 Event-based Feature Extraction	16
2.3.3 Event-based Object Detection	17
2.4 Deep Learning for Agriculture Applications	18
3 INVESTIGATION OF NOVEL EVENT-BASED DATA USING THE COLOUR EVENT-BASED VISION SENSOR	22
3.1 Introduction	22
3.2 Research Questions	24
3.3 Contribution	24
3.4 Materials and Methods	24
3.4.1 DVS Colour Pixel: CDAVIS	25
3.4.2 Hardware Assembly	28
3.4.3 Software Stack	29
3.4.4 Colour Pixel Labelling	30
3.4.5 Characterisation Measures	31
3.5 Results	31

3.5.1	Signal to Noise Ratio	31
3.5.2	Chromaticity Error	34
3.5.3	Spatial Uniformity	39
3.5.4	Pixel Temporal Noise	40
3.5.5	Chromatic and Achromatic Contrast	45
3.5.6	Colour Events Classification with KNN	47
3.6	Discussion and Future Work	53
3.7	Conclusion	55
4	EVENT-BASED OBJECT DETECTOR AND CLASSIFIER FOR FRUIT DETECTION APPLICATIONS IN CLUTTERED SCENES	57
4.1	Introduction	57
4.2	Research Questions	58
4.3	Contribution	59
4.4	Methodology	59
4.4.1	Generating of Event-based Dataset	59
4.4.2	Labelling the Dataset	64
4.4.3	Measuring Evaluation Metrics	65
4.4.4	Event-based Processing and Feature Extraction	67
4.4.5	Per Pixel Spatial Downsampling	72
4.4.6	Spatial Temporal Pooling Spatial Temporal Pooling (STP) Filter	75
4.4.7	Event-based Feature Selection	76
4.4.7.1	Without Supervisory Signals	78
4.4.7.2	With Supervisory Signals	80
4.4.8	An Event-based Noise Filter	83
4.4.9	Generalised Least Square (GLS) for Highly Imbalanced Classes .	84
4.4.10	Classification Algorithms	85
4.5	Results	86
4.5.1	N-MNIST Digit Classification Results	87
4.5.2	DVS Gesture Dataset Results	93
4.5.3	Performance on Laboratory Recorded Data	99
4.5.4	Performance on Real World Scenes	106
4.5.5	Investigation of Spatial Downsampling	109
4.5.6	Investigation of Event-based Feature Selection	115
4.5.7	Comparing Performance with a Noise Filter	116
4.5.8	Classification Performance with GLS	120
4.6	Discussion and Future Work	121
4.7	Conclusion	124
5	CONCLUSION	125
5.1	Applicability of Colour Event-based Dataset	125
5.2	Viability of Event-Based Object Classification	126
5.3	Future Work	126

References	128
A Object Classification using Statistical Properties with K-Nearest Neighbors	148
B Dichromatic Sensor Assembly: Camera Synchronisation	154
C Additional Tables and Figures for the Event-based Classifier	156
D Representations of Colour Events in Various Scenes	173

List of Tables

2.1	Summary of colour event-based sensors from 2008 to 2018.	16
2.2	State-of-the-art results of deep learning in agriculture applications. . . .	20
3.1	Characterisation measures and intended use case	31
3.2	Statistical summary of the "same object different colours" scenario. Results are shown separately for the training set of 50 recordings per class and the testing set comprising of another 50 recordings.	52
3.3	Statistical summary of the "different objects same colour" scenario. Results are shown separately for the training set of 50 recordings per class and the testing set comprising another 50 recordings.	53
3.4	Statistical summary of the "different objects different classes" scenario. Results are shown separately for the training set of 50 recordings per class and the testing set comprising another 50 recordings.	53
4.1	A summary of statistical properties for all laboratory recorded data. . .	61
4.2	A summary of statistical properties for the real-world scenes.	62
4.3	Parameters used during training and inference.	88
4.4	Comparison of classifier performance on DVS gestures with different hyper-parameters using "standard weights" architecture as described in section 4.4.7. Twenty trials of each network configuration were performed using the Extreme Learning Machine (ELM) classifier consisting of 100 neurons. σ is the downsampling factor. LC is the linear classifier.	98
4.5	Comparison of classifier performance on DVS gestures with different hyper-parameters using "mixed weights" architecture as described in section 4.4.7. Twenty trials of each network configuration were performed using the ELM classifier consisting of 100 neurons. σ is the downsampling factor.	99
4.6	Summary of classification performance for all laboratory test sequences using three network architecture and two types of the classifier on the imbalanced dataset. LC refers to the linear classifier, and ELM refers to the extreme learning machine classifier.	103

4.7	Summary of classification performance for all laboratory test sequences using three network architecture and two types of the classifier on the balanced dataset. LC refers to the linear classifier, and ELM refers to the extreme learning machine classifier.	104
4.8	Summary of classification performance for all real-world test sequences using three network architecture and two types of the classifier on the imbalanced dataset. LC refers to the linear classifier, and ELM refers to the extreme learning machine classifier.	107
4.9	Summary of classification performance for all real-world test sequences using three network architecture and two types of the classifier on the balanced dataset. LC refers to the linear classifier, and ELM refers to the extreme learning machine classifier.	107
4.10	Results summary of the per-pixel downsampling method on the laboratory recorded datasets. Showing the percentage of events correctly classified as class1 (true positive) and class0 (false positive). σ is the downsampling factor.	111
4.11	Summary of per-pixel downsampling method on the real-world recorded dataset. Showing the percentage of events correctly classified as class1 (true positive) and class0 (false positive). σ is the downsampling factor.	113
A.1	Statistical summary of the 4 classes dataset. Results are shown separately for the training set of 50 recordings per class, and the testing set comprising of another 50 recordings	149
A.2	Linear classifier performance for network architecture 1	153
A.3	Linear classifier performance for network architecture 2	153

List of Figures

2.1	Human eye structure with a drawing of the retina network. Modified from Viqueira Pérez et al. [2010], Posch [2015].	7
2.2	The differences between chromatic and achromatic edges. (a) An image of a red fruit on a green background. The chromatic and achromatic images are shown with their associate edges. (b) Shows a real world scene where contours are clearly delineated in the chromatic than in the achromatic image (Source: Hansen and Gegenfurtner [2017]).	7
2.3	The progress of the state of object detection over two historical periods: Traditional detection methods and deep learning based detection methods (Source: Zou et al. [2019]).	10
2.4	Summarise the principle of operation of the DVS camera with a rotating dot stimulus (Source: Delbruck et al. [2010]).	13
2.5	Different types of colour event-based technology design. (a) Foveon or double/triple stacked diodes. (b) Bayer filter using colour filter arrays. (c) Three-chip or a combination of EBC sensors.	15
2.6	Fruits detection using colour and near infrared vision sensors. (a) and (b) show the detection of sweet pepper with the output bounding boxes. (c) and (d) show the detection of rock melons with the output bounding boxes.	19
3.1	A detailed view of the colour Dynamic and Active VIsion Sensor (cDAVIS). (a) show the actual look of the sensor and an abstract overview of the colour filtering array. (b) An example of the RGBG Bayer filter shows the relative sensitivity with the filter wavelength, where the incoming light only penetrates through the filter depending on its wavelength. (c) The resulting pixel-wise patterns when the filter is separated into four colour channels. (d) The resulting patterns are observed through the DVS and Active Pixel Sensing (APS) sensors.	26

3.2	The Colour Dynamic and Active Vision Sensor (CDAVIS) is moving in front of three circles with different colours (Red, Green and Blue) on a white background in an illuminated condition. (a) and (c) Separately showing the events triggered by each colour filter of the Bayer matrix and for each circle using a high and low contrast threshold, respectively. Because of the high threshold, the red filters are not generating any events for the red circle, while under a low threshold, the events are noisy. (b) and (d) All the events triggered by all colour filters combined. (e) An example of the log pixel illuminance response for each colour filter resulting in ON and OFF events is shown in (f). If there is a positive change in the pixel illumination, an ON event will be triggered and vice versa.	27
3.3	Appearance of primary colours through each filter in the bayer matrix. .	27
3.4	Hardware setup. It shows the mechanical assembly of the pan and tilt platform and the position of the CDAVIS in relation to the stimuli. . . .	28
3.5	Software stack. It shows a simplified flowchart of the whole process. In this case GRBL firmware is used to process G-code commands to the pan and tilt platform and sepia to process event-based data.	29
3.6	Spatio-temporal patterns for each colour filter with the signal to noise ratio. (a) shows the pixel responses using a rotating green circle on a blue background. (b) shows the pixel responses using a rotating red circle on a green background.	33
3.7	Chromaticity coordinate with the colour checkerboard. (a) The colour checkerboard from Macbeth. (b) The colour checkerboard through the APS readout of the EBC for each colour filter. (c) The International Commission on Illumination (CIE) chromaticity diagram and the colour label for each colour patch, the different colour tones are shown and parametrised with the normalised coordinates x and y	36
3.8	The results of the chromaticity error for each of the colour patches. (a) Chromaticity error for each colour patch on the macbeth colour checker board. (b) The distance between actual and measured chromaticity coordinate for each colour patch to measure the estimated colour error. . .	38
3.9	Uncertainty and non-uniformity in the event detection time [ini, 2021]. .	39
3.10	Pixel uniformity responses. (a) The uniformity of response for the ON events using all the combinations of colour transition for primary colours. (b) The uniformity of response for the OFF events using all the combinations of colour transition for primary colours.	41
3.11	Pixel temporal noise results. (a) show the total number of events across a wide range of contrast thresholds using primary colours transitions. (b) shows the total number of events across a wide range of scene illumination using primary colours transitions.	43

3.12	Temporal noise characterisation. (a) shows the event rate to measure and quantify it in terms of sensor contrast threshold (ON and OFF threshold). (b) shows the event rate to measure and quantify it in terms of scene illumination in Lux.	44
3.13	The greyscale patterns are used in the chromatic characterisation. It consists of two blocks with different greyscale values to allow the pixel to shift from one surface to another.	46
3.14	Total number of events versus contrast comprised of black and white edges with different contrast ratios. (a) show the events triggered by the DAVIS346. (b) events triggered by the CDAVIS.	48
3.15	Colours addition experiment. (a) The APS output for each pattern showing a gradual addition to blue, green and white to the red background while keeping the green circle colour the same. (b) The total number of events through each colour filter for each colour addition condition. . . .	49
3.16	Conditions selected for classification where colours and shapes are essential features to identify the differences between different classes. The edges were colourised to indicate the colour of the object. (a) Same object with different colours. (b) Different objects with the same colour. (c) Different objects with different colours.	50
3.17	Classification Results for the K-Nearest Neighbors (KNN) classifier on statistical properties for scenes where there are objects with the same shape but with different colours. Left: Accuracy on the stream of the entire event. Right: Accuracy for each colour filter.	53
3.18	Classification Results for the KNN classifier on statistical properties for scenes where there are objects with different shapes but with the same colour. Left: Accuracy on the stream of the entire event. Right: Accuracy for each colour filter.	54
3.19	Classification Results for the KNN classifier on statistical properties for "objects with different shapes but with different colours" condition. Left: Accuracy on the stream of the entire event. Right: Accuracy for each colour filter.	54
4.1	Data collection setup. (a) Real world scenes at Hawkesburry Institute of Environment (HIE). (b) Laboratory setup to control light and motion. (c) A dimetric projection of the events data from natural scenes which shows a dense spatio-temporal patterns.	60
4.2	Laboratory recorded datasets showing images from training and testing sets. (a) and (b) train and test of Shapes translation. (c) and (d) train and test of Two classes occlusion. (e) and (f) train and test of Multiple shapes. (g) and (h) train and test of Different sizes. (i) and (j) train and test of One class occlusion. (k) and (l) train and test of Complex shapes.	62

4.3	Real world datasets showing images from training and testing sets. (a) and (b) train and test of Sequence 1. (c) and (d) train and test of Sequence 2. (e) and (f) train and test of Sequence 3. (g) and (h) train and test of Sequence 4. (i) and (j) train and test of Sequence 5. (k) and (l) train and test of Sequence 6.	63
4.4	Data labelling. (a) Illustrates the method used to calculate event volumes' sensitivity and specificity around labelled data points. A volume of radius r around a line connecting the labelled points marks the boundary between true and false volumes. The event density of each sub-region designates its volume as a positive or negative volume depending on whether it is above or below the mean density of the recording as a whole (Source: Afshar et al. [2019b]). (b) Show the events data for the whole recording for both polarities. Panels (c), (d), (e) show the labelled objects for recording 1 in a dimetric projection and across the x and y -axis, respectively. Colour indicates the object ID of the same class.	64
4.5	Data labelling graphical user interface (GUI).	65
4.6	Feature Extraction using Adaptive Selection Thresholds (FEAST) Training. Top panels: Shows the adaptation of various neural signals in the network over a single independent training cycle overtime. Bottom panels: Shows the progression of learning overtime such as the change in threshold adaptation per neuron, winner count for each neuron and the missing events (i.e. events filtered out by the network).	67
4.7	The construction of a spatiotemporal time surface and features from the event-based output of an EBC. (a) EBC generate a stream of events consisting of the pixel location, direction of the change in illuminance and time. (b) An example of the event stream. (c) An example of the decaying exponentials that generate the time surface. (d) The exponential decaying time surface. (e) The intensity of the colour encodes the time elapsed since the arrival of the last event from a particular pixel (Source: Afshar et al. [2019a])	69
4.8	Diagram of the structure and function of the event-based per pixel downsampling filters. The spatial downsampling filter reduces the spatial resolution of the input data by a fixed downsampling factor, resulting in an output stream that maintains the temporal resolution of the input (Source: Cohen et al. [2018]). AER refers to Address Event Representation.	73
4.9	The exponential time surface of the feature map was generated at 28 different spatial resolutions alongside their associated feature neuron. All surfaces are all updated in an event-based fashion for each incoming event.	74

4.10	FEAST network with feature neurons. Shows the procedure from extracting features from the data to spatial pooling/flattening the features. The process starts with the input event stream followed by an 11x11 ROI associated with 16 neurons, each neuron is colour coded representing which part of the data was learned, a feature map was created for each neuron, and then a per pixel downsampling approach was applied to reduce the resolution, finally, the last layer pool the features from the feature maps using a 3x3 receptive field.	75
4.11	First network pipeline which is called "Standard architecture". The network is unsupervised from feature extraction to inference to classification.	78
4.12	Inference with imbalanced features. Shows the uses of the trained weight during inference to construct a feature map along with the feature pooling layer. The bottom panels show the number of winner neurons for each class over time.	80
4.13	Second network pipeline which is called "Mixed weights architecture". Supervisory signals were fed as input to FEAST to extract robust and discriminative features. The features were aggregated before using them during inference.	80
4.14	Third network pipeline is called "Dedicated weights architecture". In this network architecture the supervisory labels were used during training and inference.	81
4.15	Inference with balanced features. Shows the uses of the trained weight during inference to construct a feature map along with the feature pooling layer. The bottom panels show the number of winner neurons for each class.	83
4.16	Event noise processing showing the spatio-temporal events with and without applying a noise filter.	83
4.17	Heteroskedasticity. It shows the skewness of the dataset where the variance of the observations is unequal to the mean. Upon visual inspection of the residual errors, the tell-tale sign is that they will tend to fan out over time.	85
4.18	N-MNIST Dataset. (a) Showing the three saccade-inspired motions across each digit (Source: Afshar et al. [2019b]). (b) N-MNIST patterns are represented in the time dimension to static images with pixel intensity proportional to the event rate of the pixel.	87
4.19	Learned weights during FEAST training. 11x11 feature learned from the ON and OFF events of the N-MNIST dataset. Each feature represents a normalised vector reshaped to match the size of the incoming feature patches.	88
4.20	Comparison of the classification test accuracy. (a) Test accuracy with regularisation. (b) Test accuracy with different hidden layer size.	89

4.21	Confusion matrices for the 10-category classification problem. (a) and (b) Showing the results using the pooling by counting operation using the linear and ELM classifier, respectively. (c) and (d) Showing the results using the STP filter operation on the linear and ELM classifier, respectively.	90
4.22	Correctly classified events for each digit using the Pooling by Counting operation. The digits were randomly selected for visualisation purposes. The digits patterns are collapsed in the time dimension to static images with pixel intensity proportional to the spike time of the pixel using exponential time surface.	92
4.23	Correctly classified events for each digit using the STP filter operation. The digits were randomly selected for visualisation purposes. The digits patterns are collapsed in the time dimension to static images with pixel intensity proportional to the spike time of the pixel using exponential time surface.	93
4.24	DVS Gesture dataset. Three different gestures are shown where each frame represents all the events at different time interval. <i>Right Hand Clockwise</i> (top); <i>Arm Roll</i> (middle); <i>Other Gesture</i> (bottom). Pixel are colour-coded according to the pixel time.	94
4.25	Features generation at multiple network scales showing the network consistency in learning features. Panels (a) are features generated using an Region of Interest (ROI) of size 7×7 . Panels (b) are features generated using an ROI of size 11×11 . Panels (c) are features generated using an ROI of size 15×15 . Panels (d) are features generated using an ROI of size 19×19 . Panels (e) are features generated using an ROI of size 23×23 . Panels (f) are features generated using an ROI of size 31×31	95
4.26	Evolution of the FEAST neural signals and parameters during training using 20 neurons. Top panels show the network's adaptation of various neural signals, such as the change in the threshold (Left) and the neuron weight (Right) over three trials and a randomly selected test set. The bottom panels show the interneural spike rate variance (Left), which demonstrate the neuronal homeostasis showing how the network maintained a target level of spike activity and the missing spike rate over time (Right).	96

4.27	Filtering mechanism at inference. The spike events output generated by each winner neuron shows that each neuron selects events based on the learned feature. Only a subset of neurons is shown here for visualisation purposes. Panels (a) are features generated using an ROI of size 7x7. Panels (b) are features generated using an ROI of size 11x11. Panels (c) are features generated using an ROI of size 15x15. Panels (d) are features generated using an ROI of size 19x19. Panels (e) are features generated using an ROI of size 23x23. Panels (f) are features generated using an ROI of size 31x31.	97
4.28	Correctly classified events for each activity using <i>user28.natural</i> recording. The activity patterns are collapsed in the time dimension to static image representation with pixel intensity proportional to the spike time of the pixel using the time surface. (a) Arm roll. (b) Air drum. (c) Other gestures. (d) Air guitar. (e) Hand clapping. (f) Left-hand wave. (g) Right-hand wave. (h) Left-arm clockwise. (i) Right arm clockwise. (j) Left-arm counter clockwise. (k) Right arm counterclockwise.	100
4.29	Results and confusion matrices using the network with a linear classifier and an 11x11 ROI and 20 features using the "standard weights" architecture.	101
4.30	Performance of the network on the laboratory recorded dataset from the best performing architecture. Correctly classified instances as red dots, black dots are other classes. (a), (b), (c), (d), (e), (f) show the spatio-temporal patterns of the classification output for each test sequences. Original datasets are described in Section 4.4.1.	102
4.31	Performance comparison of the linear classifier for each architecture and for balanced and non-balanced classes on the laboratory recorded datasets.	105
4.32	Performance comparison of the ELM classifier for each architecture and for balanced and non-balanced classes on the laboratory recorded datasets.	105
4.33	Performance of the network on the real-world recorded dataset. Correctly classified instances as red dots, black dots are other class. (a), (b), (c), (d), (e), (f) show the spatio-temporal patterns of the classification output for each test sequences. Original datasets are described in Section 4.4.1.	106
4.34	Performance comparison of the linear classifier for each architecture and for balanced and non-balanced classes on the real world datasets.	109
4.35	Performance comparison of the ELM classifier for each architecture and for balanced and non-balanced classes on the real world datasets.	110

4.36	Results of the downsampling sweep on the shapes translation sequence. (a) The output spatio-temporal patterns were generated from the classification output at 28 different spatial resolutions. It shows that for $\sigma = 10$ the network classified events with less noise and fewer false positives. (b) The results of the evaluation matrices used 28 different downsampling factors for all test sequences.	112
4.37	Results of the downsampling sweep on the real world sequence 1. (a) The output spatio-temporal patterns generated from the classification output at 28 different spatial resolutions. (b) The results of the evaluation matrices using 28 different downsampling factors for all test sequences. .	114
4.38	Feature selection investigation on the lab recorded datasets. (a), (b), (c) and (d) show the spatio-temporal pattern from the classification output where the number of neurons for class1 is increased from 4 to 28 neurons. (e) provides a summary of the model performance for all other laboratory recorded test sequences.	116
4.39	Feature selection investigation across the laboratory datasets. (a), (b), (c) and (d) show the spatio-temporal pattern from the classification output for all laboratory recorded test sequences. (a) Two classes occlusion, (b) Multiple objects, (c) Different sizes, (d) One class occlusion, (e) Complex shapes.	117
4.40	Feature selection results on the real world datasets. (a), (b), (c) and (d) show the spatio-temporal pattern from the classification output where the number of neurons for class1 is increased from 4 to 28 neurons. (e) provides a summary of the model performance for all other real world recorded test sequences.	117
4.41	Feature selection investigation across the laboratory datasets. (a), (b), (c) and (d) show the spatio-temporal pattern from the classification output for all real world recorded test sequences.	118
4.42	The effect of using different sizes of the temporal window after the classification output removes the false positive. (a) shows the results using the laboratory recorded sequence and (b) shows the results using the real-world recorded sequence. The output events representing class1 are made less noisy by applying a small temporal window. Top panel: A dimetric projection of the classification after applying the noise filter on the output events stream with different temporal windows. Bottom panel: The Signal to Noise Ratio (SNR) results for different temporal windows starting from 1ms to 1s.	119
4.43	Classification output results after applying GLS. (a), (b), (c) and (d) show output spatio-temporal pattern of the classifier with different imbalanced bias value. (e) shows the results of the correlation coefficient with different imbalanced bias.	120

4.44	Classification output results after applying GLS on real world test sequences.	121
A.1	The classes selected for classification.	148
A.2	Classification results for the KNN classifier based on the statistical properties of the events.	151
A.3	Two types of classification architectures using FEAST algorithm. (a) A feature detector for each class followed by a classifier network. This is called "Architecture 1". (b) A feature detector for all the classes followed by a classifier network. This is called "Architecture 2".	151
A.4	A summary of the performance of the classification architecture using ELM classifier. (a) shows results on network Architecture 1 using dedicated features for each class. (b) shows results on network Architecture 2 using features from all the classes.	152
B.1	Dichromatic sensor assembly. (a) The two chips EBC is an assembly of two monochromatic Dynamic and Active Vision Sensor (DAVIS) cameras. (b) The two chips EBC with the beamsplitter to split the incoming light. (c) A 3D render of the entire setup in full assembly on the optic table. (d) Colour filters and hot mirror wavelength range and transmission percentage.	155
C.1	A green circle rotated over a red background. In this case only the rotating circles can be seen through the red filter.	156
C.2	A blue circle rotated over a red background. Due to the contrast between the red and the blue the circle can be seen through all colour filters. . .	157
C.3	A blue circle rotated over a green background. In this case the circle is visible through the red and green filter but not the blue due to the contrast difference between the blue and the green.	157
C.4	A red circle rotated over a blue background.	158
C.5	A red circle rotated over a white background.	158
C.6	A blue circle rotated over a white background.	159
C.7	A green circle rotated over a white background.	159
C.8	Lab recorded data "Shapes translation". Each plot shows the classification output for each architecture using both classification methods on the balanced and imbalanced dataset.	160
C.9	Lab recorded "Two classes occlusion". Each plot shows the classification output for each architecture using both classification methods on the balanced and imbalanced dataset.	160
C.10	Lab recorded data "Multiple objects". Each plot shows the classification output for each architecture using both classification methods on the balanced and imbalanced dataset.	161

C.11 Lab recorded data "Different sizes". Each plot shows the classification output for each architecture using both classification methods on the balanced and imbalanced dataset.	161
C.12 Lab recorded data "One class occlusion". Each plot shows the classification output for each architecture using both classification methods on the balanced and imbalanced dataset.	162
C.13 Lab recorded data "Complex shapes". Each plot shows the classification output for each architecture using both classification methods on the balanced and imbalanced dataset.	162
C.14 Real world data "test sequence 1". Each plot shows the classification output for each architecture using both classification methods on the balanced and imbalanced dataset.	163
C.15 Real world data "test sequence 2". Each plot shows the classification output for each architecture using both classification methods on the balanced and imbalanced dataset.	163
C.16 Real world data "test sequence 3". Each plot shows the classification output for each architecture using both classification methods on the balanced and imbalanced dataset.	164
C.17 Real world data "test sequence 4". Each plot shows the classification output for each architecture using both classification methods on the balanced and imbalanced dataset.	164
C.18 Real world data "test sequence 5". Each plot shows the classification output for each architecture using both classification methods on the balanced and imbalanced dataset.	165
C.19 Real world data "test sequence 6". Each plot shows the classification output for each architecture using both classification methods on the balanced and imbalanced dataset.	165
C.20 Comparison of the linear classifier results for balanced and non-balanced classes for each testing sequence using the lab recorded datasets.	166
C.21 Comparison of the ELM classifier results for balanced and non-balanced classes for each testing sequence using the lab recorded datasets.	166
C.22 Comparison of the linear classifier results for balanced and non-balanced classes for each testing sequence using the real-world datasets.	167
C.23 Comparison of the ELM classifier results for balanced and non-balanced classes for each testing sequence using the real-world datasets.	167
C.24 Space-time plot generated at 28 different spatial resolutions on the "Two object occlusion" test sequence.	168
C.25 Space-time plot generated at 28 different spatial resolutions on the "Multiple objects" test sequence.	168
C.26 Space-time plot generated at 28 different spatial resolutions on the "Different sizes" test sequence.	169

C.27	Space-time plot generated at 28 different spatial resolutions on the "One class occlusion" test sequence.	169
C.28	Space-time plot generated at 28 different spatial resolutions on the "Complex shapes" test sequence.	170
C.29	Real world data "Test sequence 2". Space-time plot generated at 28 different spatial resolutions on the lab recorded data. It shows the classification output for each downsampling value on the test set in 3D space-time.	170
C.30	Real world data "Test sequence 3". Space-time plot generated at 28 different spatial resolutions on the lab recorded data. It shows the classification output for each downsampling value on the test set in 3D space-time.	171
C.31	Real world data "Test sequence 4". Space-time plot generated at 28 different spatial resolutions on the lab recorded data. It shows the classification output for each downsampling value on the test set in 3D space-time.	171
C.32	Real world data "Test sequence 5". Space-time plot generated at 28 different spatial resolutions on the lab recorded data. It shows the classification output for each downsampling value on the test set in 3D space-time.	172
C.33	Real world data "Test sequence 6". Space-time plot generated at 28 different spatial resolutions on the lab recorded data. It shows the classification output for each downsampling value on the test set in 3D space-time.	172
D.1	Time collapsed image in the time dimension with pixel representing the colour events index. Recording shows different geometrical shapes with different colours with horizontal translation in the field of view.	173
D.2	Time collapsed image in the time dimension with pixel representing the colour events index. Recording shows 2D complex shapes of fruits and leaves with horizontal translation in the field of view.	174
D.3	Time collapsed image in the time dimension with pixel representing the colour events index. Recording shows two classes objects with different size with horizontal translation in the field of view.	175
D.4	Time collapsed image in the time dimension with pixel representing the colour events index. Recording shows two classes objects with different size with horizontal translation in the field of view.	176
D.5	Time collapsed image in the time dimension with pixel representing the colour events index. Recording shows three circles with different colour in rotation mode.	177

List of Abbreviations

AER	Address-Event Representation
AI	Artificial Intelligence
ANN	Artificial Neural Network
APS	Active Pixel Sensing
ATIS	Asynchronous Time-Based Sensor
BSI	Back Side Illumination
CIE	International Commission on Illumination
CNN	Convolutional Neural Network
CMOS	Complementary Metal Oxide Semiconductor
DVS	Dynamic Vision Sensor
DL	Deep Learning
CDAVIS	Colour Dynamic and Active Vision Sensor
DAVIS	Dynamic and Active Vision Sensor
DPM	Deformable Part-based Model
EBC	Event-Based Camera
ELM	Extreme Learning Machine
FEAST	Feature Extraction using Adaptive Selection Thresholds
FSI	Front Side Illumination
HIE	Hawkesbury Institute of Environment
HDR	High Dynamic Range
HOG	Histogram of Oriented Gradients
GLS	Generalised Least Square
GPU	Graphical Processing Unit
OPIUM	Online Pseudo-inverse Update Method
KNN	K-Nearest Neighbors
ROI	Region of Interest
SNR	Signal to Noise Ratio

SNN	Spiking Neural Network
STP	Spatial Temporal Pooling
YOLO	You Look Only Once

Chapter 1

INTRODUCTION

1.1 Motivation

Conventional video cameras represent the world as a sequence of static images captured in rapid succession. By repeating this process, the information from the visual scene becomes sampled at a fixed rate (i.e. discrete point in time). While this is the most common way of capturing visual information, this method suffers from several drawbacks: (i) The dynamic of the scenes depends on the sampling rate. Static scenes are sampled at the same rate as high-speed scenes leading to redundancy or information loss between the frames. (ii) Motion-blur occurs during rapid motion when the shutter is open. (iii) Uniform exposure across the pixels limits the intrascene dynamic range. Despite the advances in computational power, pixel resolution, and frame rates, even the state-of-the-art computer vision methods fall far short of biological vision systems' robustness, reliability, and energy consumption.

Event-based cameras (EBC) attempt to replicate some of the benefits of biological retinas and provide a vastly different paradigm in which to sense and process the visual world. They capture visual information continuously through time, with each pixel storing a reference level of brightness and continuously comparing it with the current level and generating data only when a certain amount of change is detected. The final output of the camera is an asynchronous stream of events triggered by per-pixel changes in brightness.

While event-based vision is an emerging technology in the era of mature frame-based camera hardware and software, there is no agreement on the best methods to process events, notably because they primarily depend on the application. Different trade-offs are involved for each method, such as latency vs power consumption and accuracy or sensitivity vs bandwidth and processing capacity, which should be processed and adjusted according to the algorithm and platform capacity.

Two categories of event-based methods can be distinguished: (i) methods designed for static cameras with movable objects and (ii) methods designed for moving cameras. EBC sends only information when a change in contrast between edges is observed. When the camera is static, the events are mainly triggered by moving objects. For a moving camera, the events are triggered by the whole scene, including the foreground

and background. Static camera setups allow for robust event-based algorithms due to the sparsity of the scene achieving minimum latency. The latter is more challenging because an algorithm is not only waiting for a change in contrast in the scene but also requires an understanding of the dynamic of the entire scene, which includes real and noise events. The mode of operation of our setup comprised on an EBC moving freely in front of static scene with dense and cluttered background. This study raises some fundamental questions which drive the algorithmic design, such as:

- What to detect?
- How it should be detected?
- What kind and how many distortions can be handled?

A challenging research area involves developing methods to optimise the sensor and algorithm parameters for optimal performance and preserve temporal resolution throughout the computation and processing pipeline.

Another challenge is to develop bio-inspired systems that are natively event-based end-to-end from perception to control and actuation. Thus, event cameras pose the challenge of rethinking perception, control, and actuation. As a result, there is a need to develop new algorithms and paradigms in which to handle and process the event-based data in complex and cluttered scenes without the need to convert the events into frames.

The majority of event cameras work to date use grayscale (i.e. monochromatic) [Lichtsteiner et al., 2006, Posch et al., 2011] and only a minority of researchers have explored colour event cameras, primarily due to the challenges of building such sensors. Colour processing in the human visual system is not fully understood yet, and colour computer vision tasks are handled by ad-hoc and computationally expensive algorithms. A challenging research area in colour events is replicating some of the benefits of biological retinas in colour recognition and processing. It is an inherently challenging problem because the methods commonly applied to conventional imaging are not directly applicable to the colour event sensor. That is because EBCs transmit brightness changes per pixel and per colour filter asynchronously rather than measuring absolute brightness at a constant rate. For that reason, some fundamental questions arise, such as:

- How should we process colour spikes?
- Are colour spikes important for visual recognition?
- Do we need more complex algorithms that work on simple grey levels data or a simple algorithm for complex colour datasets?

This thesis addresses and explores these questions, provide insights and answers into detecting objects in cluttered scenes, and investigates the role and importance of colours in event-based vision.

1.2 Aims

This work explores event-driven algorithms that can be applied in agricultural applications such as fruits detection and classification. The aim is to investigate methods to extract robust features to perform detection and classification in dense and cluttered scenes using only events spikes.

This work presents insights into neuromorphic colour imagers through rigorous characterisation and identifying key parameters that can provide more relevant information for features. The findings of this thesis will provide a better understanding of the pixel response and the internal properties of the Bayer filter purely from the DVS output.

1.3 Contribution of this Work

The main contributions of this thesis are the development of end-to-end event-based object detection and classification architecture that comprises an unsupervised feature extractor with a single feed-forward (i.e. without back-propagation) classification algorithms. In addition, this work aims to characterise the novel data from an event-based colour sensor to demonstrate its key operations in colour transition and classification using the events statistical properties.

1.4 Structure of this Thesis

Each section in this thesis introduces the research gap with a list of research questions and contains a detailed discussion of the specific contribution made. This thesis is organised as follows:

- Chapter 2 presents a detailed literature review about event-based devices used in this research, such as the monochromatic and colour sensor. It also covers feature detection algorithms in conventional computer vision and the event-based domain relevant to this work. The literature also includes recent advances in agricultural applications, which is the main application of this thesis.
- Chapter 3 provides an in-depth study about the CDAVIS sensor by characterising the response of the colour events purely from the DVS output and showcases the spatio-temporal output patterns generated by each colour filter.
- Chapter 4 introduces three different variants of event-based object detection and classification architectures that take into account supervisory signals in the event-based visual data across different types of datasets such as laboratory recorded data and recording from real-world scenes. Multiple algorithm configurations were tested with a range of back-end classifiers, with the performances being analysed at each processing stage. The performance of the network was evaluated on the N-MNIST and DVS Gesture dataset.

- Chapter 5 provides the conclusions to work performed in this thesis and includes a discussion of potential future work.

Appendix A contains a detailed method for classifying colour against non-colour event-based data using the statistical properties of the events. Appendix B presents the initial dichromatic colour event prototype comprised of two event-based sensor chips with two colour filters and using a dichroic beam-splitter to ensure that the full spatial resolution of the sensors is preserved. Appendix C presents additional figures and tables included as supplementary materials related to chapter 4. Appendix D provides more examples of how the spatio-temporal pattern of colour events look in various conditions.

Chapter 2

LITERATURE REVIEW

2.1 Colour Sensing and Perception

The retina is part of the eye that detects colour, and the visual cortex is the part of the brain that processes the information it receives from the retina. The information is interpreted from available light and used to build and construct a representation of the visual world. This process begins when light passes through the transparent element of the eye and hits the retina. When viewing a scene, the human visual system extracts information about the light wavelength, which is why we see in colour. This process results in the perception in which colour is one of the elements.

Throughout evolution, the benefit of colour for our ancestors was clear: seeing in colour makes it easier to find food, such as the colour of the fruit against leaves and the ability to detect camouflaged animals. Considering the sheer amount of information that surrounds us and how much of it is based on colour, it is not surprising that most visual information in our world is colour-coded, such as traffic light, advertising, graphic design, and the digital world.

Conventional digital processing systems have primarily focused on intensity grayscale images, and colour was just considered as a dimensional extension of intensity dimension. That is, colour images were treated just as three grey value images, not taking into consideration the multidimensional nature of human colour perception or colour sensory system in general [Trémeau et al., 2008].

Colour is an internal sensation produced by the visual system from light wavelengths emitted by external objects. The relationship between chromatic stimuli and perception is complicated and not fully understood.

Historically, human colour perception is described by two major theories: the trichromatic colour theory and the colour opponent-process theory. The Young-Helmholtz trichromatic theory, which was introduced in 1801 [Lee, 2008] gave a complete description of colour perception and suggested that the retina contained three types of nerve fibres (receptors), which are stimulated by a lesser or greater wavelength that corresponds to red, green, and violet colours. This finding led to the hypothesis that normal colour vision is based on the activity of three types of receptors, each with different peak sensitivity.

A few years later, James C. Maxwell [noa, 1865] confirmed Young and Helmholtz's theory and demonstrated that any colour in the spectrum could be matched with the monochromatic primary colours. However, they incorrectly assumed that colour perception resulted from a linear combination of cell activities, and the colour appearance was not studied yet. Hering's opponent colours theory introduced in 1892 [Lee, 2008] was the first to bring the issue of colour appearance to prominence, and he proposed the colour-opponency mechanism-based of four colours: yellow, red, blue, and green arranged in mutually excluding pairs, and two achromatic colours: black and white. The theory explains colour vision phenomena that result from how photoreceptors are interconnected neurally. However, Jameson and D'Andrade [1997] have proved that colour perception is not due to opponent processes and that the colour perception emerges further up in the visual pathway. More recently, [Land, 1959] proposed the retinex theory (Land's Retinex), which refers to the contraction of the Retina and cortex. The theory indicates that colour perception with colour constancy involves all levels of visual processing from the Retina to the visual cortex contradicting the previous theories that assume colour perception happens at the photoreceptor level.

As illustrated in Figure 2.1, the human eye is made up of three distinct layers of tissue: Sclerotic coat, Choroid coat and Retina. The sclerotic coat is the outer layer; it contributes to the image-forming process by refracting light entering the eye. The choroid coat is the intermediate layer, this layer is pigmented with melanin that reduces reflection of stray light in the eye, and it also forms the iris, which adjusts the size of the pupil to regulate the amount of light admitted into the eye. The nervous system controls the pupil, in dim light, the pupil opens wider letting more light into the eye, and in bright light, the pupil closes [Zaidi et al., 2007]. The contraction of the pupil is thought to be more of a primary defence mechanism to protect the Retina against sudden changes in light. The inner layer of the eye is the Retina. Anatomically, the Retina contains three types of photoreceptor cells: cones, rods and horizontal, bipolar, amacrine and ganglion cells. A photoreceptor is capable of performing phototransduction, by which the emitted photons are converted into an electrical signal which is then transmitted to other neurons. Cones and rods are responsible for high precision spatio-temporal light sensing. In terms of light, rods can function in low light, whereas cones need much higher light levels. Rods are responsible for night vision, and cones are used in daytime vision seeing colour and visual acuity. Cone cells exist in three types: S, M and L [noa, 1990]. Given an electromagnetic spectrum, the photoreceptor excitation can be derived from the sensitivity curves. Photoreceptors can also change their sensitivity curve based on the amount of light received which is a slow adaption process and can take several minutes.

The human visual system can adapt to colours, where colours remain unchanged regardless of the changes in illuminance [Viqueira Pérez et al., 2010]. It is known as chromatic adaption and colour constancy which is defined by the ability to deduct light spectrum to preserve the chromatic appearance of a particular object within the visual field. Colour adaptation is an internal process, and it happens after the photoreceptor

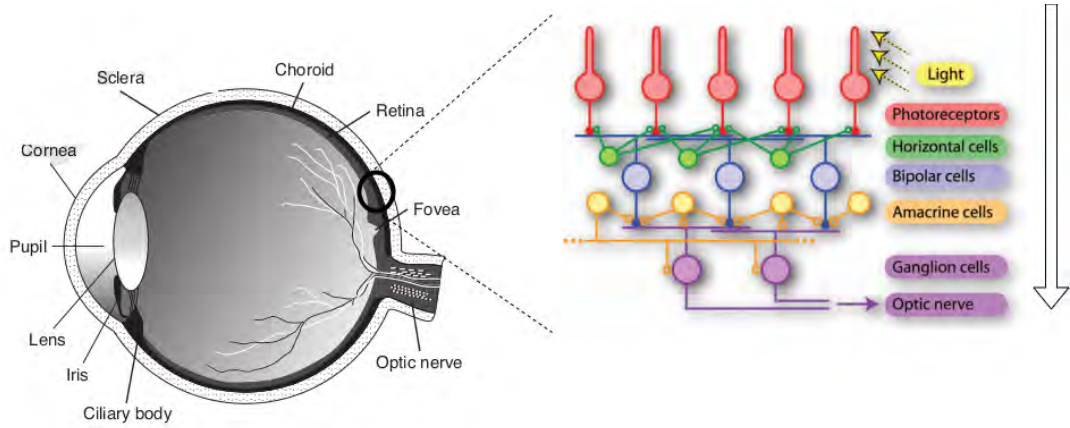


Figure 2.1: Human eye structure with a drawing of the retina network. Modified from Viqueira Pérez et al. [2010], Posch [2015].

receives the chromatic stimulus from the outside world. Colour constancy is an extreme case of chromatic adaptation that associates a colour to an object regardless of the light in which the object is seen. For instance, object colour might look similar during day and night, even with fluorescent lighting.

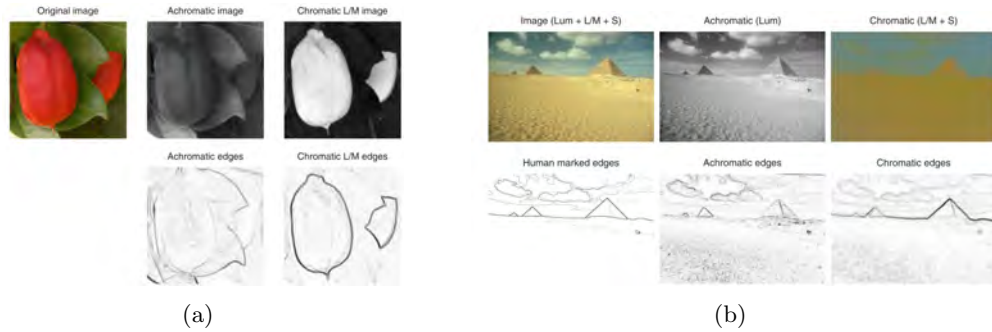


Figure 2.2: The differences between chromatic and achromatic edges. (a) An image of a red fruit on a green background. The chromatic and achromatic images are shown with their associate edges. (b) Shows a real world scene where contours are clearly delineated in the chromatic than in the achromatic image (Source: Hansen and Gegenfurtner [2017]).

Recent work in Hansen and Gegenfurtner [2017] investigated the benefits of chromatic edge contrast on object-contour perception. This study shows that chromatic information is essential for representing object contour and detecting objects quickly and easily. Furthermore, chromatic edges (e.g. red-green) do not result from shadow but indicate a change in surface reflectance, which may signal an object contour. As shown in Figure 2.2, in natural scenes, most of the edges combine luminance and colour, which are only represented chromatically. However, in the achromatic image, the edges of the fruits are hardly detectable because the luminance of the fruits and the background is almost the same. This is particularly important for the DVS because if the edge contrasts are delineated, events will be triggered, indicating the presence of an object. The study also suggests that strong chromatic edges likely signal an object

boundary, while strong achromatic edges can also result from shadow.

2.2 Frame-based Computer Vision

The ability to recognise patterns and objects are an ordinary human skill. The same recognition problem presents an unprecedented challenge for machine vision systems. To allow the machine to recognise objects, feature extraction techniques have become an apparent need in many processes which have much to do with computer vision, object detection and localisation.

In the following sections, we provide a comprehensive overview of feature extraction methods in Section 2.2.1, followed by a summarisation of recent methods in object detection and their contribution to artificial intelligence in Section 2.2.2.

2.2.1 Feature Extraction

Extracting relevant features from the environment presents a challenge due to the high information content of the visual input. Computers currently face challenges in processing all the information they receive from the world in real-time due to the computational power needed. To compensate, computer vision systems often select only the relevant part of the scene because locating and retrieving a particular feature in more complex environments require an attentive system that allows an algorithm to isolate their target within the environment.

In recent times, the volume of available computer vision data has grown tremendously, including the number of classes and the amount of raw information each instance contains. Given the excessive amounts of raw information, the task of feature extraction is as critical as ever for the successful application of machine learning. Such applications for feature extraction include natural language processing, image recognition, text categorisation, audio analysis, bioinformatics. Many efforts were spent on building and maintaining complex feature extraction pipelines, which has driven the research in both industrial and academic fields and has made it into a broad and diverse topic [Sculley et al., 2014]. The primary goal of feature extraction is to extract salience features that are understandable to a learning algorithm from input data while removing noise and unwanted events [Guyon et al., 2006]. The input information is then transformed into a feature space that can be used as input for a learning algorithm. Without informative features, it is impossible to generalise a trained network, but if relevant information is retrieved and extracted, then a simple method can lead to superior results [Yang and Pedersen, 1997].

A common approach for feature extraction is feature selection or variable subset selection which deals with selecting a subset from a large input feature set [Blum and Langley, 1997]. Feature selection is implemented on various methods such as variable ranking [Rakotomamonjy, 2003], feature subset selection [Narendra and Fukunaga, 1977] and penalised least squares [Fan and Li, 2001]. Another general approach is feature re-weighting, which aims at finding the best weight for each feature [Wettschereck

et al., 1997].

Another approach is feature normalisation which involves feature centring, rescaling to target range or scaling to unit balls [Aksoy and Haralick, 2001]. Feature construction is a different approach of feature extraction that involves creating new features from input data such as bag-of-words or n-gram based vector [Liu and Motoda, 1998], and clustering, in which case the constructed features are cluster centres instead of original variables. Feature embeddings involve a map from the input data to a manifold to preserve some statistical measure on the data, such as variance and reconstruction error. An example of feature embedding is dimensionality reduction which involves projecting features to lower-dimensional subspace [van der Maaten et al., 2009]. It aims at learning a map from input data to a linear dimensional geometric structure. Principal Component Analysis [Pearson, 1901], Random Projection [Hegde et al., 2008], Linear Discriminant Analysis [Fisher, 1938] are also examples of dimensionality reduction. Extensive studies have been devoted to nonlinear dimensionality reduction, which aims at learning a map from input data onto some nonlinear low-dimensional geometric structure or manifold [Lee and Verleysen, 2007].

The majority of these feature extractions method relies mostly on a distance metric on the input space. Distance metric learning is a crucial area of feature extraction. Some distance metric algorithms are local LDA [Fan and Li, 2001], relevance component analysis [Bar-Hillel et al., 2003], large margin nearest neighbor [Weinberger and Saul, 2009], Bayesian active distance metric learning [Yang et al., 2012].

The breakthrough in Artificial Neural Network (ANN) and Deep Learning (DL) had a significant impact on feature extraction because the learning mechanism is composed of multiple nonlinear transformations of the input features. This mechanism of extracting features through hierarchical layers is called representation learning. The aim of representation learning leveraging ANN is an automated way for feature extraction without the need for human-engineered handcrafted features.

2.2.2 Object Detection

Object detection is a crucial computer vision task that detects instances of visual objects of a particular class. Object detection aims to build computational models and techniques that provide information for any computer vision application about the object type and where it is located within the visual field. In the last two decades, it has become widely accepted that the progress of object detection has generally gone through two significant historical periods Zou et al. [2019] which was highly influenced by AlexNet architecture and considered a breakthrough in the field of object detection as illustrated in Figure 2.3. Most of the early attempts to develop object detection algorithms were based on handcrafted features due to the lack of sufficient datasets at that time and fewer computational resources. That is because they showed superior performance, and they were easier to train with a much smaller sample [Lin et al., 2020, Sejnowski, 2020]. Consequently, researchers worked on designing sophisticated feature detection and developed multiple ways to increase the speed of computing resources.

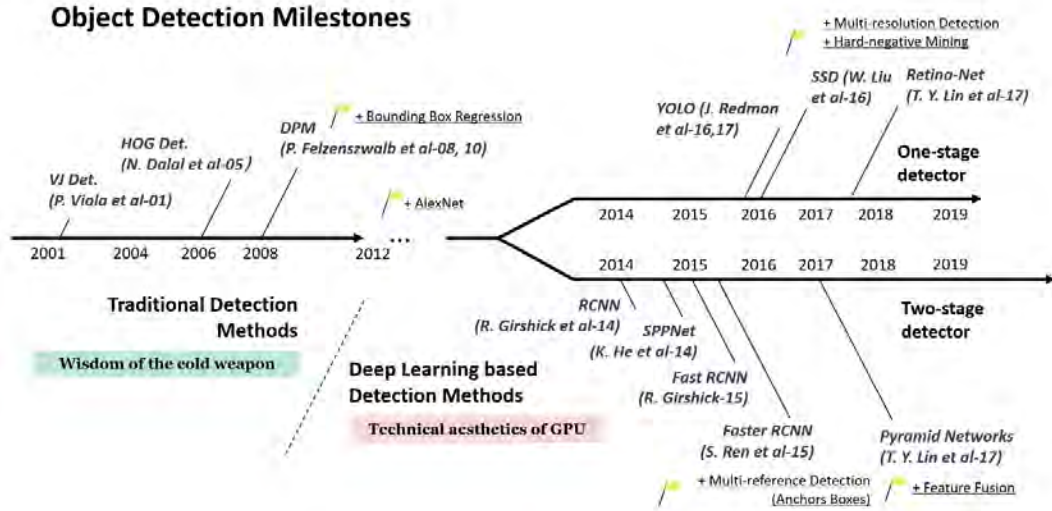


Figure 2.3: The progress of the state of object detection over two historical periods: Traditional detection methods and deep learning based detection methods (Source: Zou et al. [2019]).

Notable work on developing a robust and reliable object detector was proposed by [Viola and Jones, 2001, 2004] (Viola-Jones Detectors), which achieved real-time detection of human faces for the first time and faster than any other algorithms at the time under comparable detection accuracy. The Viola-Jones detector involves sliding windows to go through all possible locations and scales in an image to see if any window contains a human face. Dalal and Triggs [2005] proposed Histogram of Oriented Gradients (HOG) feature descriptor algorithm. HOG was developed to detect a variety of object classes of different sizes to balance the feature invariance (i.e. translation, scale, illumination, etc.) and has proved to be effective and efficient across a wide variety of applications. Felzenszwalb et al. [2008] extended on HOG and proposed Deformable Part-based Model (DPM) algorithm. DPM follows the detection philosophy of "divide and conquer", where the training is considered as the learning of a proper way to decompose an object, and the inference is considered as an ensemble of detections on different object parts. Although many of current object detection algorithms surpassed DPM in terms of performance and speed, many of them are still deeply influenced by its valuable insights.

On the one hand, a significant increase and availability of high-performance computing power such as Graphical Processing Unit (GPU) led to the fast evolution of object detection techniques. Also, the resurgence of Convolutional Neural Network (CNN) [Lecun et al., 1998] improved the performance of detection algorithms by automatically learning features as opposed to using handcrafted methods [Nanni et al., 2017]. Thus, handcrafting features has become unnecessary for most applications, as CNN learns what features to extract via backpropagation.

In recent times, the rapid development of DL has brought new types of object detection algorithms, which have led to remarkable breakthroughs in terms of performance, computational efficiency, and speed. Thenceforth, object detection started to evolve at

an unprecedented speed. Deep learning-based object detection can be divided into two methods: one-stage detection, which frames the detection in one step, and two-stage detection, which frames it as a coarse-to-fine process.

One Stage object Detection: You Look Only Once (YOLO) was proposed by [Redmon et al., 2016] as a one-stage detector. It is a real-time object detection algorithm that can predict up to 100 bounding boxes per image by straight extracting features from input images to predict class probabilities. However, a major disadvantage of YOLO is that it fails to detect accurate object localisation. YOLOv2 [Redmon and Farhadi, 2017] is an extension of YOLO and is an improvement over the previous design in terms of speed and precision. Such improvements involved are listed below:

- Batch normalisation [Ioffe and Szegedy, 2015] which outputs activations with the same distribution ahead of each convolutional layer.
- A higher resolution classifier includes a fine-tuned network to adjust to higher resolution frames.
- Convolutional with *Anchor Boxes* which predict class and objectness for every anchor box by removing fully connected layers.
- Predicting the size and aspect ratio of anchor boxes by using K-means clustering on the training set bounding boxes to get good priors automatically.
- Fine-Grained Features concatenates the higher resolution features with the low-resolution features by stacking adjacent features into different channels.
- Multi-Scale Training where the network chooses different image dimensions after several iterations.

YOLOv3 [Redmon and Farhadi, 2018] adapts to more complex datasets with many overlapping labels, and it uses three different feature map scales for bounding box prediction using a robust feature extraction pipeline (DarkNet-53). Single Shot Detector (SSD) [Liu et al., 2016] introduces multi-reference and multi-resolution detection techniques and significantly improve the detection accuracy specifically for small objects as well as precision. Deconvolutional Single Shot Detector (DSSD) [Fu et al., 2017] increases the resolution of feature maps strengthen features. Every deconvolution layer predicts a variety of objects of different sizes. RetinaNet [Lin et al., 2018] is a one-stage detector that introduces focal loss function by reshaping the standard cross-entropy loss so that detector puts more focus on complex and misclassified examples during training. The network has achieved higher accuracy and maintained a higher detection speed with the focal loss.

Two Stage object Detection: RCNN [Girshick et al., 2014] is a two-stage region-based CNN detector. It generates region proposals on the feature map by selective search and extracts a fixed-length feature vector from each region proposal. A linear support vector machine (SVM) then applies to classifying objects in one image to predict the bounding box location. R-CNN suffers from slow detection speed due to the

redundant feature computations on many overlapped proposals. Fast R-CNN [Girshick, 2015] comprised of extracting features from the input image then passes ROI pooling layer to fixed-sized features which will be fed to the classifier to predict the bounding box. The features are extracted once and are then sent to CNN for classification, unlike, R-CNN which performs a forward pass for each region proposal without sharing computation. Faster-RCNN [Ren et al., 2016] replaced the proposal ROI from Fast-RCNN by the novel Region Proposal Network (RPN) as a fully connected network to efficiently predict region proposals with a wide range of scales and aspect ratios. The RPN speed up the region proposal mechanisms because it shares the full convolutional feature maps with the detection network. Experiments with Faster-RCNN greatly improved precision and detection efficiency.

The development of new object detection networks was bolstered by the availability of open-source datasets and benchmarks. A number of well-known datasets and benchmarks have been released such as PASCAL VOC Challenges which includes VOC2007 [Everingham et al., 2015b] and VOC2012 [Everingham et al., 2015a], ImageNet challenges [Russakovsky et al., 2015], MS-COCO Detection Challenge [Lin et al., 2015]. The uses of challenging datasets as a benchmark are significant in advancing the current state of DL because they can draw a standard comparison between different algorithms and set goals for new solutions. These benchmarks became the standard in evaluating every model and were adopted to measure the performance of algorithms with the corresponding dataset. However, these benchmarks do not apply to neuromorphic algorithms based on the standard frame-based dataset.

With the continuous development of object detection, there is a need for more accurate and precise real-time systems to achieve high accuracy and efficient detection. New directions need to focus on extracting rich features, exploiting good representations, improving processing speed, training from scratch, anchor-free methods, solving sophisticated scene issues (small objects, occluded objects), increasing localisation accuracy and enhancing classification confidence [Jiao et al., 2019].

2.3 Event-based Computer Vision

Event-based sensors pose a paradigm shift in the way visual information is acquired. Their novelty has been exploited in applications where high speed, low data rate are crucial elements, such as in mobile robotics, augmented and virtual/augmented reality and video game applications, to name a few. However, because they work fundamentally different from standard vision sensors, novel methods are required to process their output and unlock their potential.

In the following sections, we describe the event-based vision sensor and its mode of operations 2.3.1 for both monochromatic and colour vision. We provide an overview of recent methods for event-based feature extractions 2.3.2 following by current implementations to perform event-based object detection and classification in Section 2.3.3.

2.3.1 Event-Based Sensing

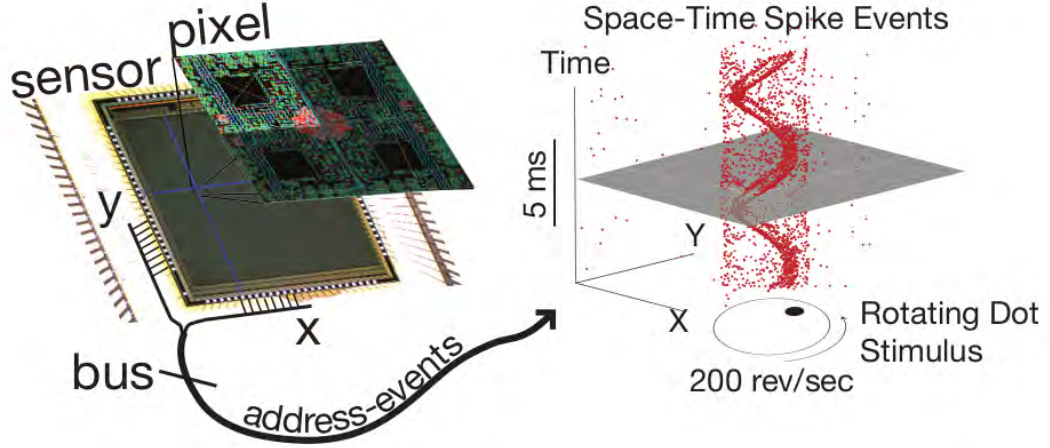


Figure 2.4: Summarise the principle of operation of the DVS camera with a rotating dot stimulus (Source: Delbruck et al. [2010]).

The rise of computer vision as a field has been deeply inspired by Hubel and Wiesel [1959] work which expanded our understanding of the biological visual system by introducing several new analyses and approaches for neuronal processing in the cortex. They demonstrated that visual processing always starts with simple structures, such as oriented edges. Inspired by this experiment, Fukushima et al. [1983] developed the first-ever neural network capable of recognising patterns using a sliding window moved across images. A few years later, Lecun et al. [1998] developed a unique learning algorithm called back-propagation and was applied to Fukushima’s CNN architecture, which became the essential ingredients of current image recognition models. The field then took advantage of the rapid growth in the amount of annotated data and the significant improvements in the power of optimised co-processors, such as GPU. As a result, computational models could far exceed the performance of previous forms of artificial intelligence in standard machine learning tasks. The CNN became the de-facto method to solve complex image-driven pattern recognition tasks. It created a range of new avenues for computer vision research offering a better understanding of its mechanism in ANN architectures. However, at the implementation level, only marginal similarities can be recognised between brain-like computing and analogue neural network, as used in Artificial Intelligence (AI) applications. This is likely because conventional datasets are mainly captured using a frame-based sensor, which is fundamentally different from the human retina mode of operation. Frame-based sensors capture still and static images taken at a constant rate resulting in a snapshot of the whole scene and containing redundant data, making it a disadvantage in applications where low latency, low power, and high dynamic range are critical for decision making.

This idea of creating computational models that mimic the biological systems has stimulated numerous studies to understand the relationship between the human sensory and central nervous systems and apply computational neuroscience knowledge to

construct intelligent machines. An example of this is the development of the first silicon retina event cameras, which was made by Fukushima et al. [1970] and the work of Mahowald [1992], which presented the first viable biologically-inspired device. Over the next few decades, the neuromorphic community has developed a series of different forms of silicone retina, including asynchronous spatial and temporal contrast sensors by Boahen’s group Zaghoul and Boahen [2004a,b], temporal intensity sensors by Mallik et al. [2005], temporal difference sensors by Kramer [2002], Lichtsteiner et al. [2006], and spatial contrast sensors by Barbaro et al., Lenero-Bardallo et al. [2009], Posch et al. [2011].

Silicone retina or EBC differs from conventional frame cameras by outputting a stream of asynchronous events that encode the time, location, and polarity of brightness changes. EBCs offer several vital properties such as the high temporal resolution on the order of μs , a high dynamic range greater than 120 dB, low power consumption, and high pixel bandwidth on the order of kHz. For an EBC, each pixel is triggered independently and asynchronously in response to the logarithmic luminance change when it exceeds a fixed threshold. A frame-based camera acquires full images at a constant rate regardless of whether this information has changed since the last frame. Collecting and processing this additional data wastes resources and increases channel bandwidth and memory requirements and high transmission power dissipation. The events bundle spatial and temporal information and boolean polarity encoding, indicating that there is a significant change corresponding to an increase or decrease in brightness. Events are transferred using Address-Event Representation (AER) communication protocol through a USB bus. Figure 2.4 illustrates an example of a high-speed stimulus that generates a sparse and asynchronous digital stream of address-event, which rapidly signifies changes in the scene reflectance.

Bio-inspired vision sensors, such as the EBC, have shown their potential to provide advantages in a range of applications such as object tracking Delbruck and Lang [2013], Glover and Bartolozzi [2016], surveillance and monitoring Litzenberger et al. [2006a], object recognition Lee et al. [2014], Amir et al. [2017], Litzenberger et al. [2006a], depth estimation Rogister et al. [2012a,b], 3D scanning Matsuda et al. [2015], optical flow estimation Zhu et al. [2018a], Matsuda et al. [2015], image reconstruction with high dynamic range Rebecq et al. [2019], Kim et al. [2014b], Simultaneous Localisation and Mapping (SLAM) Kim et al. [2016], Rebecq et al. [2017], Vidal et al. [2018], image deblurring Pan et al. [2018], and space situational awareness Cohen et al. [2019], Chin et al. [2019].

Several approaches to colour sensing have been considered for neuromorphic sensors starting in 2007. Such approaches were (i) Foveon method (i.e. stacked photodiodes) consists of vertically stacking photodiodes with different colour filters resulting in high spatial resolution for each colour, (ii) Bayer filters method (i.e. Colour filter array) consists of placing a discrete array of colour filters over pixels which implies reducing the sensor resolution at least three times and (iii) three chips method which consists of combining three-camera chips and using an array of beam splitting mirrors and

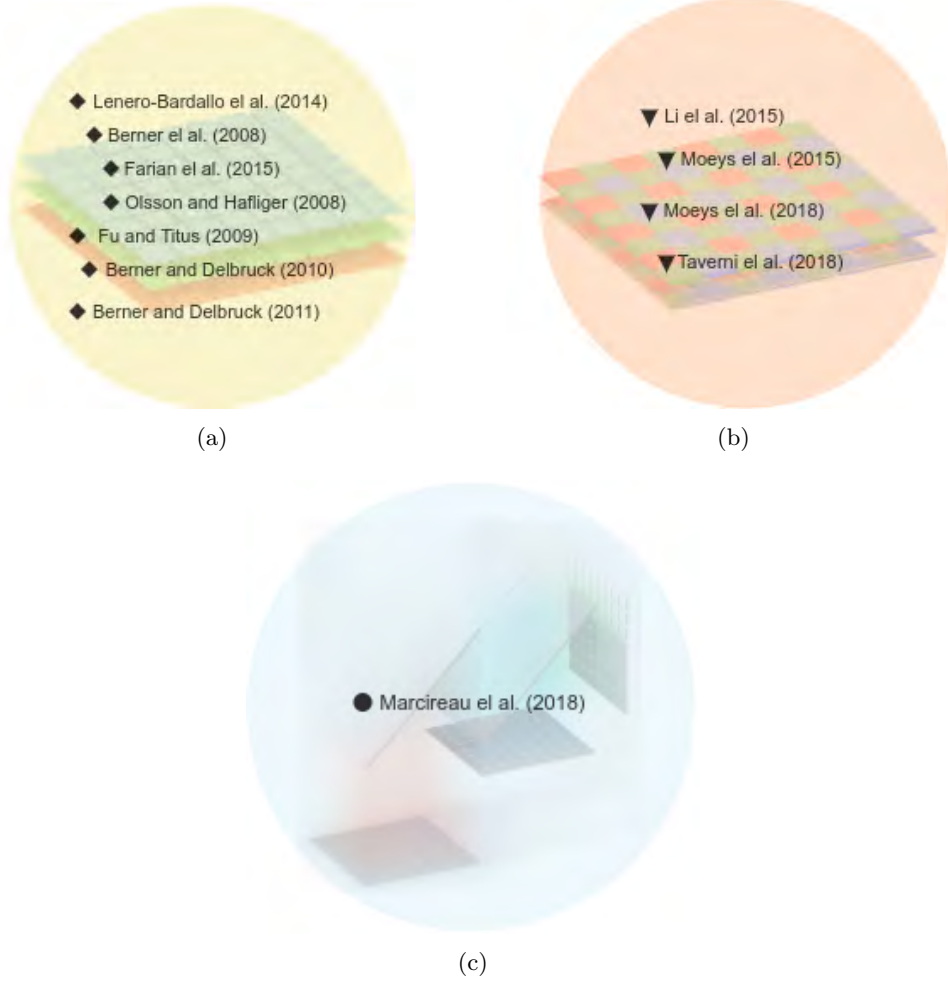


Figure 2.5: Different types of colour event-based technology design. (a) Foveon or double/triple stacked diodes. (b) Bayer filter using colour filter arrays. (c) Three-chip or a combination of EBC sensors.

colour filters to channel light into three separate wavelengths (see Figure 2.5 and Table 2.1 for a summary). One of the earliest attempts to fabricate a colour sensitive silicon retina was made by Berner et al. [2008] using the Foveon method by stacked two-diode structure to measure relative long and short-wavelength spectral content. Shimonomura [2011] proposed a three silicon retinas design and demonstrated colour constancy, Marcireau et al. [2018] built a three-chip colour sensor using Asynchronous Time-Based Sensor (ATIS) sensors and demonstrated a low-power colour signature and colour tracking algorithm. Fu and Titus [2009] proposed a neuromorphic chip which performs colour disambiguation. Berner and Delbruck [2011] proposed a colour vision DVS or CDAVIS using a single buried double junction (BDJ) photodiode, which detects both brightness changes, as would a DVS pixel, and wavelength changes but does not provide the absolute brightness. The design was limited by large pixels and poor colour separation. Li et al. [2015] combined the design of DAVIS with a Bayer filter RGB sensor, which results in an RGBW sensor that can read out events and frames and in which the white channel used is the DVS pixel photodiodes. Lenero-Bardallo

et al. [2014] and Farian et al. [2015] proposed a tricolour silicon retina design using stacked photodiodes offering information on different colour spectra from the same 2D location in the focal plane obtaining full-resolution colour information. The latest EBC design was the colourDAVIS346, which was proposed by Taverni et al. [2018], which takes advantage of the advancement of the back-side illumination technology and Bayer filtering with improved sensitivity and had a superior quantum efficiency and fill factor. Scheerlinck et al. [2019] have published a colour-event based dataset produced by both the CDAVIS346 sensor and a colour event simulator.

Table 2.1: Summary of colour event-based sensors from 2008 to 2018.

Sensor Design	Resolution	Colour Technology	CMOS Technology	Functionality	Characterisation	Dataset	Availability
Berner et al. [2008]	4x5	BDJ	1.5 μm CMOS	Asynchronous time-to-first-spike	✓	✗	✗
Olsson and Hafiger [2008]	N/A	BDJ	0.35 μm CMOS	Linear transformation of intensity	✓	✗	✗
Fu and Titus [2009]	N/A	BDJ	1.5 μm CMOS	Color change-intensity	✓	✗	✗
Berner and Delbruck [2010]	N/A	BDJ	0.5 μm CMOS	Color change-intensity	✓	✗	✗
Berner and Delbruck [2011]	N/A	BDJ	180 μm CMOS	Colour/log intensity change	✓	✗	✗
Lenero-Bardallo et al. [2014]	22x22	BTJ	90 nm CMOS	Pulse frequency modulation	✓	✗	✗
Farian et al. [2015]	16x16	BTJ	90 nm CMOS	Color temporal contrast detection	✓	✗	✗
Li et al. [2015]	QVGA	RGBW CFA	0.18 μm CMOS	Log-intensity change	✗	✗	✗
Moeyss et al. [2017]	192px	RGBW CFA	180 nm CMOS	Log-intensity change	✓	✗	✗
Moeyss et al. [2018]	192px	RGBW CFA	180 nm CMOS	Log-intensity change	✓	✗	✗
Taverni et al. [2018]	346x260	RGBG CFA	180 nm CMOS	Log-intensity change	✗	✓ Scheerlinck et al. [2019]	✓
Marcireau et al. [2018]	304x240	Three-chip	180 μm CMOS	Change detection and PWM	✓	✗	✗

2.3.2 Event-based Feature Extraction

Due to the many years of research in understanding and improving frames data, current frame-based feature extraction approaches have produced numerous sophisticated and robust algorithms. However, event-based feature extraction poses a different and more challenging task because the nature of processing the events data is not standardised. It differs based on the camera mode of operation, application and system trade-offs, which heavily influence the dataset’s quality. Since the EBC output is neither a single aggregated frame nor a video consisting of frame sequences, the majority of existing feature detectors are not applicable for event-based data or require an additional layer of processing to convert the events to frames prior to feature extraction. While methods to convert events to frames provide a clear representation of events in an image form, it does not take into account the behaviour of every individual pixel, which is considered one of the critical properties of the event-based sensor.

Event-based processing systems commonly consist of transforming incoming events into alternative representations to facilitate the extraction of meaningful features. Such representation is individual events [Kim et al., 2014a, Gallego et al., 2018], Event packet [Rogister et al., 2012a, Rebecq et al., 2018], 2D histogram [Cook et al., 2011, Liu and Delbruck, 2018], Time surface [Lagorce et al., 2017, Afshar et al., 2019b], Voxel grid [Bardow et al., 2016, Wang et al., 2019], 3D point set [Sekikawa et al., 2019], Point set of image plane [Litzenberger et al., 2006b, Zhenjiang Ni et al., 2012] and Events to frames reconstruction [Scheerlinck et al., 2018]. The camera mode of operation heavily influences the methods used to process events, and the subjects exist in the scenes, such as sparse vs dense scenes. As a result, different types of representation motivated different types of feature extraction. Camunas-Mesa et al. [2012] proposed

an event-driven convolutional module for computing a set of 2D convolutions on such event streams, which were assembled in a hierarchical multi-layer convolutional network. Their proposed convolutional kernel selects the convolution kernel depending on the event’s origin. Sironi et al. [2018] proposed HATS which make use of a local memory time surface, which divides the image into a regular grid of cells and uses a histogram of averaged time surfaces to extract features, followed by a standard classifier. Lagorce et al. [2017] proposed HOTS, introducing event-driven spatio-temporal features extraction using time surfaces event representation and combined with unsupervised feature extraction and classification to form an event-based convolutional network which was named Hierarchy of Time Surfaces (HOTS), where each neuron in the hierarchical architectures learn features based on local learning rules inspired by the human visual cortex. Afshar et al. [2019c] proposed FEAST proposed a local adaptive threshold that guarantees homeostasis between the learning and activity of all neurons, and the threshold adaptation enables equal learning among all the neurons. The main key property of event-based feature extraction is to be simple and less computationally expensive to enable efficient implementation of the algorithm in neuromorphic hardware.

2.3.3 Event-based Object Detection

The wide adoption of EBCs and the last decade of development in the field of computer vision has motivated the development of a range of event-based algorithms for object detection and recognition. These algorithms often consist of several stages such as event pre-processing, core processing (feature extraction) and post-processing. Depending on the processing method, some algorithms are natively event/spike-based, and some are considered an adaptation of DL algorithms with event data as an input. The former achieves lower latency by preserving the temporal resolution and exploiting the sparsity of the events, and the latter enables the re-utilisation of image-based computer vision tools leveraging more than 40 years of computer vision research.

On the one hand, a few works [Alonso and Murillo, 2018, Maqueda et al., 2018, Zhu et al., 2018b] approached object detection by mapping the events stream to a dense representation, thereby enabling the use of standard DL architectures (i.e. gradient descent). Rebecq et al. [2019], Ronneberger et al. [2015] used a recurrent network to reconstruct high-quality frames from the events and used it in image Segmentation. Cannici et al. [2019] implemented an event-driven YOLO network [Redmon et al., 2016] using an asynchronous CNN network. Li et al. [2017] used faster-RCNN network [Ren et al., 2016] with temporally pooled binary images reconstructed from the event camera. However, these methods add a further computational step which loses the events temporal resolution, but gains in terms of accuracy and scalability.

On other hand, Kaiser et al. [2020] combined the plasticity of Spiking Neural Network (SNN) with the scalability of ANN using local learning rules [Mostafa et al., 2018], and surrogate gradient descent [Neftci et al., 2019]. This enables gradients to be computed locally at each layer. Some methods [Kasabov et al., 2013, Lee et al., 2016] uses

SNN to exploit and preserve the temporal resolution of the events stream. However, applying these approaches to large and noisy events remain difficult, and their efficacy has mainly been demonstrated on classification tasks using datasets with lower spatial resolution.

Despite the notable advantages of event cameras, there remains a significant performance gap between event-driven algorithms and their frame-based counterparts for various vision problems. This is partly due to a requirement of totally new event-by-event processing paradigms. However, the recent event-based detection focuses on closing the gap using deep SNN [O'Connor et al., 2013].

2.4 Deep Learning for Agriculture Applications

Detection, counting, and localising fruits in orchards are essential tasks in agriculture automation. They allow farmers to manage and optimise resources and make critical decision making during harvest. Researchers have used a variety of sensor technologies to tackle the problem of fruits detection, including RGB/RGB-D camera, laser sensor, thermal imaging sensor, and spectral imaging sensor. These tedious and laborious tasks (i.e. fruit picking) are still mainly performed by manual efforts. Automating plants phenotyping and fruit counting is required to meet the large-scale genotype and phenotype analysis. A robust system consists of an efficient fruits detector because the detection step is executed before performing the picking. For instance, if the fruit is not detected or seen, it cannot be picked.

Computer vision and DL have experienced significant breakthroughs due to the availability of large scale datasets and advancement in computational power, and image-based recognition received lots of attention in fruit-related studies. In recent times, there exist several methods in the image-based detection of fruits: conventional machine-learning-based algorithms [Gongal et al., 2015] and deep-learning-based algorithms [Sa et al., 2016, Roy and Isler, 2017]. The former method uses the image feature descriptors to encode the feature information and then apply the machine-learning-based classifier to perform the segmentation or detection of the fruit within the image.

A detection pipeline is comprised of three different steps: image capturing and annotation/labelling, feature extraction and object detection. However, to extract features such as shape, appearance, colour etc., sparse coding and multi-kernel learning methods need to be applied, which require precision engineering and domain expertise. Many expert-coded feature descriptors have been adopted to be used for agricultural application, such as the histogram of gradient [Dalal and Triggs, 2005], colour coherence vector [Pass et al., 1996], and local binary patterns [Ahonen et al., 2006]. Zhou et al. [2012] proposed a logistic regression classifier based on colour features to detect apples in an unconstrained environment like an orchard. Song et al. [2014] used a bayesian classifier with a support vector machine algorithm to learn the colour and texture features, and it was used to detect pepper with an RGB camera. Luo et al. [2016], Wang et al. [2018] used the colour-based and texture-based features and used an AdaBoost classifier to perform fruit detection.

Deep learning approaches demonstrate their effectiveness by automatically extracting features from the input image and learning low-level to high-level features represented in the image, eliminating the need to hand-craft features. This has helped achieve more robust and superior performance than conventional machine learning approaches. Sa et al. [2016] utilised a faster-RCNN network to detect peppers, rock melons and apples. Bargoti and Underwood [2017] adopted a faster-RCNN network to detect apples and mangoes in orchards. Yu et al. [2019] utilised mask-RCNN network to perform the detection and segmentation of the strawberry in the greenhouse. These methods use the classification accuracy as an evaluation metric to evaluate the network as shown in Table 2.2, which summarises the performance of various DL architectures with their corresponding state-of-the-art results.

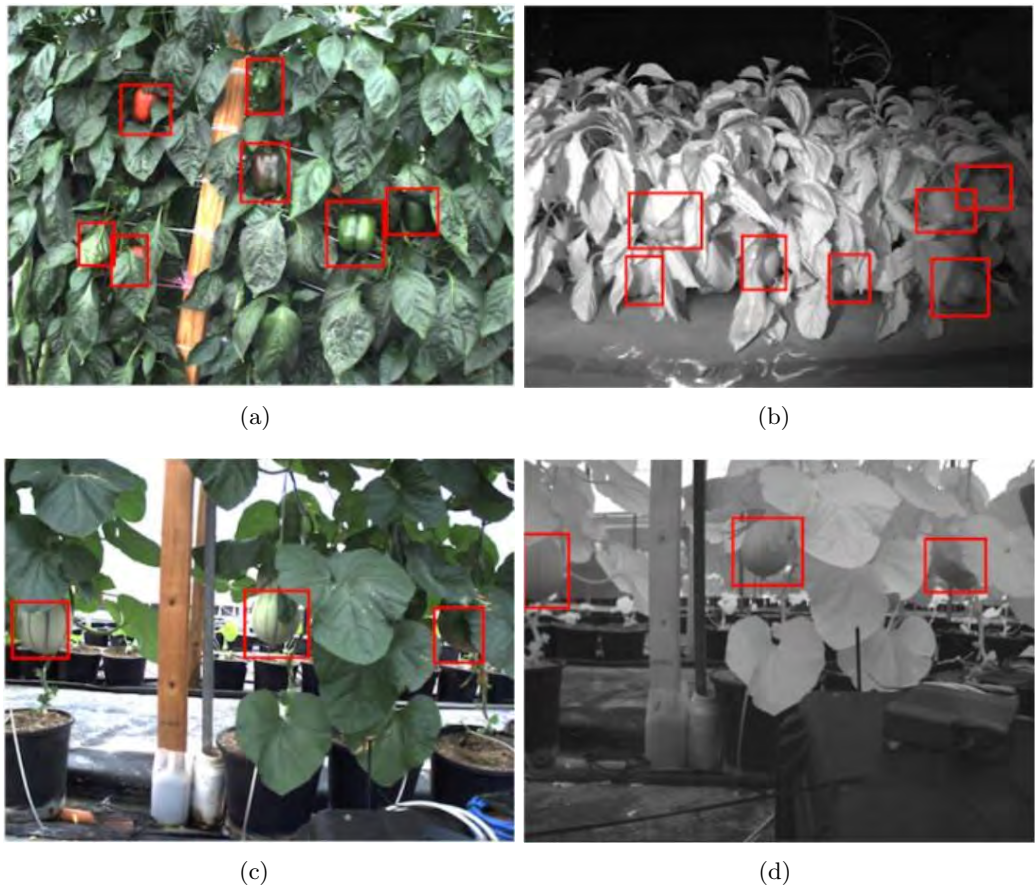


Figure 2.6: Fruits detection using colour and near infrared vision sensors. (a) and (b) show the detection of sweet pepper with the output bounding boxes. (c) and (d) show the detection of rock melons with the output bounding boxes.

Most of the frame-based recognition systems are deployed under a controlled laboratory environment. When the system is transferred to the real-world environment, intrinsic and extrinsic variations in the wild pose significant challenges, this can significantly deteriorate the detection system, as shown in Figure 2.6. Such challenges become more problematic, especially in a real-time real-world application. These challenges can be mostly boiled down to the variations in the field-based environment such

as:

1. Fruits emerge over time and vary significantly in shape and size as plants/trees grow.
2. Fruit exhibits a similar visual appearance to the background as shown in Figure 2.6.
3. Different cultivars of fruits exhibit different appearance variations, such as colour and texture.
4. Illumination changes dramatically due to the different weather conditions, especially during a sunny day.
5. Image angle and perspective distortions due to the wind cause various pose variations.
6. Occlusions frequently occur, which renders the difficulty of counting even for a human expert.
7. The cluttered background make visual patterns of fruits diverse and misleading.
8. The quality of the image degrades because of the dust or raindrops on the camera lens.
9. Textural patterns also change essentially due to different flowering statuses.

To overcome these, a well-generalised model that is invariant and robust to brightness and viewpoint changes and highly discriminative feature representations are required.

Table 2.2: State-of-the-art results of deep learning in agriculture applications.

Ref.	Problem	Proposed model	SOTA
Ha et al. [2017]	Classification	CNN-based	97.4%
	healthy and Fusarium wilt of radish		
Ma et al. [2018]	Recognition of cucumber diseases	DCNN	93.4%
Lu et al. [2017]	Recognition of rice diseases	CNN-based	95.48%
Liu et al. [2017]	Identification of apple leaf diseases	AlexNet-based	97.62%
Zhong et al. [2019]	Crop classification	LSTM and Conv1D	85.54%

Mehdipour Ghazi et al. [2017]	Plant identification	Fine-tune VGGNet, AlexNet, and GoogLeNet	80%
Dias et al. [2018]	Apple flower detection	CNN-based	90%
Rahnemoonfar and Sheppard [2017]	Fruit yield estimation	Modified Inception-ResNet	91%

Chapter 3

INVESTIGATION OF NOVEL EVENT-BASED DATA USING THE COLOUR EVENT-BASED VISION SENSOR

3.1 Introduction

Colour is created by using two properties of light, energy and frequency of wavelength. How our brain separates and recombines them into colour perception is still a mystery [Gouras].

Colour provides essential information about the environment. It is utilised to determine whether the water we drink is clean or whether an apple is ripened. Most illuminants and surfaces have broad spectra that contain many wavelengths. Hence, the perception of colour depends to no small degree on other colours in the whole scene, and by taking all colours into account, the visual system can discount changes in illumination (e.g. colour constancy) [Viqueira Pérez et al., 2010], and compute the colour closely related to the reflectance spectrum of a surface. Various visual processing stages interact to extract a robust estimate of the reflectance spectrum, giving rise to the sensation of colour. The first step in sensing colour occurs in the retina, where three types of cone photoreceptors measure light. Then, the retina visual input is transmitted in these three channels via the Lateral Geniculate Nucleus (LGN) to the cortex to form colour categories. This makes colour perception a multi-stage process, involving - at different degrees - all visual cerebrocortical areas.

Although colours can provide valuable information, there is no clear evidence of how strong variations in colour information influence classification performance [Buhrmester et al., 2019, Funt and Zhu, 2018]. Recognition algorithms are originally developed on static frames and videos because colour frames are well understood with many decades of research and development. Colour events have been rarely studied, and their benefits and applications are not yet apparent. The advantages of EBC make them well-suited

candidates to overcome existing limitations in computer vision. For example, they are power-efficient, generating sparse events with spatial and temporal rich information. Their high temporal resolution allows for various simplifying assumptions, with complex behaviours emerging from simple, high-speed algorithms. However, the colour events do not encode absolute luminance information, that is because the colour measurements of the DAVIS pixels are performed only on APS side in a frame-based fashion. Therefore, they do not benefit from the event-based approach’s advantages.

This study aims to investigate the novelty of the pixels response from the CDAVIS sensor, in a view to expanding our understanding of how these sensors work and how they respond to different achromatic stimuli. The goal is to determine the appropriate parameters to record colour events data more efficiently. These parameters are the camera settings (i.e. biases), scene illumination (i.e. light source), and colour contrast between surfaces (i.e. stimuli). The sensor’s internal circuit voltage, photodiode current, quantum efficiency, and activity leaks were out of the scope of this work. Instead, the main focus is to observe and analyse the events read out to characterise their response. Several characteristics of the DVS sensor are identified and measured without asking the manufacturer to provide them exhaustively. Measurements of characteristics such as noise, contrast uniformity, and spatio-temporal statistics are performed on the whole area of the imager. Also, it is investigated across a wide range of parameters that we can control, such as camera threshold, lighting condition and objects contrast ratio.

We began by (i) investigating the SNR from the DVS sensor to get a better estimate of the noise produced by colour filters, (ii) characterising the APS to get a better estimate of the colour quality using a colour checkerboard, (iii) quantifying the spatial uniformity of the sensor to observe its responsiveness and sensitivity, (iv) investigating the pixel temporal noise using the total number of events and the event frequency (events/s) under different illumination and contrast threshold conditions in uniform and noiseless scene, (v) characterising the sensor based on chromatic and achromatic stimuli, and (vi) investigating the use of events for multiclass classification tasks with KNN algorithm. All experiments were performed by considering that the light spectrum is not perfectly linear and considering that colour stimuli might contain a mix of different colours due to the colour mix in the printer ink.

A complete characterisation study enables the development of a better simulation model for the Complementary Metal Oxide Semiconductor (CMOS) retina, which in return increases the databases necessary to validate a vision algorithm for these new imagers, and ultimately recording more efficient datasets that are application-specific, and ultimately contributing toward building better colour EBCs. Appendix B we show the initial prototype of the colour sensor consisting of a two-chips camera assembly with dichromatic vision capabilities.

3.2 Research Questions

Based on the literature and the current understanding of the colour event-based spikes, below are the main research questions:

1. How can the colour differences between surfaces change the DVS pixel response?
2. Is it possible to correlate the colour of a given stimulus and the events triggered by the colour filters?
3. Can the total number of events triggered by each colour filter be used to resolve the intrinsic properties of objects such as actual objects' colours?
4. Is colour information relevant in event-based classification and object colour discrimination?
5. What are the advantages and shortcomings of the colour event-based vision sensor?

3.3 Contribution

The benefit of colour events in its infancy, and the work on spiking colour sensors has been rare, and there is little existing work on the subject of colour events processing and analysis [Marcireau et al., 2018]. This section includes several techniques and makes the following contributions to the existing body of knowledge:

- Builds an experimental setup to facilitate data collection and analysis using a linear slider platform with optical equipment to reduce the external noises.
- Provides an in-depth explanation into the mode of operation of the CDAVIS using primary colours as a source of stimuli.
- Characterises and analyse the CDAVIS camera in terms of biases/settings and external parameters such as objects colours and scene illuminations. Moreover, studying the behaviour of the DVS pixels and the APS image.
- Derives and demonstrates a mechanism for performing classification on the colour-event data using the internal properties of the events stream from each colour filter.

3.4 Materials and Methods

This section describes the structure and nature of the colour events-based sensor, the software-hardware setup to operate the pan and tilt platform, the method used to label the colour pixels and the measures used to perform the detailed characterisations on the DVS and APS pixel.

3.4.1 DVS Colour Pixel: CDAVIS

The colour Dynamic and Active Vision Sensor (cDAVIS) or colourDAVIS [Taverni et al., 2018] is the first commercially available sensor that combines a DVS and APS pixels patterned with an RGBG Bayer filter array. The sensor concurrently outputs rolling or global shutter RGBG coded VGA resolution frames and asynchronous RGBG coded QVGA resolution temporal contrast events (see Figure 3.1). The CDAVIS follow the same design rules described by Taverni et al. [2018] where the Back Side Illumination (BSI) circuit design was proven to outperform the Front Side Illumination (FSI) in terms of quantum efficiency and pixel’s form factor. In this design, the DVS part outputs a stream of brightness change events, and each event signals a change of log intensity $\Delta \ln I_p$ exceeding a pair of temporal contrast thresholds $\Theta_{on} > 0$ and $\Theta_{off} < 0$ as shown in Equation 3.1 and Equation 3.2.

$$\Delta \ln I_p > \Theta_{on} \quad (3.1)$$

$$\Delta \ln I_p < \Theta_{off} \quad (3.2)$$

The pixels store the value of $\ln I_p$ after the event is sent. The readout is a variable data-rate stream of events consisting of the addresses of the pixels and the signs of the brightness change (i.e. polarity). However, each colour filter can produce a different change in brightness related to the amount of contrast and the type of colour in the scene. For instance, the sensor can be more sensitive to a specific colour in the scene for a given contrast threshold. In this case, the number of events and noise profile triggered will vary between each colour filter. For example, Figure 3.2 show the correlation between the number of events for each colour filter with the contrast threshold. (a) and (b) show the space-time plot of the event stream under low and high contrast thresholds. When the contrast threshold increases, the number of events is reduced, and the rigid edge between surfaces becomes visible. It is evident that the red circle stopped triggering events for the red filter, that is because the log pixel illuminance is affected by the colour filter as shown in Figure 3.2(e) and 3.2(f), resulting in no change in brightness for the red circle. This show that the DVS readout for each colour filter is highly affected by the sensor threshold as well as the lighting conditions and the contrast ratio between objects.

The motivation behind adding colours to the EBC is to help us tell apart objects that are otherwise identical as well as facilitate post-processing by increasing the separability of features.

The Bayer filter on the DVS pixels does not capture the absolute illuminance information, but it improves the contrast depending on the colour differences between the figure and the background. For instance, when white light shines on a red object, all white light colours are absorbed except red, which is reflected, making the object

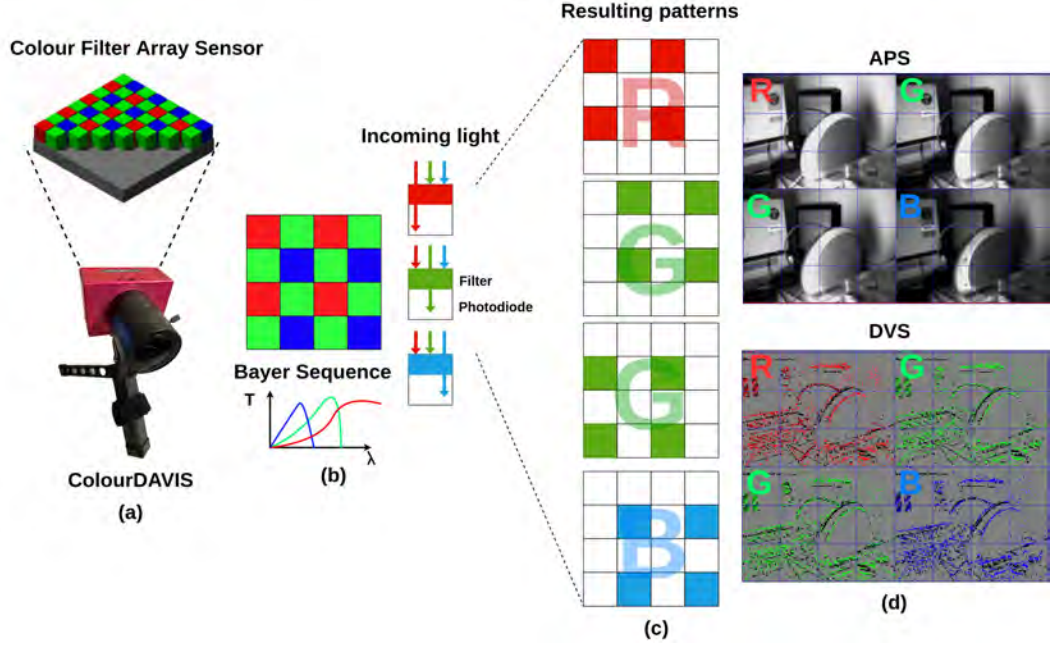


Figure 3.1: A detailed view of the colour Dynamic and Active Vision Sensor (cDAVIS). (a) show the actual look of the sensor and an abstract overview of the colour filtering array. (b) An example of the RGBG Bayer filter shows the relative sensitivity with the filter wavelength, where the incoming light only penetrates through the filter depending on its wavelength. (c) The resulting pixel-wise patterns when the filter is separated into four colour channels. (d) The resulting patterns are observed through the DVS and APS sensors.

appear red. A green colour filter will only let green through and absorb all other colours in a similar case. Hence, when the blue light is allowed through a blue filter onto a blue object, the object will still reflect blue and appear blue. However, when blue light hits a red object, the blue will be absorbed, and little light will be reflected, then the object will appear black. Another example, when a red light hits a red filter, a red object appears white. In contrast, a blue object appears black due to the light absorption and reflection as shown in Figure 3.3 which illustrates the appearance of primary colours through each colour filter and demonstrate colour absorption.

From the CDAVIS perspective, a combination of two or more colours will result in pixels sensing a change in the energy at particular frequency. This makes colour DVS pixels a beneficial sensor to sense the contrast difference between colours at different frequency band.

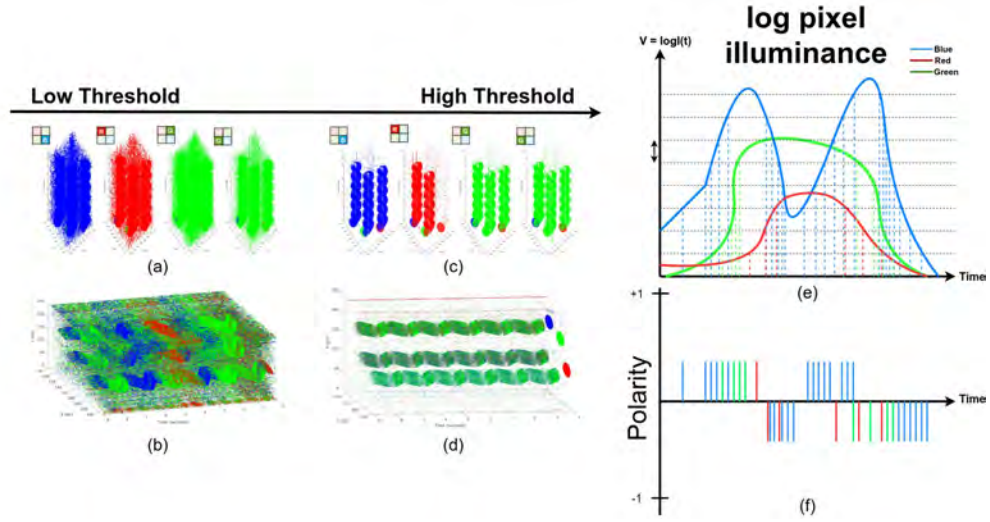


Figure 3.2: The CDAVIS is moving in front of three circles with different colours (Red, Green and Blue) on a white background in an illuminated condition. (a) and (c) Separately showing the events triggered by each colour filter of the Bayer matrix and for each circle using a high and low contrast threshold, respectively. Because of the high threshold, the red filters are not generating any events for the red circle, while under a low threshold, the events are noisy. (b) and (d) All the events triggered by all colour filters combined. (e) An example of the log pixel illuminance response for each colour filter resulting in ON and OFF events is shown in (f). If there is a positive change in the pixel illumination, an ON event will be triggered and vice versa.

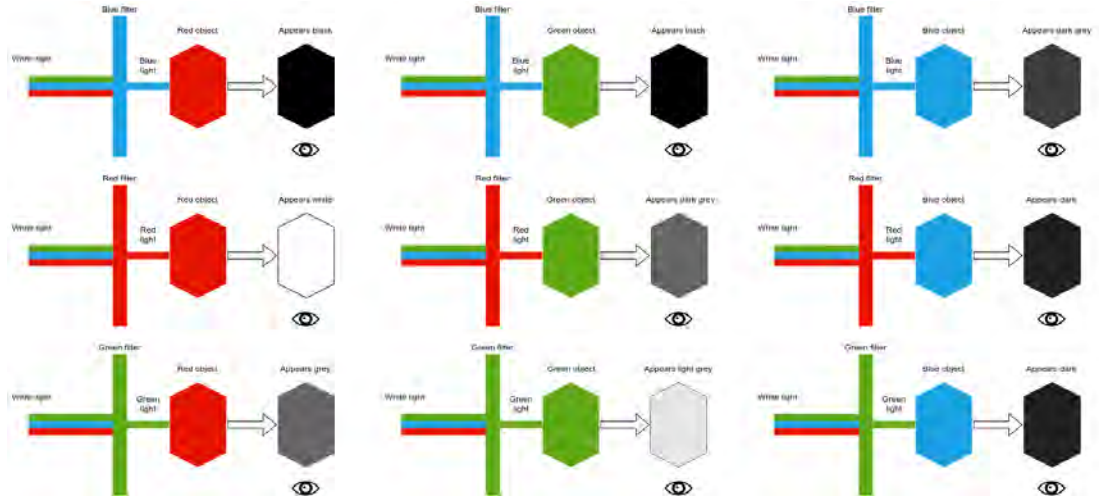


Figure 3.3: Appearance of primary colours through each filter in the bayer matrix.

3.4.2 Hardware Assembly

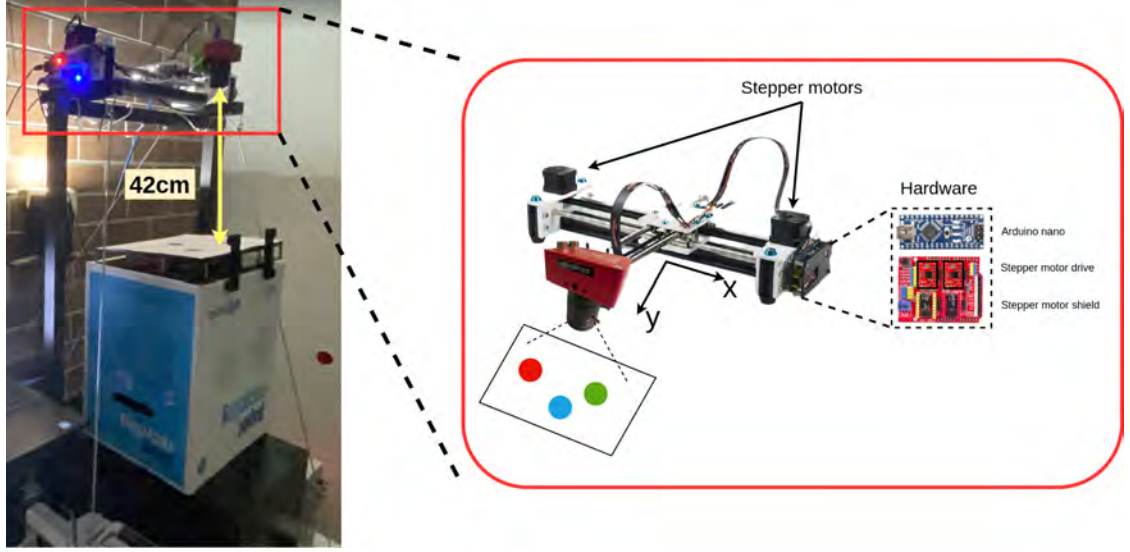


Figure 3.4: Hardware setup. It shows the mechanical assembly of the pan and tilt platform and the position of the CDAVIS in relation to the stimuli.

The setup is built in a view to converting static images into spikes events. The conversion system relies on the CDAVIS sensor for recording. To control the motion of the CDAVIS, we constructed our pan and tilt mechanism. The mechanism consists of two hybrid phases 42H34S-1304A stepper motors connected through an elastic toothed belt. The motors interface directly to an Arduino Nano ¹ with two MP6500 stepper motor driver ² connected to a custom carrier board from EleksMaker. Each motor allows programming of a target position, speed and acceleration. A custom housing for the CDAVIS including lens mount and a connection to the pan-tilt mechanism was 3D printed. The motors themselves in an enclosure made from acrylic materials. A toothed belt is attached to two pulleys mounted on the stepper motor and a bearing system on the centre of the platform to allow the camera to move in the x and y-axis.

A static ink-based display ³ was used to display static patterns with colours. Using an ink-based display and not an LCD eliminates the refresh rate generated by the LCD screen frame rate, which includes unnecessary noise because motion on the screen is discontinuous, consisting of discrete jumps in position at each monitor update. These discontinuities are visible in the data as described in Orchard et al. [2015], whereas in the ink display, the pixels are static-filled with ink without and do not require any frame update. Since the pattern is static, the camera moves to trigger a change in contrast and generate events.

The distance between the sensor and the stimuli was adjusted to 42cm to give the platform a sufficient height to line up with the vertical centre of the pattern. The whole platform was initially used as a CNC drawing robot ⁴, but extensive modifications have

¹<https://www.arduino.cc/en/pmwiki.php?n=Main/ArduinoBoardNano>

²<https://www.pololu.com/product/2966>

³<https://github.com/neuromorphicsystems/epaper>

⁴<https://wiki.eleksmaker.com/doku.php?id=eleksdraw2019>

been applied to it to make it suitable for our experiment. To remove all the external noise from the fluorescent light source as much as possible, the platform was placed in a dark room with all the external light blocked. We used the non-flickering LED OSOLON SSL 80 LUW CR7P (EQW) ⁵. The high flux, high efficacy, and low thermal resistance make the light source more stable and uniform than the standard fluorescent light. Besides, it also removes the light flicker, which the EBC can detect due to the low latency. Figure 3.4 illustrates the hardware assembly.

3.4.3 Software Stack

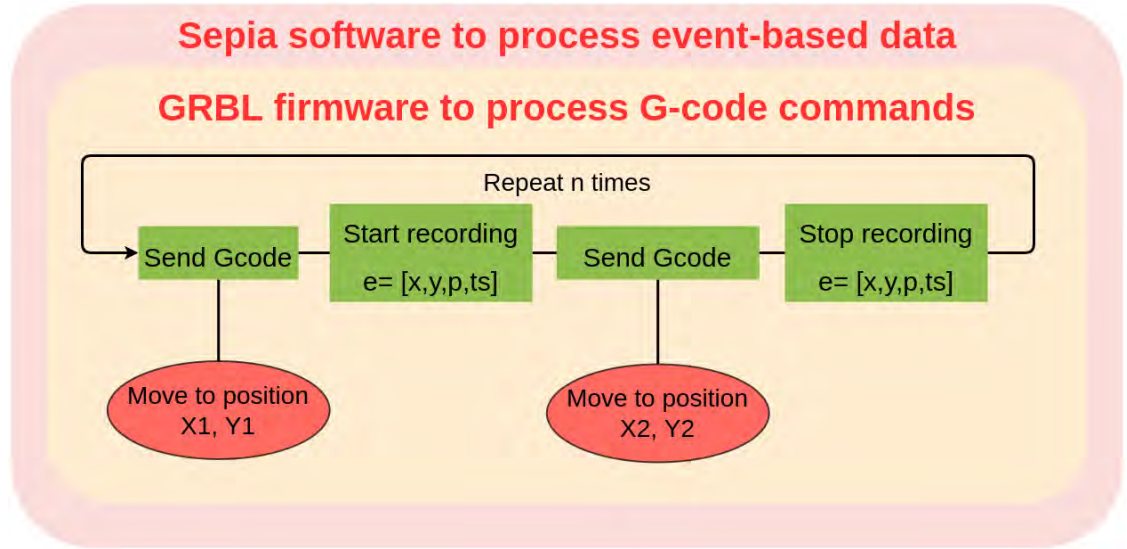


Figure 3.5: Software stack. It shows a simplified flowchart of the whole process. In this case GRBL firmware is used to process G-code commands to the pan and tilt platform and sepia to process event-based data.

The software stack is used to drive the platform was based on the Sepia framework developed by [Marcireau et al.] which is a header-only and modular C++ framework in a view to facilitating the implementation of event-driven algorithms. GRBL ⁶ firmware is integrated within Sepia to send commands to the platform. Sepia is an input/output library for event-based devices that facilitate communication between devices and EBCs, such as reading and writing events data to an event stream file.

GRBL is a parallel-port motion control firmware for CNC milling machines that supports Atmega328 ⁷ microcontroller built-in Arduino board, GRBL is used due to its look-ahead capability, which allows the controller to look at 18 motions into the future and plan its velocities to deliver smooth acceleration and jerk-free cornering. Then, G-Code commands can be sent directly from the host PC to the Arduino through Sepia.

The sepia framework is running on the host PC interface with the pan-tilt robot and the CDAVIS. Another separate thread controls the display of images on the ink display.

⁵<https://docs.rs-online.com/f9ce/0900766b815fe2fd.pdf>

⁶<https://github.com/grbl/grbl>

⁷<https://www.microchip.com/wwwproducts/en/ATmega328>

Since the data recording thread and the images display thread was not connected, the images were displayed on the ink display first, followed by the Sepia framework. The frames readout were disabled to prevent any interference between the DVS and APS output stream, as it is known that when the APS frame generation is active, it can change the behaviour of the events adding unnecessary noises.

At the beginning of the initialisation state, Sepia connects to the camera through libUSB (a library that provides generic access to USB devices), an empty .es file ⁸ is created locally in the specified directory, Sepia then send G-code commands to the Arduino board. The G-code includes setting for the initial velocity and acceleration and the coordinates of the initial and final position of the camera. For simplicity, the camera was only moved from left to right and then right to the left in a periodic motion. The camera's motion is linear and not random to better understand the pixel from one colour to another. Once the first G-code command is sent and a check is performed to ensure that the request is received, the events data will be recorded and written to the events file. After the last G-code command is received, the same check operation is applied and then the software stop recording events to the file.

Sepia repeats the process of recordings events depending on the user input, which includes how much data needed to be recorded, and at each run, a new event file is created separately. A wait state of two seconds is performed between each recording cycle. This automated process of data collection ensure reproducibility and repeatability of the experiments and that each event data contains the same amount of data with the same recording duration removing all the external noise. Figure 3.5 illustrates the hardware assembly.

3.4.4 Colour Pixel Labelling

The CDAVIS events readout is similar to its counterpart, such as the DAVIS. The events stream consists of an array of the locations of the active pixels (x and y), the direction of change of brightness (polarity), and the timestamp on the microsecond scale. However, pixel's data do not contain absolute luminance information. To do this, we implemented pixel labelling to indicate the colour index for each pixel, this imply converting the colour information to spatial location due to different addresses of colour pixels. Based on the Bayer matrix (RGBG) position over the pixel array, it became trivial to know which pixel belongs to each colour filter as shown in Figure 3.1. The colour labelling method works as follows:

- If the x and y pixel positions are odd, this pixel is blue.
- If the x and y pixel positions are even, this pixel is red.
- If the x position is even and the y position is odd, this pixel is green1.
- If the x position is odd and the y position is even, this pixel is green2.

⁸https://github.com/neuromorphicsystems/event_stream

Using this method we were able to add another dimension to the event stream in post processing as shown in Equation 3.3 below:

$$e_i = [x_i, y_i, p_i, t_i]^T \rightarrow e_i = [x_i, y_i, p_i, c_i, t_i]^T, i \in \mathbb{N}^+ \quad (3.3)$$

where c is the colour index of each pixel. For red pixel $c=1$, for blue pixel $c=2$, for green1 and green2, c is 3 and 4 respectively.

3.4.5 Characterisation Measures

The characterisation measures proposed in this section aim at measuring the main characteristics of the colour DVS pixel under an indoor environment to estimate the overall response of the sensor. The primary goal is to characterise the sensor based on parameters listed in Table 3.1.

Table 3.1: Characterisation measures and intended use case

Characterization Parameter	Usage
Signal to noise ratio	Assessing the signal quality and get a better estimate of the noise produced by colour filters
Chromaticity Error	Measuring the colour pixel values from the APS to get a better estimate of the colour quality
Spatial Uniformity	Assessing the sensor's spatial response and sensitivity
Pixel Temporal Noise	Assessing the variation of the events stream over a range of threshold and illumination
Chromatic and Achromatic Contrast	Evaluating the event stream in chromatic and achromatic spaces

3.5 Results

In the following sections, we provide the results for each characterisation measure described in Section 3.4.5 and interpret the results to highlight to contribution for each experiment. The section further provides a detailed analysis of a KNN algorithm applied on the colour events for classification tasks.

3.5.1 Signal to Noise Ratio

Our interest lies in extracting moving objects and measuring the noise produced in each colour filter. The signal is defined as the events captured by each colour filter when the object is in motion. Noise is considered to be events or activities present when the object is not moving. The EBCs not only capture the change in the light

intensity at a location due to moving objects but also produces some noise activity at various pixel locations due to uneven illuminations (e.g. bright spots or shadows) or slight movement of background objects and noise generated by the actual circuits. Computing the SNR has the benefit of filtering away all unwanted noise and extracting events from the object of interest for classification tasks. That is because it allows us to understand the noise profile through time which can be eliminated.

The SNR is calculated by recording the stimulus in two different states: during motion, and when the object is static. The number of events is calculated over the same time interval for each state. Then the logarithm with base 10 of the ratio of the events when the object in motion to the number of events when the object is static is taken as shown in Equation 3.4 [Padala et al., 2018].

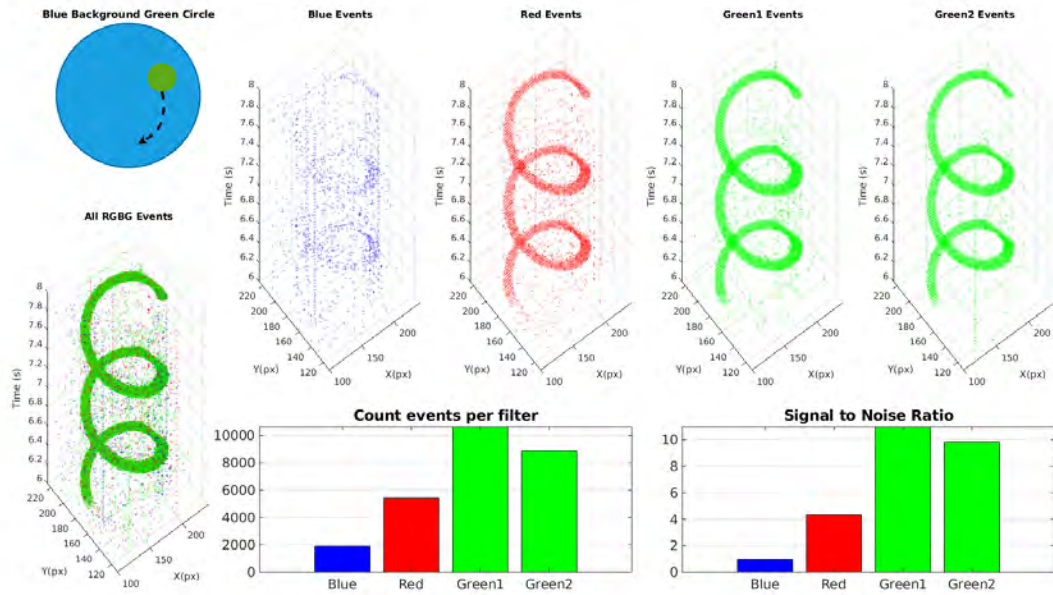
$$SNR = 10 * \log_{10}\left(\frac{E_{Signal}}{E_{Noise}}\right) \quad (3.4)$$

Where E_{Signal} is the number of events (i.e. pixels) generated by the object and E_{Noise} is the number of events generated by the noise. To quantify the signal against noise events, the SNR was compared with the total number of events from each colour filter to investigate any possible correlation between the two measures. To do that, we considered using colour transition patterns using a rotational disk as shown in Figure 3.6, where the colour of the rotated circle and the background changes to trigger contrast change during motion. The platform’s motion was stabilised to ensure a consistent event stream during data collection. Due to the infinite number of colours in the world⁹ we only used the primary colours such that a combination of two primary colours is used to trigger contrast change. A brushless motor was used and rotated for 2 seconds under constant illumination of 53 lux.

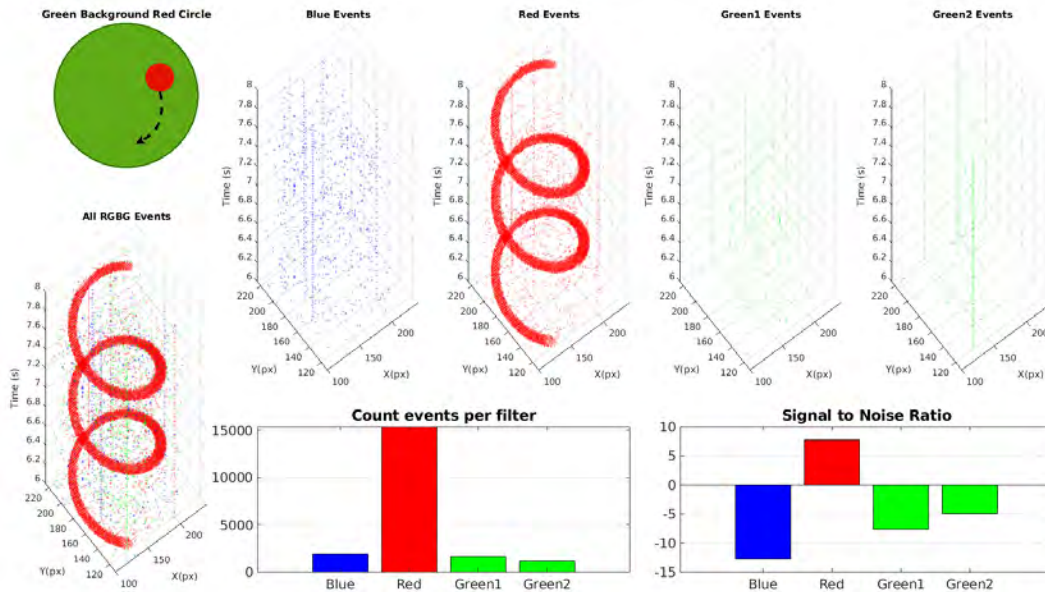
Figure 3.6, show the events output from each colour filter for two recording conditions. In Figure 3.6(a), a green circle over a blue background is presented to the CDAVIS as input stimulus. We measure the response to blue \rightarrow green \rightarrow blue across the four channels. Results show that the red, green1 and green2 filters were spatially denser and had more spatial contrast events representing the actual shape of the circle, whereas the blue filter had the lowest event activity. This is due to the contrast change in each filter which was illustrated in Figure 3.7, although the blue colour is dominant in the scene, the contrast difference between the blue and green was very low, causing the blue filter to trigger only noise events, in contrast to the red filter and green1/green2 filters where the contrast was relatively higher. In this example, there is a strong correlation between the total number of events and the SNR.

A red circle over a green background was shown to the camera, as shown in Figure 3.6(b), to allow green \rightarrow red \rightarrow green colour transition. As shown in the space-time plots, only the pixels with the red filter triggered events, whereas pixels with the blue and green1/green2 filters trigger only noise events. That is because the red circle appears brighter through the red filter, and the green background appears dark through

⁹<http://markfairchild.org/WhyIsColor/files/ExamplePage.pdf>



(a)



(b)

Figure 3.6: Spatio-temporal patterns for each colour filter with the signal to noise ratio. (a) shows the pixel responses using a rotating green circle on a blue background. (b) shows the pixel responses using a rotating red circle on a green background.

the red filter due to colour absorption, which only causes a higher contrast on the red filter during motion and very low contrast for the green and the blue filter as both surface colours appear dark. Based on the observations, we found a strong correlation between the number of events and the number of events that belong to the actual signal.

These experiments showed that even under controlled parameters such as illumination and camera biases, there was no correlation between the actual colour of the stimulus and the number of events triggered by each filter, making it impossible to retrieve the actual colour purely based on the events stream. The same concept can be applied to other colour filters. It becomes challenging to identify the actual colour of the stimulus purely from the events rate and events internal properties (e.g. number of ON events, number of OFF events, ON/OFF ratio, etc.). Several factors contributed to this colour pixel behaviour such as, the colour patterns were generated using printer-based inks in which the primary colours are generated from an additive of two or more colours, non-linearity of the light source and colour temperature, the value of the contrast threshold, and the mismatch in the photodiodes.

Theoretically, it will be possible to generate colour events based on data based on the object’s true colour, but under a rigorous environment with very constraints, parameters are considered rare in real-world scenes. The same experiment was repeated using different combination of primary colours as shown in Appendix C from Figure C.1 to Figure C.7.

The results presented in this section have been obtained with a set of fixed parameters for all recording conditions given the lighting condition. In particular, the camera’s internal electrical components were tuned via a set of analogical values called ”biases”, which correspond to a list of voltage values that set the operating conditions of the electronics. Depending on their values, the property of the pixels such as the detection threshold, the latency, the refractory period and many others may change. For our experiments, we used a fixed set of biases with the hypothesis that the camera’s parameters should remain unchanged when the colour of the patterns changed.

In this section, we found and learned that there is a significant variance between the response of each colour filter depending on the colour of the background and the object chosen. We also saw that the noise coming from each filter also varies with the colour. The colour filters appeared to be sensitive to patterns with different colours, however, it was not possible to see a correlation between the colour of the object and the event readout mainly because a slight tweak to the colour of the background also affects the event readout.

Appendix D shows additional examples of colour event data in various scenes where each pixel is represented according to its colour index.

3.5.2 Chromaticity Error

The human visual system can efficiently discount the colour of the incident light when interpreting objects in the scene (i.e. colour constancy). However, from the perspective of a CMOS camera, the same surface can appear different in images captured under illuminants with different colours. The primary purpose of this study is to characterise the response of the APS and examine how well it can represent different colours under constrained illumination with different exposure times. Given that this chapter focuses on DVS pixels, the APS characterisation is considered helpful for calibration purposes

because the DVS pixels cannot see non-moving objects, hence making the focusing of the lens harder.

Efficient red, blue, green filters ensure that the APS and DVS pixels are sensitive to the visible wavelength and that the filter can interpret the colour efficiently. Since the APS provide us with the absolute intensity of each pixel, it makes more sense to use the APS to characterise colour stimuli. To do that, we used the colour checkerboard from Macbeth ¹⁰ (see Figure 3.7(a)), which has 24 squares of painted samples, as an input stimulus and computed the chromaticity coordinates of each colour patch and then compared it with the actual coordinates from the CIE colour diagram ¹¹. These colour patches have spectral reflectances intended to mimic natural object such as human skin, foliage and flowers, for the consistency of the colour appearance under various lighting conditions.

The chromaticity diagram (i.e. CIE XYZ or CIE 1931) maps human colour perception based on two CIE parameters. Consequently, it can be expressed on a 2D chromaticity diagram as shown in Figure 3.7(c). It provides us with two crucial measures that correlate with perceptual attributes, such as hue and saturation, which provide a visual understanding of the properties of colours. It also contains a white point reference related to the colour of the object and not its intensity; The white point of an illuminant is the chromaticity of a white object under the illuminant. However, it does not correspond uniquely to only one illuminant. In addition, the white point can help in calibration, colour constancy and gamut mapping. This diagram also shows all the hues perceivable by the standard observer for various (x,y) pairs and indicates the spectral wavelengths of the dominant single frequency colours. The colour space in CIE allow us to measure the tristimulus values, X, Y and Z, which are a device-invariant representation of colour, that is because the CIE colour space encompasses all colour sensations that are visible to a person with average eyesight ¹².

The process begins by pointing the CDAVIS toward the colour checkerboard. A white non-flickering LED at a fixed intensity was used to minimise the external noise caused by fluorescent lights. jAER software ¹³ is used to control the CDAVIS and record frames and store the data locally. Fifty-five experiments were conducted where the shutter speed (i.e. exposure time) was the only variable component. In this case, the exposure time was set to be from 5 ms to 200 ms, where low exposure allows less light to come in and the scene appears dark, whereas high exposure allows more light to come in and the scene appears brighter. One frame from each recording was selected. Since we were only interested in how the colour patches were represented through the APS, we selected a spatial window of 21x16px over each colour patch to exclude the black lines that separate the colour patches and the background colour. We converted the colour space of the cropped colours patches from sRGB to CIE.XYZ colour space and computed the mean for each colour channel over each colour patch. The mean of

¹⁰<https://poynton.ca/notes/color/GretagMacbethColorChecker.html>

¹¹https://en.wikipedia.org/wiki/CIE_1931_color_space

¹²https://en.wikipedia.org/wiki/CIE_1931_color_space

¹³<https://github.com/SensorsINI/jaer>

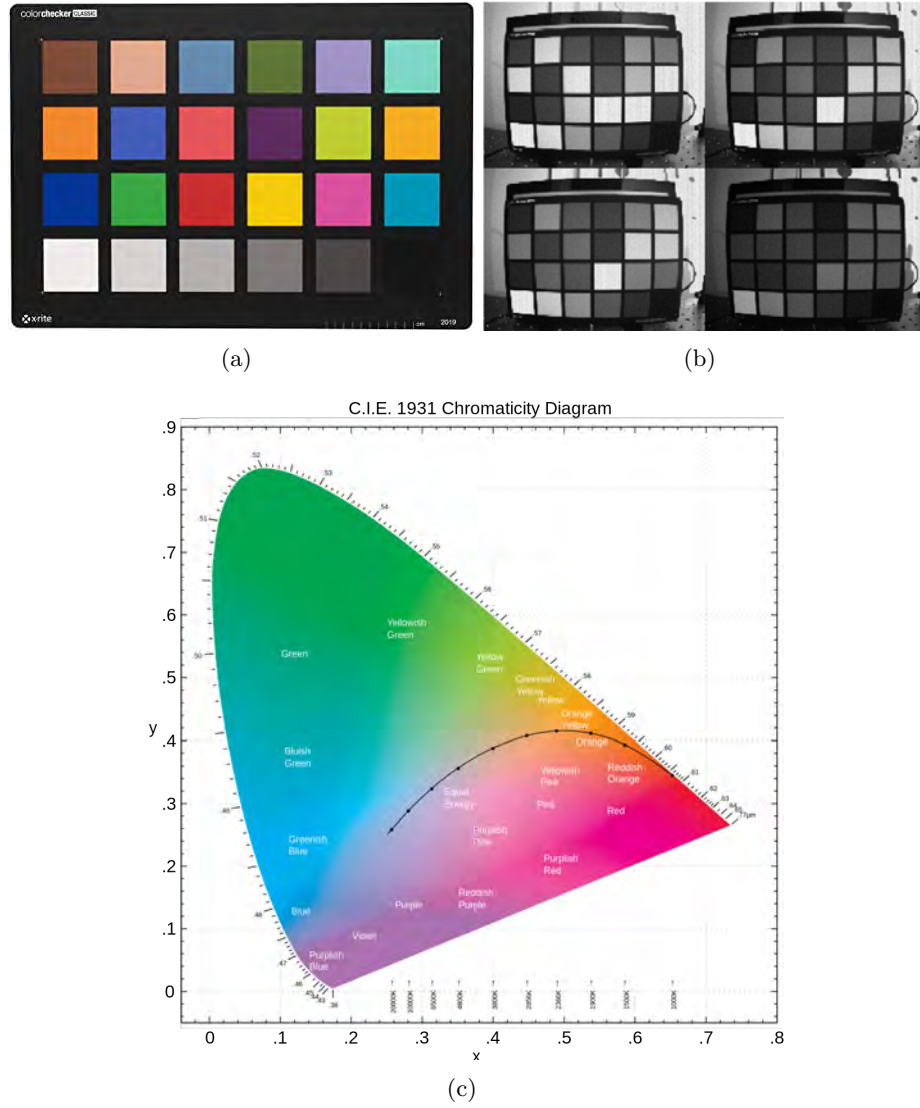


Figure 3.7: Chromaticity coordinate with the colour checkerboard. (a) The colour checkerboard from Macbeth. (b) The colour checkerboard through the APS readout of the EBC for each colour filter. (c) The CIE chromaticity diagram and the colour label for each colour patch, the different colour tones are shown and parametrised with the normalised coordinates x and y .

each colour channel was multiplied by a transformation matrix¹⁴ as shown in equation 3.5 to compute the tristimulus values (i.e. XYZ). These values were then normalised to produce the chromaticity of the light (x, y). These coordinates are compared with the actual colour coordinates from the chromaticity diagram to estimate the quality of colours better. The calculation of x and y was performed using Equation 3.6 and Equation 3.7.

¹⁴<https://www.image-engineering.de/library/technotes/958-how-to-convert-between-srgb-and-ciexyz>

$$\begin{bmatrix} 0.41 & 0.35 & 0.18 \\ 0.21 & 0.71 & 0.07 \\ 0.01 & 0.11 & 0.95 \end{bmatrix} * \begin{bmatrix} \sigma_R \\ \sigma_G \\ \sigma_B \end{bmatrix} \leftarrow \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \quad (3.5)$$

$$x = \frac{X}{X + Y + Z} \quad (3.6)$$

$$y = \frac{Y}{X + Y + Z} \quad (3.7)$$

where σ is the normalised value for each colour channel, X , Y and Z are the tristimulus values, and x and y are the final chromaticities coordinate. Figure 3.8(a) and Figure 3.8(b) show the chromaticity error for each coordinate and the chromaticity distance error respectively. The chromaticity error was computed using the relative absolute error Botchkarev [2019] which takes the positive difference between the expected coordinate and the computed one (Equation 3.8). The absolute error was performed for each coordinate to observe whether the error value is shifted toward the x or y-axis. The error distance between the expected and the computer chromaticity coordinate was performed as illustrated in equation 3.9 and shown in Figure 3.8(a) and 3.8(b).

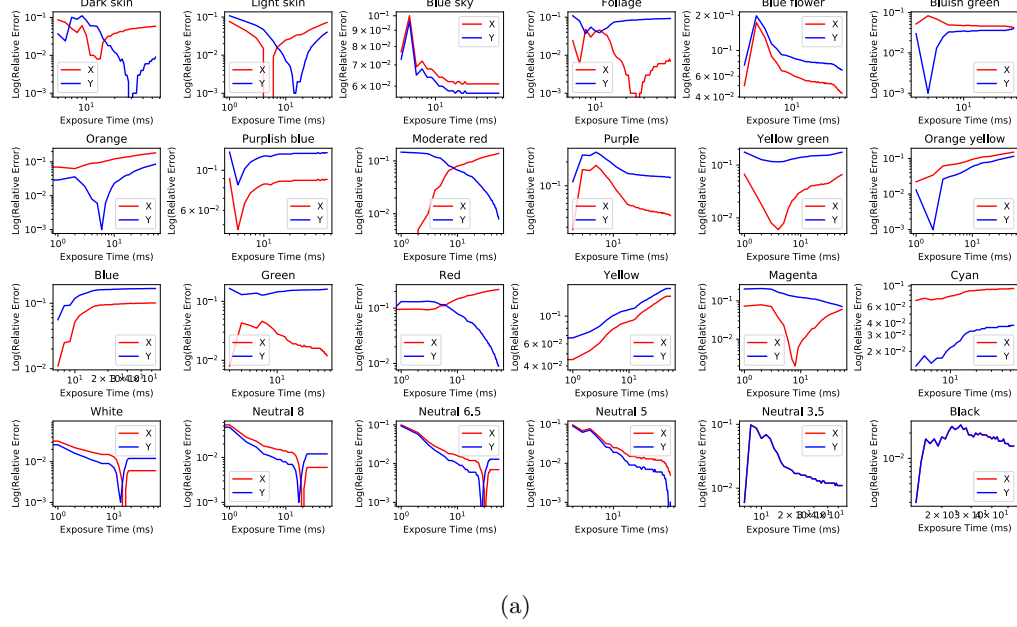
$$RAE = |V_A - V_T| \quad (3.8)$$

$$Distance = \sqrt{(X_A - X_T)^2 + (Y_A - Y_T)^2} \quad (3.9)$$

where V_A is the approximated value and V_T is the actual value. As shown in Figure 3.8(a), there was a high correlation between x and y coordinates in Neutral five and black colour. The error increased dramatically at medium-high exposure, and the error decreased when the exposure time was lower. Similarly, for the White, Neutral 8, Neutral 6.5 and Neutral 5. In this case, the error decreased at a specific exposure time, and it was high for all other values. Colour patches such as Orange, Orange-yellow, Bluish-green and Cyan had the same error patterns where it was lower for the y coordinate at medium-high exposure and high error for the x coordinate because orange and orange-yellow are relatively closer to each other on the chromaticity diagram, similarly to the cyan and bluish green. Blue sky and blue flower patches showed lower chromaticity error at high exposure time. Overall, yellow, white, blue, bluish-green, cyan, neutral 3.5 and black showed lower error at lower exposure time compared to other colour patches. As a result, these colours can be seen even under lower lighting conditions.

This experiment is independent from the DVS pixels and it does not take into account the event response and behaviour, therefore, it serves as a guide for calibrating the APS as well as the DVS pixels.

Chromaticity Coordinate Error



Distance between calculated chromaticity coordinate and ground truth

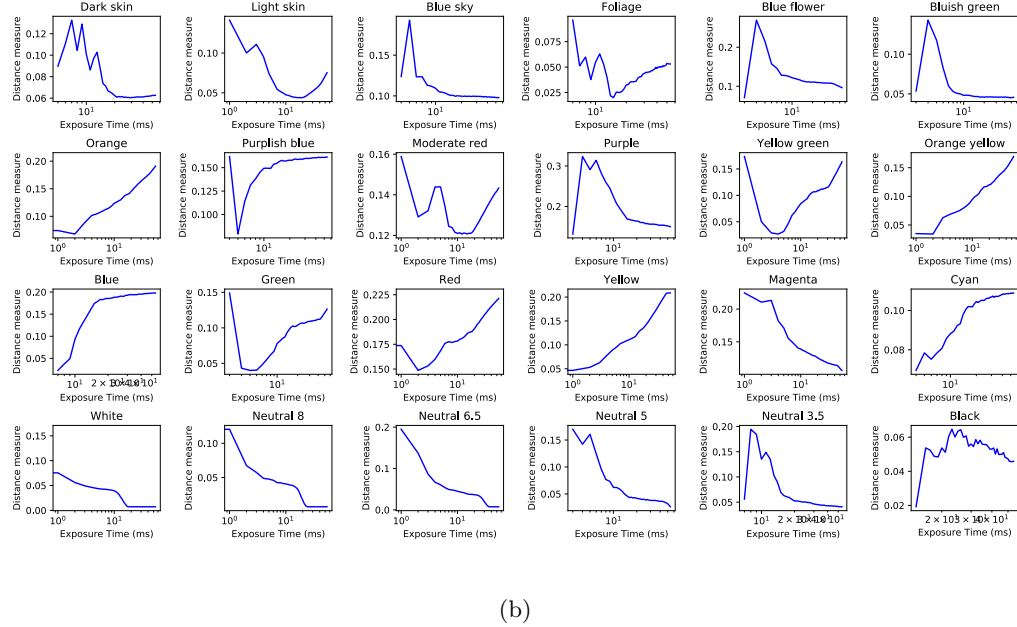


Figure 3.8: The results of the chromaticity error for each of the colour patches. (a) Chromaticity error for each colour patch on the macbeth colour checker board. (b) The distance between actual and measured chromaticity coordinate for each colour patch to measure the estimated colour error.

3.5.3 Spatial Uniformity

It is essential to evaluate the uniformity across the DVS pixels to gain a better understanding of its output response under different illumination conditions and different contrast thresholds. As mentioned in [Gallego et al., 2020], the EBC suffers from various problems such as noise and dynamic effects due to the inherent shot noise in photons, transistor circuit mismatch, circuit thermal noise, fixed pattern noise and the sensor non-idealities. These types of noise introduce uncertainty and non-uniformity in the event readout, typically in the order of hundreds of μs under normal lighting conditions, as shown in Figure 3.9. This is especially true for EBC where the process of quantising temporal contrast is complex and has not been thoroughly characterised and investigated. The DVS pixels reduce the mismatch generated by the ON and OFF comparators by referring the mismatch to the input, which is minimised by the gain from $\pm 20\text{mV}$ to as low as $\pm 2\text{mV}$. This is crucial as the mismatch introduces a random variation in the threshold along with the differencing amplifier, which is inevitable in the DVS sensor Lichtsteiner et al. [2006], Lichtsteiner et al.. To characterise the sensitivity and the non-uniformity of the sensor, we investigated the spatial distribution of the events readout across a hardware-enabled ROI.

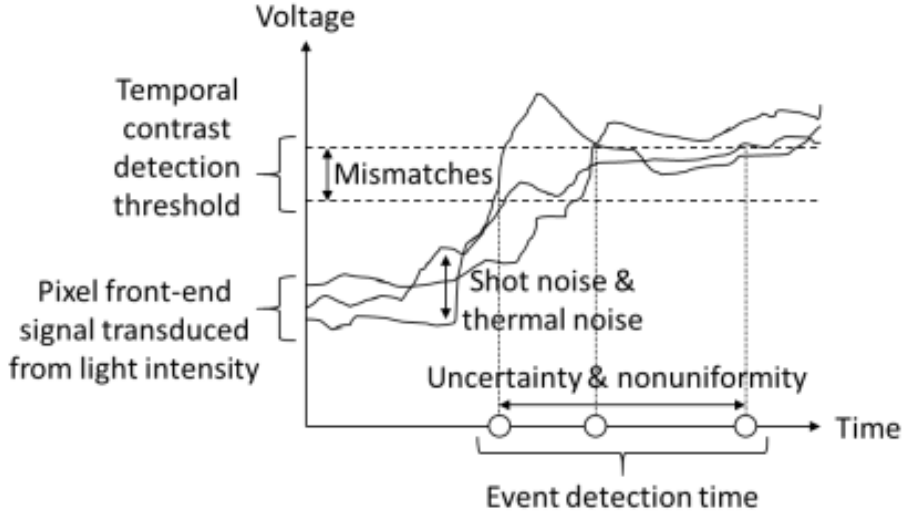


Figure 3.9: Uncertainty and non-uniformity in the event detection time [ini, 2021].

In this experiment we used the experimental setup described in Section 3.4.2. To quantify the uniformity of the DVS pixels, a colour transition pattern was presented to the CDAVIS in a sliding motion (i.e. right \leftrightarrow left) 30 times using nine unique stimuli. This was performed to ensure the events readout are consistent and the variation over time is precisely measured (e.g. average of events and error). The stimuli are composed of a combination of two primary colours. To minimise the effect of the arbiter and reduce the mismatch, a hardware-based ROI was selected to disable all the pixels outside of the ROI. The number of events for each condition was calculated per column for each polarity (e.g. ON events and OFF events) and all colour filters. It was observed that the differences between the events polarities change depending on three main

factors: (1) scene contrast, (2) sensor contrast threshold and (3) scene illumination. Sepia [Marcireau et al.] was used to change the sensor contrast threshold for the ON and OFF polarities.

Figure 3.10(a) and Figure 3.10(b) show the spatial uniformity response across different contrast thresholds for ON and OFF events respectively. We concluded that the spatial uniformity varies with the object/background contrast based on these results. For instance, the contrast is higher when the background colour is white for all primary colours, as shown in the bottom panels. The output response becomes more uniform across the ROI pixel array. In contrast, the output spatial response was less uniform and sparser across the columns when the colour transition was from blue to red, red to blue, blue to green and green to blue. It became less uniform at higher threshold values. Accordingly, the contrast ratio between the object and background can significantly change the output responses of the DVS pixels. With high contrast threshold (i.e. camera bias), the pixel array stops responding to changes in contrast in the scene. For that reason, very few events were triggered. It was clear that at a high threshold value, the OFF events were more uniform, and there the distribution was higher in terms of the number of events than the ON events. Overall, the spatial uniformity gradually decreases when the threshold increases and at a very high contrast thresholds (e.g. 100%), the events inside the ROI window no longer trigger events for both polarities.

Overall, this experiment shows the variance of the uniformity of the DVS response across the primary colours. This shows that the colour event output is not uniform in every case, that is mainly due to the contrast threshold selected. This confirms that the number of events for each colour filter is not only affected by the colour of the object and the background, but also the sensor threshold.

3.5.4 Pixel Temporal Noise

This Section analyses the temporal noise under different conditions regarding sensor threshold and scene illuminance. A light diffuser was used to ensure that the light was equally scattered over the patterns and measured the irradiance across the whole field of view. This step was performed to prevent having a bright spot on the patterns, introducing unnecessary noise events. The same colour transition patterns used in Section 3.5.3 were also used in this experiment. The goal was to measure the following: (1) total number of events for each colour filter as a function of contrast threshold, (2) the total number of events for each colour filter as a function of illuminance in Lux, (3) events rate (events/seconds) in terms of contrast threshold and (4) events rate (events/seconds) as a function of illuminance in Lux. The test was repeated 30 times to ensure that the data were consistent and without outliers.

In Section 3.5.1 we showed the space-time plot for different colour transition conditions. We found no correlation between the actual colour of the objects and the events triggered by each colour filter. That is because the contrast ratio between the object and background influences the event output and not the object's colour. For

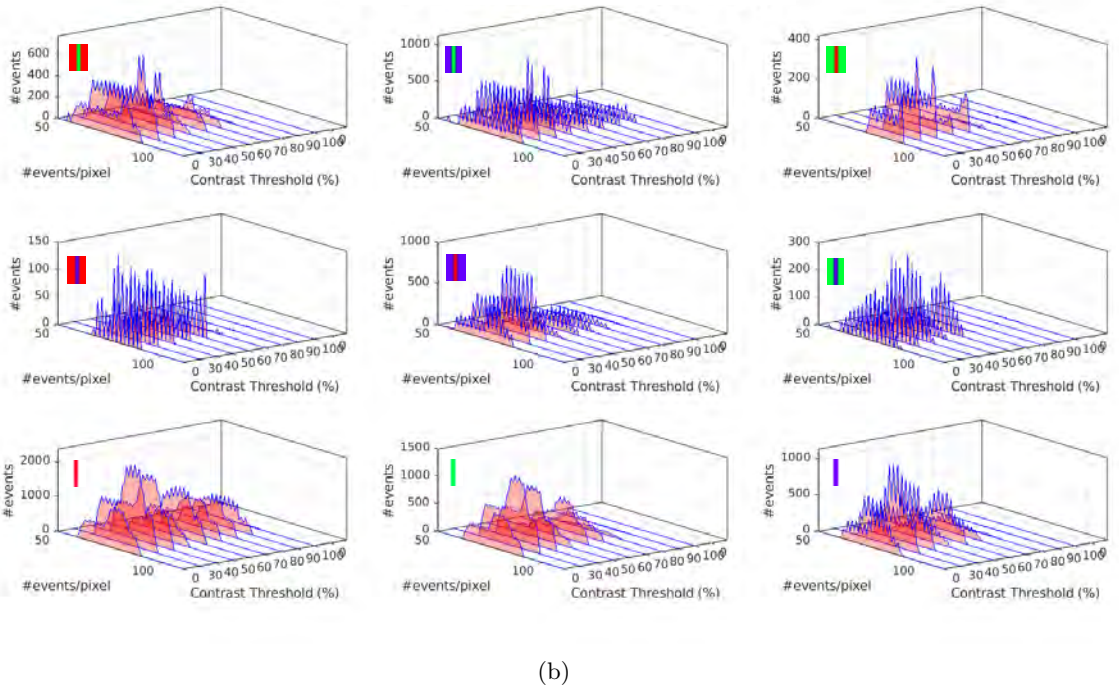
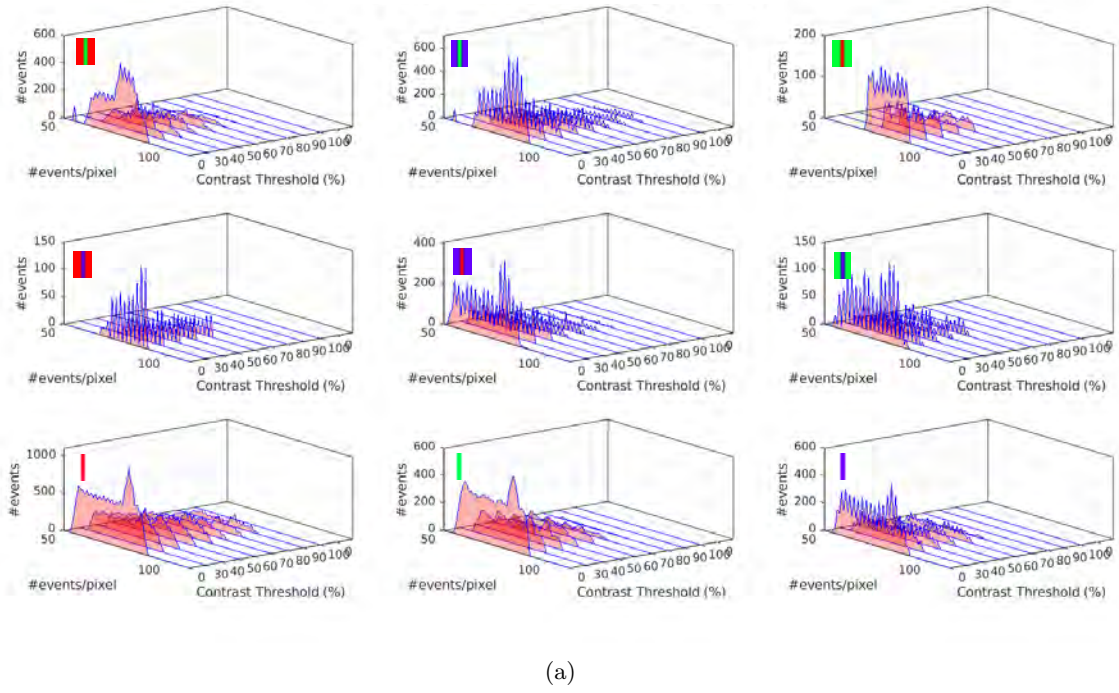


Figure 3.10: Pixel uniformity responses. (a) The uniformity of response for the ON events using all the combinations of colour transition for primary colours. (b) The uniformity of response for the OFF events using all the combinations of colour transition for primary colours.

that reason, we aim to analyse the same nine colour transitions through a wide range of contrast threshold and scene illumination and investigate the change in the behaviour

of the events. Since there is no universal approach to characterising colour events, we followed the conventional way of characterising the standard DVS by studying the total number of events and the event rate (events/s) for each colour filter.

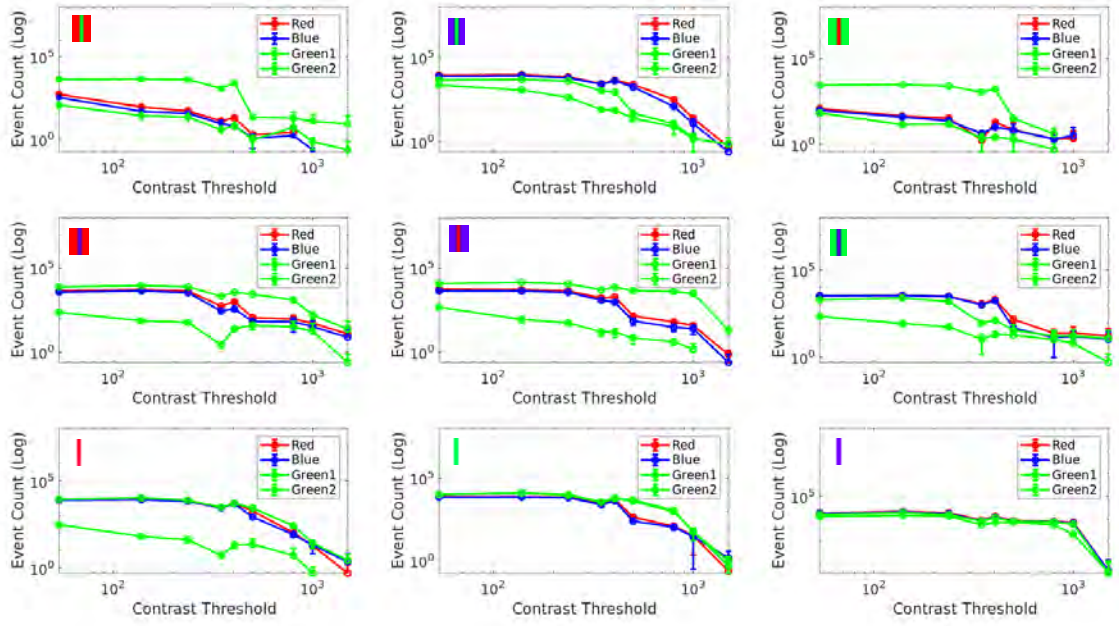
In Figure 3.11(a) and Figure 3.11(b), we used the event counts per colour filter as a function of contrast threshold and illumination as the primary characterisation measure for all nine colour transition patterns. Based on the event counts measured, we found that the event count decreases for all colour filters at higher thresholds, showing few events triggered by the sensors. Based on the change in the events count with different thresholds, it is difficult to identify the object’s actual colour based only on the event counts. Similarly, when the transition from one colour to another is in the opposite direction. For instance, the transition from red to green and then green to red results in the same event counts.

Moreover, it was evident that the distribution of red and blue events was similar throughout the threshold range in each of the nine experiments. In addition, when transitioning from white to blue, all colour filters trigger events equally, leading to the exact event count. Finally, at a higher threshold, the error increased, indicating non-idealities in the pixels array when the threshold is higher. When characterising the event count as a function of the illuminance, we found that the events count decreases inverse proportionally to the light intensity for all colours transition conditions.

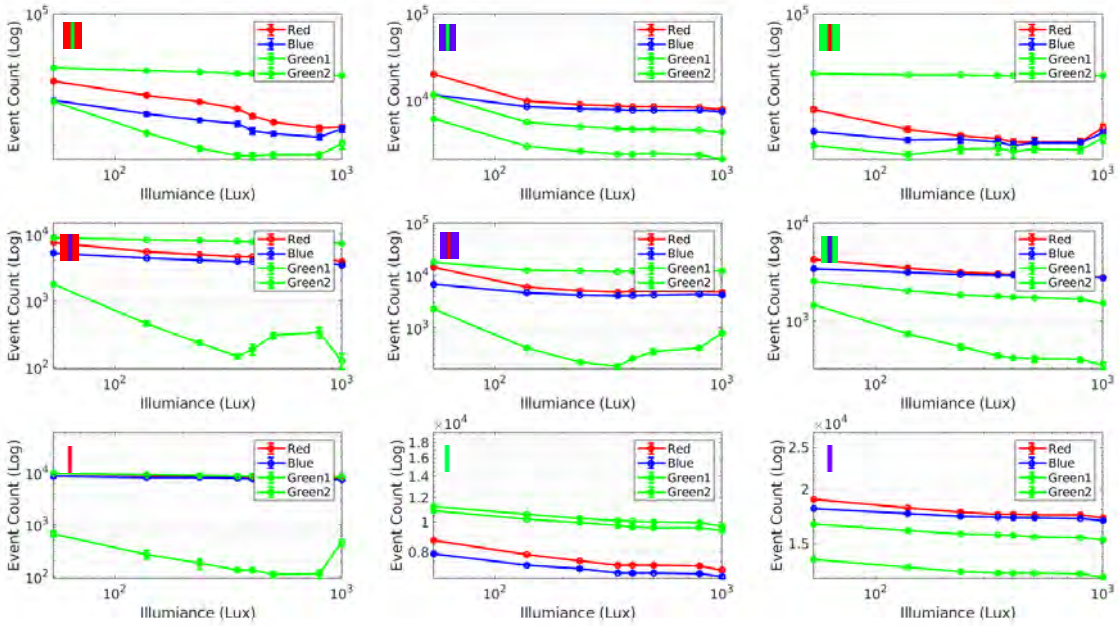
In Figure 3.12(a) and Figure 3.12(b), we considered the events rate as the primary measure for each polarity, and we investigated it in terms of the sensor threshold and scenes illuminance and using the nine colour transition patterns. We observed that the event rate decreases dramatically for both polarities at a higher threshold based on the event rate. At a lower threshold, the events rate increased but with a high error indicating the presence of noise in the DVS pixels at a lower threshold. A lower sensor threshold generally triggers more events with pixel noise. In addition, when transitioning from white to blue at a higher threshold, we observed no OFF polarity events and an increase in the ON events. When transitioning from red to green, blue to green and blue to red, we observed that the ON event rate was higher than the OFF event but both ON and OFF event rates became similar at a much higher threshold.

On the other hand, when we experimented with different illuminance, we observed that the ON events rate increased at a very high illuminance while the OFF events rate kept decreasing, except for colour transitions from red to green and blue to green, which resulted in a low event rate at the maximum illuminance. The pattern was the same for both polarities. We also observed that the error increased when the light intensity increased. However, overall the patterns were the same when the background was white, which was reasonable because the white background appears brighter through all colour filters, contrasting the rigid line and the background enough to trigger spatially rich events.

By considering the events count and average event rate as characterisation measures, it was possible to investigate the number of events produced by the DVS pixels under different contrast and lighting conditions as well as the events rate under specific



(a)

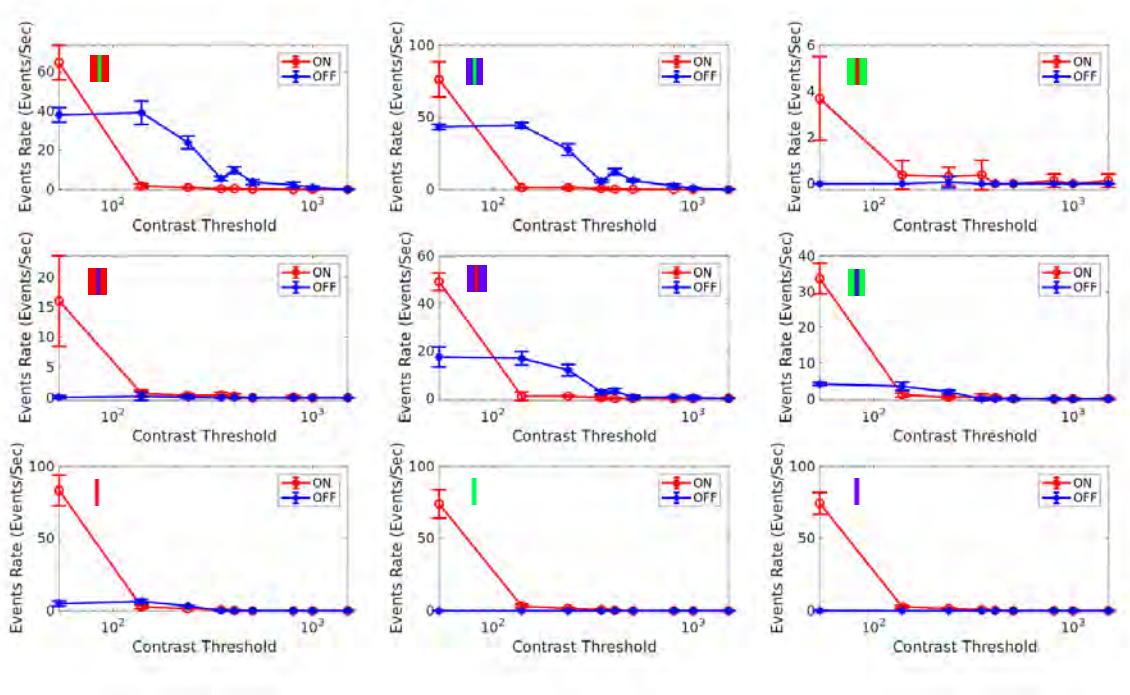


(b)

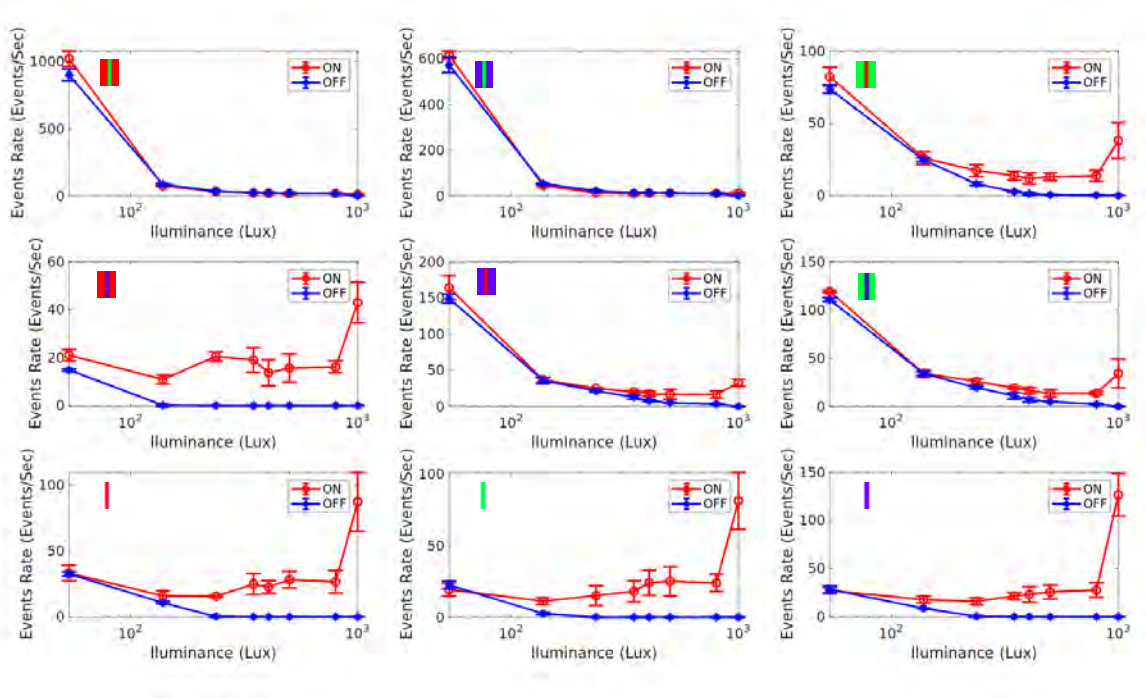
Figure 3.11: Pixel temporal noise results. (a) show the total number of events across a wide range of contrast thresholds using primary colours transitions. (b) shows the total number of events across a wide range of scene illumination using primary colours transitions.

conditions. This can help record colour events data efficiently and calibrate the colour sensor for specific scenes. While, these two measures do not provide information about

the actual colour of an object. They provide more information about the difference between two or more colours (i.e. contrast).



(a)



(b)

Figure 3.12: Temporal noise characterisation. (a) shows the event rate to measure and quantify it in terms of sensor contrast threshold (ON and OFF threshold). (b) shows the event rate to measure and quantify it in terms of scene illumination in Lux.

3.5.5 Chromatic and Achromatic Contrast

By definition, achromatic colours have lightness but no hue or saturation. However, chromatic colours have hue, and both can be created by mixing complementary colours. The elements of hue, lightness and saturation found in chromatic colours are referred to as the three attributes of colour, and specific colours can be represented by specifying the values for each of these attributes.

There have been notable works where the role of colours was investigated in human visual perception. For instance, Breuil et al. [2019] showed that providing colour images can improve edge classification compared to grayscale images and suggested that colour information facilitates the identification of material properties, transparency, shadows and the perception of shape-from-shading. Hansen and Gegenfurtner [2017] investigated the benefits of chromatic edge contrast on object-contour perception, and they show that chromatic information is essential for better representing object contour and detecting objects quickly and easily. We investigated two things as motivation: (1) grayscale patterns with different contrast ratios and (2) gradual colour mixing. The edges were rigid to enable a rapid change between two surfaces with distinct contrast in both cases. We investigated grayscale and colour patterns to observe the effect of grayscale edges against edges with coloured and characterise the sensor sensitivity using a wide range of contrast ratios for both cases.

In this experiment, we used the same pan and tilt platform described in Section 3.4.2 and used a constant illuminance using the non-flickering LED. Thus the only variable is the colour of the background. We considered the total number of events as the primary measure to calculate the amount of activity in each colour filter for each condition—the patterns comprised of printed colours on a paper. Here, we investigated the DVS pixels behaviour using grayscale patterns where the only variable parameter used was the colour of the patterns (i.e. contrast).

The patterns consist of two blocks, each with a different grayscale level, creating a rigid line between two surfaces. This allows the pixel to have an instant shift between the surfaces. For each test, the contrast ratio between both surfaces was varied by either increasing or decreasing the grey-level value of each block. We considered the total number of events as the primary measure to characterise the camera sensitivity for high and low contrast. A 104x7 hardware-enabled ROI was selected to activate the pixels within a small region and prevent interferences by the other pixels in the imager. For that reason, we only receive events from the pixels inside the ROI. For recording, the same pan and tilt platform described in Section 3.4.2 was used to move the camera in a linear motion across the x-axis over the patterns as shown in Figure 3.13. The duration for each recording was 2 seconds, and it was repeated 30 times to measure the mean and standard deviation of the events. A wait state is activated between each recording for 2 seconds to let the pixels recover from possible refractory period effects. Using both DAVIS and CDAVIS we considered the total number of the events for both polarities and all colour filters (in case of CDAVIS).

Figure 3.14(a) and Figure 3.14(b) show the difference in the events readout for the

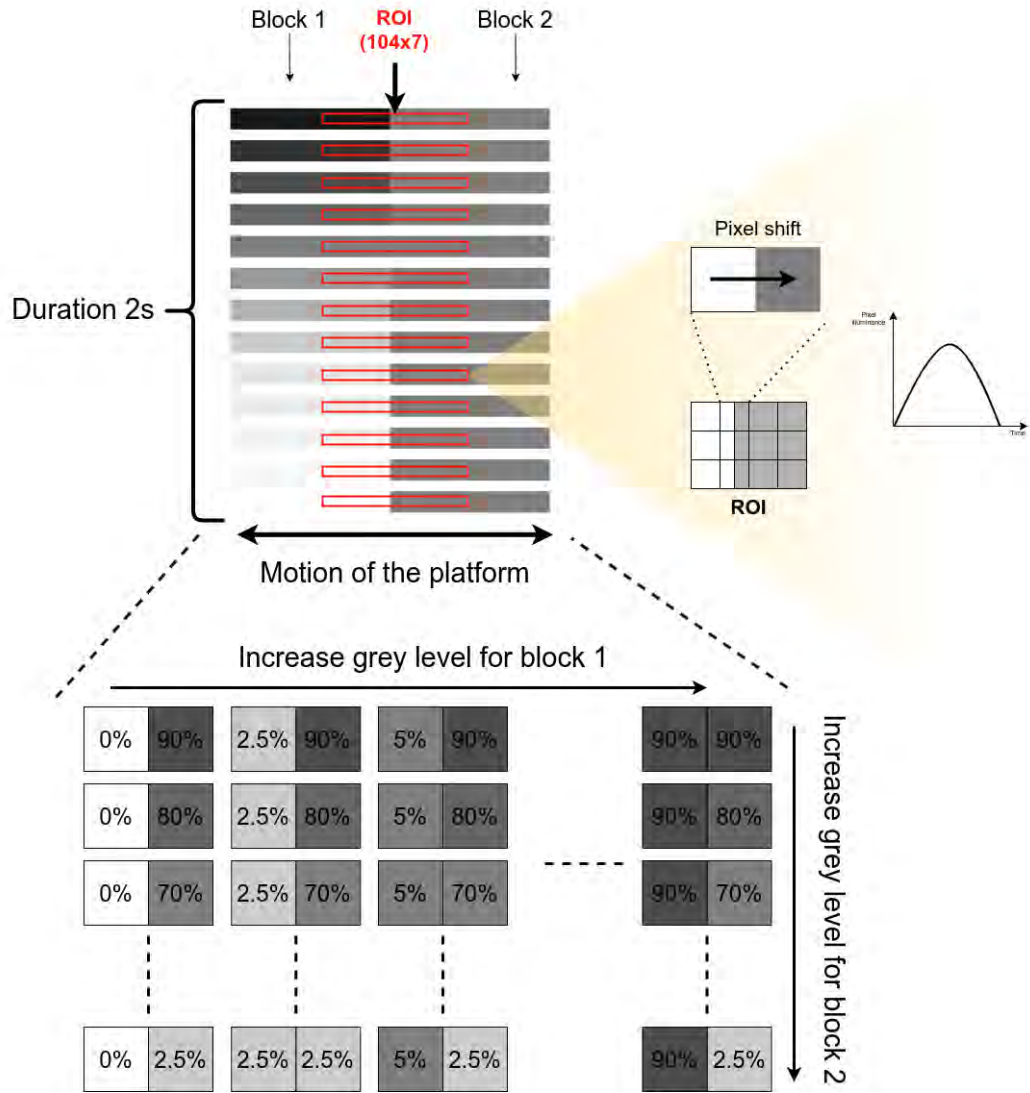


Figure 3.13: The greyscale patterns are used in the chromatic characterisation. It consists of two blocks with different greyscale values to allow the pixel to shift from one surface to another.

DAVIS and CDAVIS respectively. It was evident that each camera exhibited different behaviour on the same patterns. For the DAVIS, it appeared that when pixels shifted from white (block 1) to block two that has a grey-level value from 90% to 0%, the total number of events decreased until there was no visible contrast which led to no more events triggered in the ROI. The same case behaviour occurs when setting the grey-level to 2.5%, 5%, 7.5% and 10% for block 1. In this case, the contrast becomes very similar to a grey-level between 0% and 10%. Furthermore, the total number of events increased when the grey-level in block one was 20%.

On the other hand, when the grey level for block 2 exceeds 20%, the total number of events increases proportionally to the greys level value of block one. Compared with the DAVIS and under the same conditions such as scene illuminance, lens focusing, and camera bias, the CDAVIS generated approximately 2X more events across the ROI considering that all events from all colour filters were counted. The CDAVIS

appeared to be more sensitive to grey-level edges. The total number of events exhibited a parabolic shape indicating the presence of contrast within the selected ROI. However, the main challenge is to find whether the events were triggered from the actual edge between the surfaces or just noise events generated by the colour filters (e.g. due to circuit mismatch). The standard deviation was higher at high contrast ratio and lower at low contrast ratio, indicating a high variance in the pixel response at high contrast ratio, which might be caused by the surface edge and its effect on the refractory period.

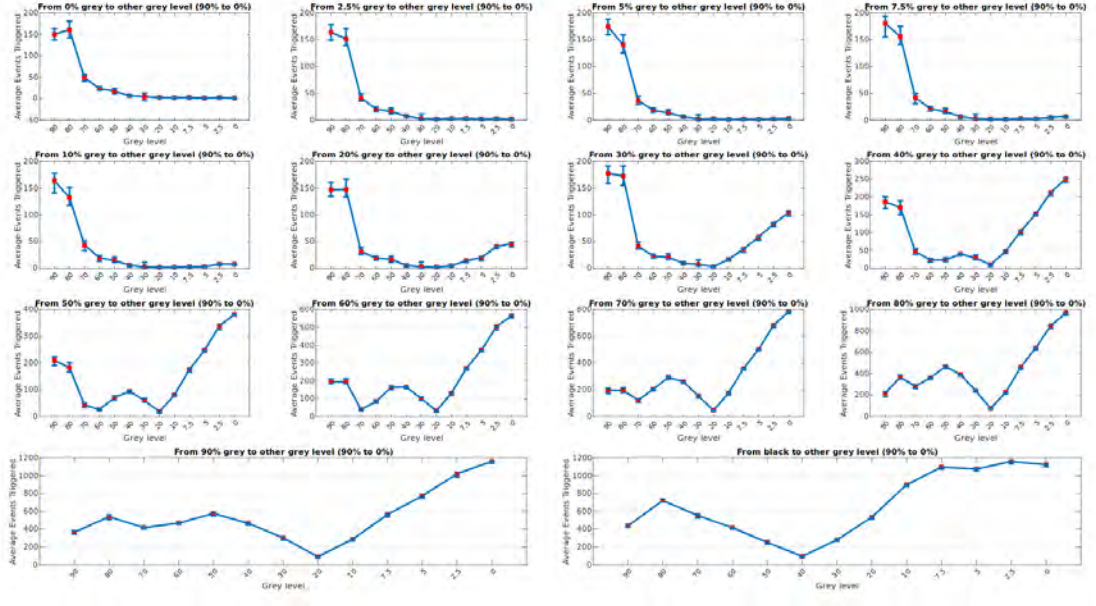
In a different experiment, we used achromatic patterns to characterise the response of the DVS for each colour filter across a wide range of colour variations. The aim is to observe the influence of background colour on the event's output. A green circle on the top of a red background was chosen as an input stimulus, and another colour was gradually added to the red background, such as blue, green, and white, to change the background colour. We observed how the background colour changed the contrast ratio and how the events generated by each colour filter changed according to different conditions. As shown in Figure 3.15(a), it was apparent that the APS output became brighter when observing the red background through the red filter. In this case, the green circle becomes darker. In contrast, the scene looks much darker through the blue filter, as shown in Figure 3.3. The addition of green colour to the red background add more contrast, making the appearance between the pattern and the object clearer. Similarly, when adding white to the red background.

In Figure 3.15(b), we showed the results using the total number of events for each condition. We observed that by adding blue colour to the red background, the contrast increases for every colour filter because the final magenta colour was reflective of red and blue. Also, we observed an increase in the total number of events for the green1 and blue filters compared to other colour filters, which had no significant change. This indicates that adding a blue to red colour produced more events in the green1 and blue filters. Due to the dark appearance of the background through the blue filter, there were more events through the blue filter, which indicates the presence of an excessive amount of noise events for short-wavelength (i.e. lower sensitivity). While gradually adding green to the red background, the contrast increased in each colour filter, and the background in both green filters became brighter. From the perspective of the DVS, we found that the events in each colour filter increased proportionally with the amount of green colour added to the background. In addition, while adding white to the red background, we observed that the number of events increased for each colour filter simultaneously by adding more white to the red background. This experiment indicates that the achromatic edges can affect the output of the event, and by adding different colours to the background, the contrast ratio will change. The behaviour of the DVS will vary accordingly.

3.5.6 Colour Events Classification with KNN

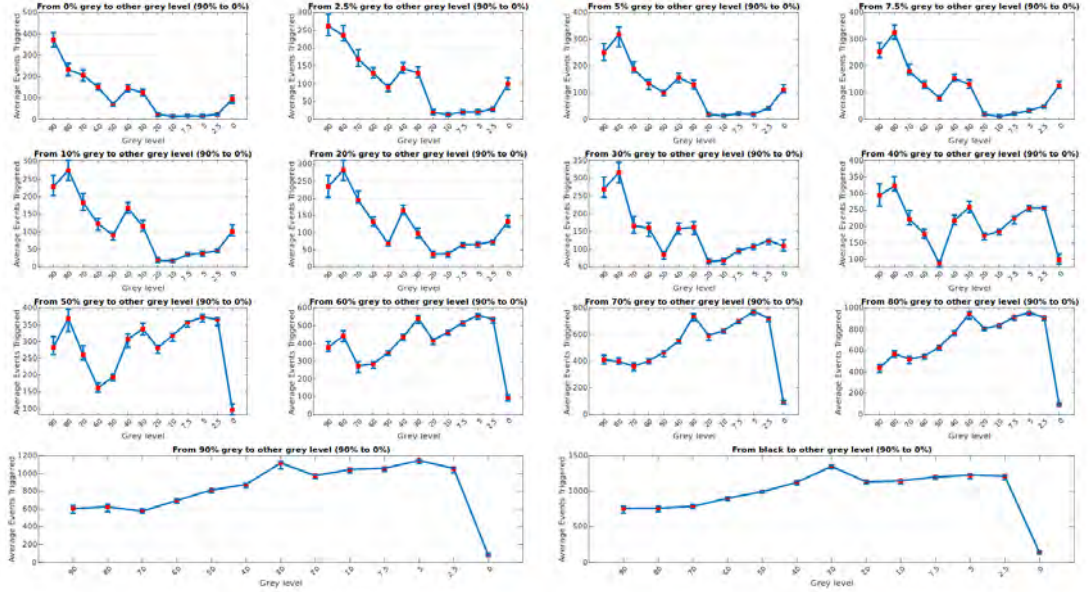
A colour event dataset was created using the CDAVIS sensor. The dataset was made to investigate the importance of colours events information in distinguishing between

With DAVIS346



(a)

With Colour DAVIS346



(b)

Figure 3.14: Total number of events versus contrast comprised of black and white edges with different contrast ratios. (a) show the events triggered by the DAVIS346. (b) events triggered by the CDAVIS.

different patterns in classification tasks. It consists of 50 events data files for each class where there is a single object per recording. In this case, the only variable is the colour of the background and the colour of the object. The dataset was divided into three different categories: (1) objects with different shapes and with the same colour, (2) objects with different shapes and with different colours, and (3) objects with the same

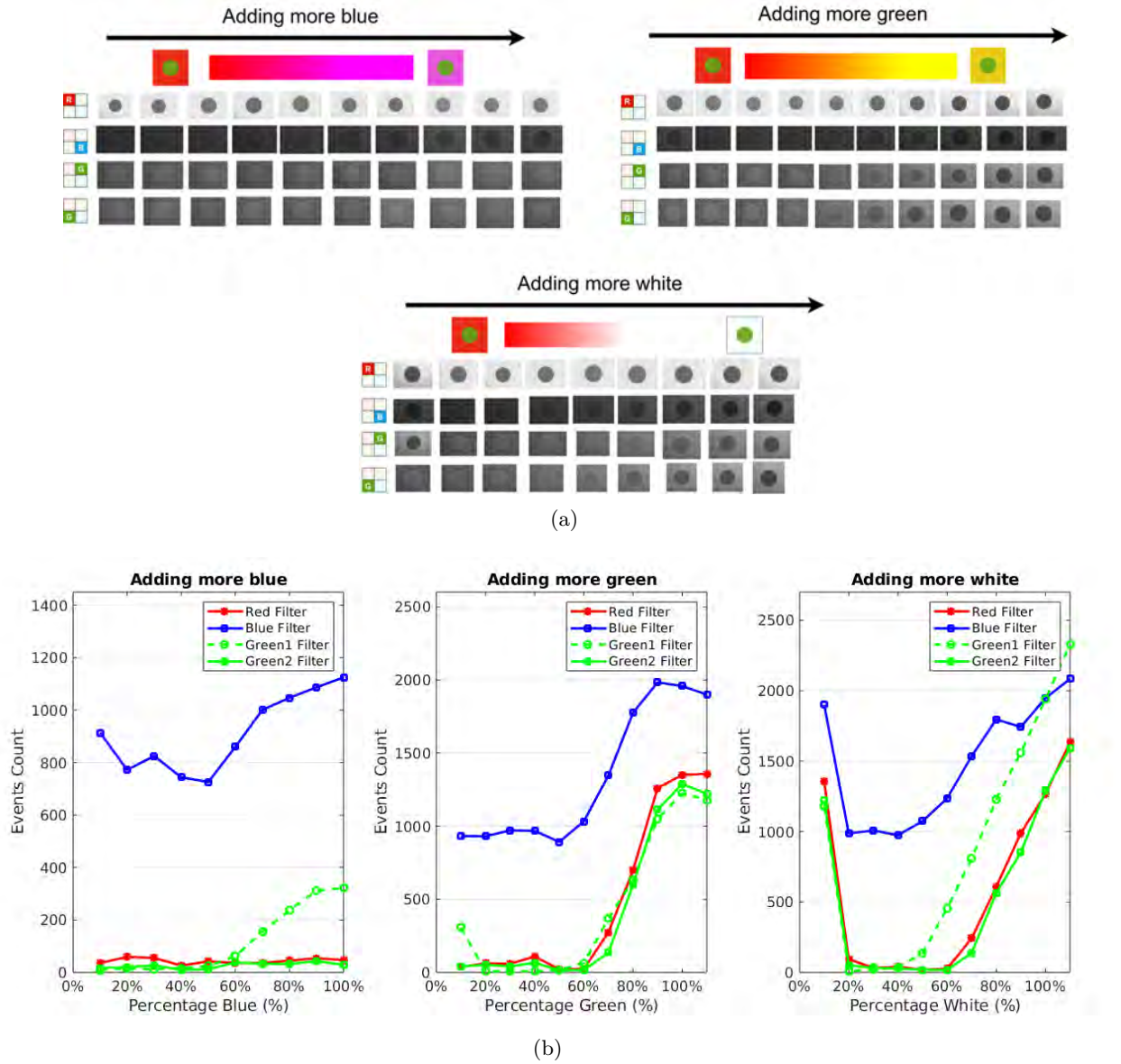


Figure 3.15: Colours addition experiment. (a) The APS output for each pattern showing a gradual addition to blue, green and white to the red background while keeping the green circle colour the same. (b) The total number of events through each colour filter for each colour addition condition.

shape and with different colours. Simple shapes were chosen, such as circles, lines, stars and triangles with primary colours. The dataset contains the same number of testing and training items, and the distribution of images between them is preserved. We found that the characteristics related to each sequence vary slightly because the recordings were performed using a physical device in a repetitive motion resulting in timing and accuracy in the hardware and acquisition process. Consequently, non-idealities and non-uniformity in terms of noise also existed in the dataset, adding another dimension of complexity, such as real-world scenarios.

Since the object does not occupy the entire field of view, only the events that belong to the object were selected, reducing the high dimensionality of the data and removing

all the excessive, unnecessary noise events generated by the background, which have less contribution compared it to the events from the real object. This was performed by removing the pixels on the boundary in post-processing. The resolution of the data becomes 100x100 pixels centred on the object. The average length of the recording is 2 seconds. We used the pan and tilt platform described in Section 3.4.2 to swipe through the patterns in both directions. The average number of events varies between each patterns due to the difference in size and the colour as shown in Table 3.2, Table 3.4 and Table 3.3. For example, for "different objects same colour", there were more OFF events in the train and test set. Given that the number of events per colour filter is different, even for objects with the same shape, that means the number of events triggered by the DVS is not only influenced by the size and shape of the object but also by the colour.

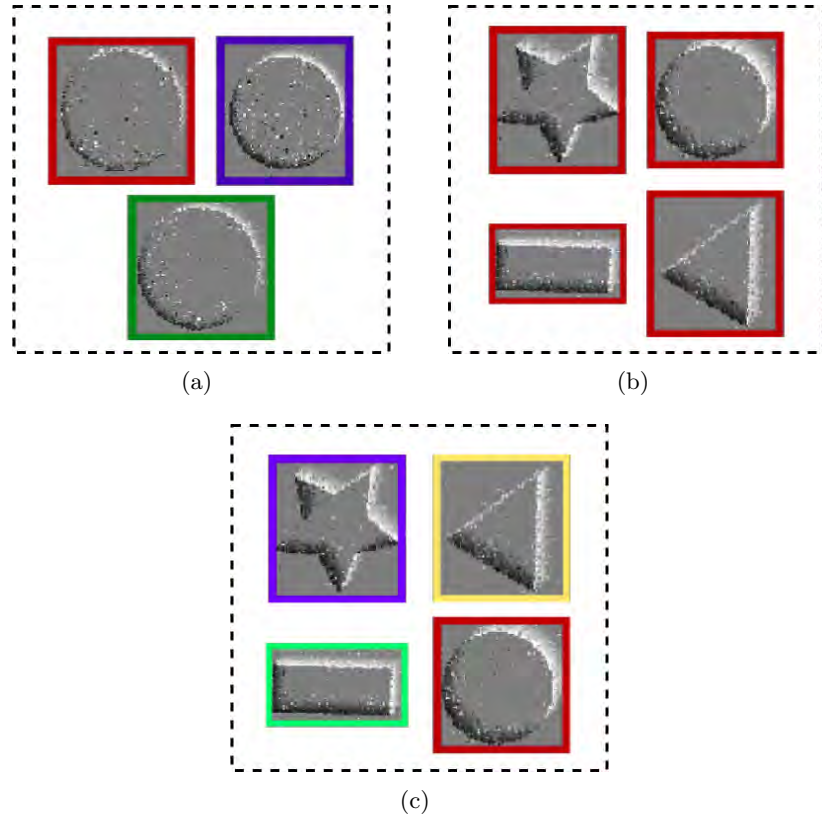


Figure 3.16: Conditions selected for classification where colours and shapes are essential features to identify the differences between different classes. The edges were coloured to indicate the colour of the object. (a) Same object with different colours. (b) Different objects with the same colour. (c) Different objects with different colours.

To explore the internal properties of the events, a simple classifier namely KNN was constructed to classify each category. KNN classifies an unknown example with the most common class among K closest examples. There is no inherent training with a KNN classifier but rather a calculation step that generates the labelled data. The testing process performs the actual nearest neighbour clustering. The calculation allows for the varying number of neighbours (e.g. the k value) during the testing. This also

inherently allows for multiple and parallel testing, allowing for the broad sweeps of k-values performed. As the entire process is deterministic, there is no need to run multiple trials. The order of the data cross validation also plays no role.

The Euclidean distance was used as the primary distance measure for the algorithm as shown in the Equation 3.10.

$$d(x, y) = \sum_{i=1}^n |x_i - y_i| \quad (3.10)$$

Different K values were tested, and we found $K = \sqrt{x}$ where x is the total number of the training set to be the optimal solution as it works best as an odd number [Hassanat et al., 2014], and the results stabilised at this value. This had the following advantages in increasing the speed of the algorithm by avoiding the even classifiers, to avoid the chance of two different classes having the same number of votes and the pilot experiments having the even K's show no significant change of the results. Overall, this can avoid the problem of ties that occur when two or more points are equidistant from an unclassified observation, thereby making it difficult to choose which neighbours are included. The same statistics were then found for each testing sequence, and the closest k neighbours were extracted. This approach takes the most direct means of constructing a classifier by examining the patterns internal properties. KNN was implemented and tested across all dataset categories for nine different statistical properties calculated across each training and testing sample. For each pattern, the following statistical properties were used:

1. The total number of events in the pattern
2. The duration of the motion in milliseconds
3. The number of ON events in the pattern
4. The number of OFF events in the pattern
5. The ratio of ON events to OFF events in the pattern
6. The mean x address calculated across all the events in the pattern
7. The mean y address calculated across all the events in the pattern
8. The standard deviation of the x addresses in the pattern
9. The standard deviation of the y addresses in the pattern

The same parameters were calculated for each training sequence and then used as the input dataset for the KNN classifier. We showed the accuracy over the entire data and each colour filter to better understand the contribution of colours in classification. It was immediately apparent that given the selected K value as $K = \sqrt{(x)}$ where x is the number of samples, the classifier performs well on all the statistical properties

except for the event’s duration, which was expected as all recording duration were the same. For example, for the ”same object different colours” category, the classifier achieved an accuracy of 98% and 100% across the standard deviation of X and Y addresses, respectively, as shown in Figure 3.17. For the other two categories, such as in Figure 3.18 and Figure 3.19 the classifier recognition results achieved an accuracy of 73% across all the statistical properties except for event duration, that is because in both cases, the patterns used were the same, but with different colours. Colours in these cases showed a minimal effect as the shape of the object was a dominant feature.

It was also apparent that the classifier based on the sequence duration yielded statistically insignificant classification results even when splitting the colour filter into four event streams. However, it produces an overall recognition accuracy well above the chance level. Overall, we concluded that for the ”same object different colours” category, colours significantly contribute to making the patterns separable, although they had the same shape and size. This was also possible without splitting the events into four colour channels. The colour becomes negligible when different objects have different shapes, such as in ”different patterns different colours” and ”different patterns same colours”.

These results are insightful, as they showed that colour events become an essential source of information when the object’s appearance is not relevant. Hence, to take advantage of the colour filters, one should record data in a colour-rich environment or where a rapid colour change needs to be detected. Future work should address those in more details.

In appendix A we presented a more detailed investigation using DAVIS sensor using objects with different shapes and different colours using the events statistical properties as input to an ELM classifier as well as spatio-temporal features using FEAST network. This study compares the same network architecture on both cameras to provide a better understanding of the differences between monochromatic and colour events.

Table 3.2: Statistical summary of the ”same object different colours” scenario. Results are shown separately for the training set of 50 recordings per class and the testing set comprising of another 50 recordings.

Statistics	Training Set		Testing Set	
	Mean	Std	Mean	Std
Duration of Recordings (s)	2.06s	1.30s	2.04s	1.32s
Number of Events (*1e5)	3.64	3.15	3.60	3.15
Number of ON Event (*1e4)	0.23	0.53	2.32	5.23
Number of OFF Event (*1e5)	3.36	2.67	3.36	2.67
X Address (px)	186.74	2.94	186.74	2.94
Y Address (px)	133.87	0.97	133.87	0.97

Table 3.3: Statistical summary of the "different objects same colour" scenario. Results are shown separately for the training set of 50 recordings per class and the testing set comprising another 50 recordings.

Statistics	Training Set		Testing Set	
	Mean	Std	Mean	Std
Duration of Recordings (s)	2.96	0.28	2.97	0.87
Number of Events (*1e3)	3.88	0.67	3.88	0.68
Number of ON Event (*1e3)	2.07	0.34	2.07	0.34
Number of OFF Event (*1e3)	1.80	0.34	1.80	0.35
X Address (px)	179.69	106.60	179.69	106.60
Y Address (px)	124.54	78.31	124.54	78.31

Table 3.4: Statistical summary of the "different objects different classes" scenario. Results are shown separately for the training set of 50 recordings per class and the testing set comprising another 50 recordings.

Statistics	Training Set		Testing Set	
	Mean	Std	Mean	Std
Duration of Recordings (s)	2.96	0.45	2.96	0.45
Number of Events (*1e3)	3.82	1.90	3.82	1.90
Number of ON Event (*1e3)	2.05	1.01	2.05	1.05
Number of OFF Event (*1e3)	1.77	0.98	1.77	0.98
X Address (px)	180.90	106.61	180.90	106.61
Y Address (px)	75.96	78.84	122.55	78.84

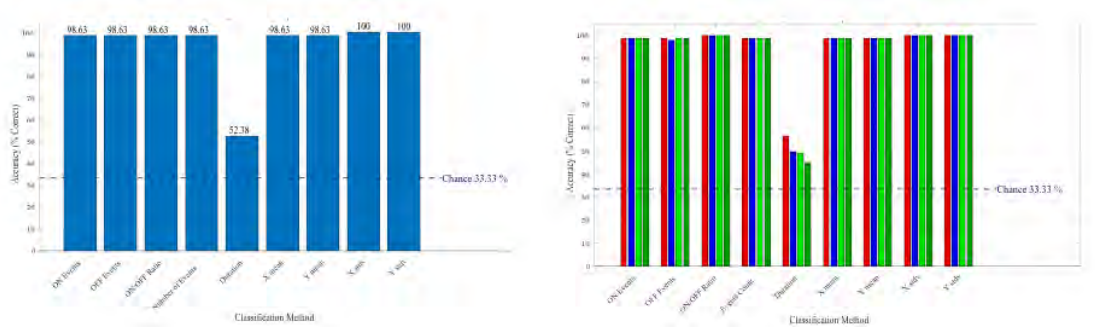


Figure 3.17: Classification Results for the KNN classifier on statistical properties for scenes where there are objects with the same shape but with different colours. **Left:** Accuracy on the stream of the entire event. **Right:** Accuracy for each colour filter.

3.6 Discussion and Future Work

The CDAVIS is the only commercially available working prototype of a single event-based sensor that can filter light from visible wavelengths using colour filter arrays (i.e. Bayer filters) combined into one photosensor. This sensor can only detect colour variation on the DVS part and absolute illuminance on the APS part. Thus, events from the DVS pixels do not provide information about the colour in the scene. To the best of our knowledge, the only way to achieve such a task is by reconstructing the absolute luminance and computing the RGB intensity for each pixel mathematically [Scheerlinck et al., 2018]. However, it requires additional operations, which increases

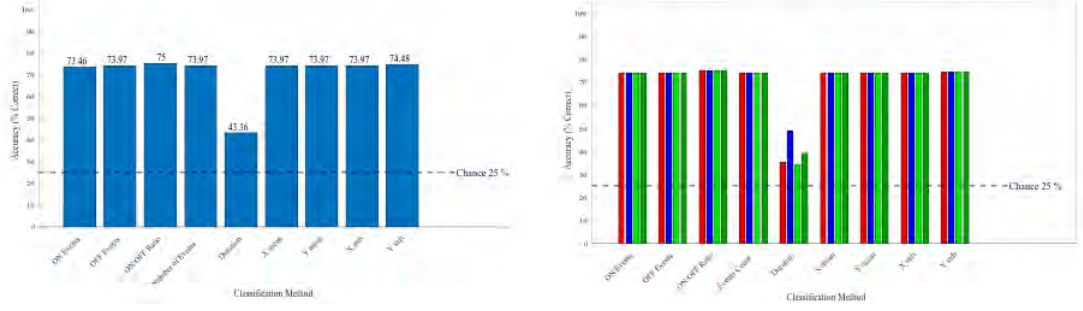


Figure 3.18: Classification Results for the KNN classifier on statistical properties for scenes where there are objects with different shapes but with the same colour. **Left:** Accuracy on the stream of the entire event. **Right:** Accuracy for each colour filter.

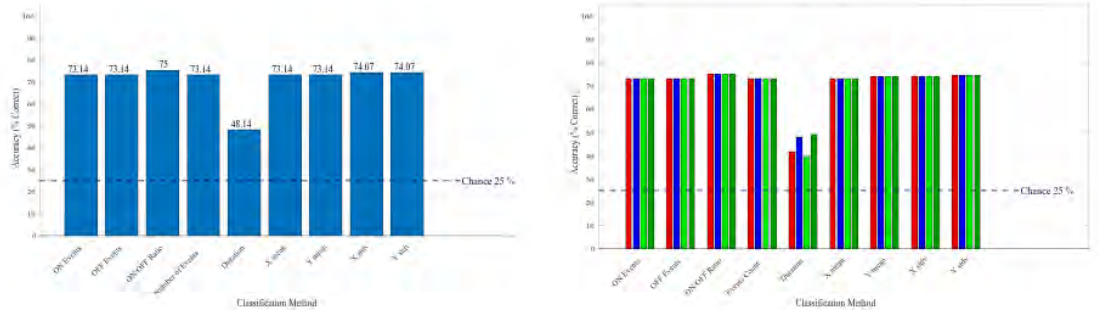


Figure 3.19: Classification Results for the KNN classifier on statistical properties for "objects with different shapes but with different colours" condition. **Left:** Accuracy on the stream of the entire event. **Right:** Accuracy for each colour filter.

the computational cost, making the whole mechanism synchronous due to the colour demosaicing operation. In this work, colour reconstruction is out of our scope as we purely investigated the DVS part. That is because the colour events from the DVS pixels are not fully understood and explored. Considering the characterisation study conducted in this chapter, we were able to identify the key advantages of the colour pixels in various domains as listed below:

1. Eliminate spatial redundancy and noise events. This can be achieved by actively selecting the colour filter with high SNR and removing events from filters will fewer activities.
2. Detecting rapid transitions between colours.
3. Detect transitions between different wavelengths such as radiation in the near-infrared band and the visible spectrum.
4. Distinguishing shadows from real objects in situations where the DVS does not see borders between two objects due to the same reflectance.
5. Imaging of neural activity [Moeys et al., 2018] by reconstructing images with High Dynamic Range (HDR) from the DVS colour pixels eliminating the need of expensive CMOS sensors.

Although colour events can benefit some applications, it suffers from various drawbacks, such as not having a universal approach to characterise events based on colour patterns as the majority of the EBC are by defaults monochromatic (i.e. greyscale). Thus, the same characterisation techniques and the exact measurements used might not be directly applicable to colour sensors. In addition, the current colour sensors are only characterised in terms of their internal circuit voltage and current properties and not the internal properties of the event.

This makes the processing and handling of colour spikes unclear and challenging. The only colour event dataset has been published by Scheerlinck et al. [2019] to give the researcher access to new types of datasets and motivate them to work in this direction. However, no further follow up work can be found since 2018, and the dataset is not application focused as it was recorded in various scenarios and scenes. Thus, future efforts must be dedicated to producing colour events datasets specific to particular applications.

There has been more focus on events to frame reconstructions algorithms as a way to process the incoming colour events from the CDAVIS and produce frames out of the events data. For that reason, the processing pipeline becomes fully synchronous once the demosaicking is included in the process. An emphasis on frames reconstruction shifts away from developing a native even-based end-to-end system that takes advantage of the camera capabilities (e.g. low power, low data rate, high pixel bandwidth, etc.) with the colour events transition advantages.

This has limited the progress of colour event data characterisation and processing techniques. It is advantageous to consider the difference between the monochromatic DAVIS and the CDAVIS as each requires domain and application knowledge, for example, using the CDAVIS in cases where colours are an essential source of visual information and using the monochromatic DAVIS where spatial features are more dominant than colour information. It is beneficial to consider the suitability of the sensor for the proper application and right recording scene.

Moreover, it is beneficial to consider that the data from the colour event sensor represents only the first step of colour detection in the visual pathway and not the whole pipeline. For instance, the human retina works as a colour data collector and the colours detection and merging happen in the later stage of the visual pathway Seymour et al. [2016]. Therefore, the colour sensors do not represent the full biological model of the colour detection and discrimination system, and they should only be considered the first step and the first layer in colour processing.

The next chapter presents a native end-to-end event-driven network pipeline to detect and classify events in a dense and complex environment using an unsupervised feature extraction algorithm and a supervised classifier network.

3.7 Conclusion

In this work, we conducted an in-depth investigation of the colour events sensor from the DVS pixels in a view to provide a better understanding of how the CDAVIS works in

various conditions. To rigorously investigate the effects of different colours and different camera biases, the exploration was only conducted on primary colours. We choose the three primary colours (i.e. Red, Green, Blue), and the investigation of secondary colours and many other complex patterns can be informed by these primary colours results. In this chapter, the characterisation results were based on three key factors: (i) an in-house built setup that supports a stable motion of the CDAVIS, (ii) a range of experimental conditions taking into account the camera biases, scene illuminations and scene contrast ratio and (iii) using two characterisation measurements such as the total number of events and event frequency/rate (events/second).

We have observed the following: (i) the number of events for each colour filter is primarily affected by the colour of the object and the background, (ii) the colour filters appeared to be sensitive to patterns with different colours where a slight change in colour induced by an object can lead to different event readouts pattern from the colour filters, (iii) it is challenging to identify the absolute illuminance (i.e. actual colour) of the objects purely from the DVS readout, that is because the DVS provide events with no information about colour, colour information can be recovered using intensity reconstruction techniques, (iv) it is possible to classify events data in scenes with identical objects which have the same shape and size but have different colours. Finally, this characterisation study provides more understanding of how to record better datasets and the datasets expected from the colour filter under a specific condition. However, in future work, more experiment need to be conducted to address the inconsistency in the colour output. This includes experimenting with a large range of colours and expands the recording duration to test and observe the change in sensor behaviour.

The work presented in this chapter is an exploration of the domain of colour events-based data and serves as a motivation for future work in colour data processing and analysis where DVS readout is considered without reconstructing intensity-based frames. An event-based colour processing can retain the camera’s technical capabilities in use, such as low data rate, high pixel bandwidth, and high temporal resolution.

Chapter 4

EVENT-BASED OBJECT DETECTOR AND CLASSIFIER FOR FRUIT DETECTION APPLICATIONS IN CLUTTERED SCENES

4.1 Introduction

Vision sensors in agricultural environments deal with high dimensional data streams conventionally acquired and analysed at a fixed sampling frequency. A system with a fixed sampling frequency limits the temporal resolution of the data-processing and the amount of data that can be processed. Firstly, the data streams must be sparse by sending only information when there is a change in the environment to address these limitations. Secondly, the incoming data stream can be processed parallel and asynchronous fashion. To fully demonstrate the advantage of an event-based approach, the target is to solve the problem of fruits detection as it is well studied and algorithmically understood using various other passive and active sensors such as Lidar, Radar and RGBD vision sensors.

Early event-based detection methods focused on simplicity to demonstrate the low-latency and low-processing requirements of event-driven vision systems. They assumed a stationary camera and tracked moving objects as clustered blob-like sources of events [Delbruck and Lichtsteiner, 2007, Delbruck and Lang, 2013, Litzenberger et al., 2006a]. In this case, only pixels that generate events are processed. Some example of applications are traffic monitoring and surveillance [Litzenberger et al., 2006a], high-speed robotic tracking [Delbruck and Lang, 2013], and particle tracking in fluids [Drazen et al., 2011]. Detecting of more complex, high-contrast user-defined shapes has been solved using Iterative Closest Point (ICP) [Zhenjiang Ni et al., 2012], gradient descent [Ni et al., 2015], Mean-shift and Monte-Carlo methods [Lagorce et al., 2015], or parti-

cle filtering [Glover and Bartolozzi, 2017]. However, they only work for a limited class of object shapes, and they are only useful where the objects’ appearance is clear and defined, simplifying the computations. However, methods to determine more robust features become necessary when the number of classes becomes large.

Most applications of EBCs have involved tasks that are particularly suitable to the sensor’s mode of operation, that is, sparse or static scenes with a limited number of non-dense moving objects in the sensor field of view. These applications provide ideal conditions for the use of the EBCs as they allow the sensor to compress the visual scene into sparse event streams significantly. Nevertheless, suppose EBCs are to, in fact, be used in a wide range of applications. In that case, they must also be able to operate in conditions that are challenging to the sensor’s mode of operation, for example, non-sparse, visually dense environments which cause the cameras to generate a very large number of events, significantly impairing the sensor’s ability to transmit valuable sparse data.

For that reason, an event-based object classification algorithm should reliably classify or recognise under a wide range of affine transformations, photometric changes and cluttered scenes. This chapter explores a means of performing robust object classification on event-based data streams for fruit detection and thoroughly examines and characterises the performance. This work primarily makes use of a classification algorithm based on FEAST algorithm [Afshar et al., 2019c], as it provides several important features which make it particularly well suited to the applications used in this section. This work also aims the following (i) to showcase the efficacy and efficiency of a detector algorithm, (ii) to validate its performance in constrained and unconstrained environments, and (iii) to tackle the problem of highly imbalanced event-based datasets.

The use of EBCs in the domain of fruits detection presents unique challenges and requires the development of specialised algorithms to take full advantage of the event-based paradigm offered by these sensors. Such algorithms are tailored to the data, resulting in highly efficient systems capable of handling the consistent change in the environment.

4.2 Research Questions

Based on the literature and recent advances in event-based visual detection, below are the research questions:

1. Can we use detect objects in highly textured scenes with non-uniform sensor motion?
2. Can we extract robust and discriminative features from the object of interest in dense and cluttered scenes?
3. How can we classify objects using highly imbalanced and noisy data?
4. Will supervisory signals improve the quality of the features representations and lead to better classification performance?

4.3 Contribution

The field of event-based vision in agricultural applications is recent, with no existing work on event-driven fruits detection and recognition. This section makes the following contributions to the existing body of knowledge:

- Derives and demonstrates a mechanism for performing classification on event-based data for binary classification problems for scenes with multi-objects in the same field of view and using recorded datasets from the constrained environment in a laboratory setup and unconstrained environment from real-world scenes and comparing the model performance.
- Quantifies the performance of the feature extraction and classification pipelines to validate the information encoded in spatio-temporal patterns and serves to validate the event-based processing paradigm for complex and cluttered visual data.
- Investigates various ways of using supervisory signals (i.e. ground truth) in feature extraction to apply feature selection to improve the classification performance.
- Provides an in-depth investigation into the effects of spatial downsampling on the classification performance for event-based data.

4.4 Methodology

This section describes the structure and nature of the events generated by the EBC. The method used to record data in different environments, the method used to label the dataset and the metrics used to report sensitivity, specificity and informedness from the event streams. The section further details a complete event-based detection and classification network.

4.4.1 Generating of Event-based Dataset

The dataset was captured using the DAVIS [Brandli et al., 2014] sensor. We recorded data in two environments: (1) a real-world environment at the HIE and (2) a constraint laboratory environment as shown in Figure 4.1(a) and Figure 4.1(b) respectively. The motivation behind creating a laboratory setup is to obtain a baseline architecture on a small set of data with less complexity and controllable parameters such as scene illumination, the colour of the objects, movement of the camera etc. This will help identify the network’s weaknesses and strengths as early as possible. Given that it is difficult to obtain real-world agricultural data due to weather, seasons, and fruit ripeness schedules, having a laboratory setup that mimics these scenes is beneficial to create an unlimited amount of dataset. This will enable a more extensive and detailed analysis of the proposed algorithms.

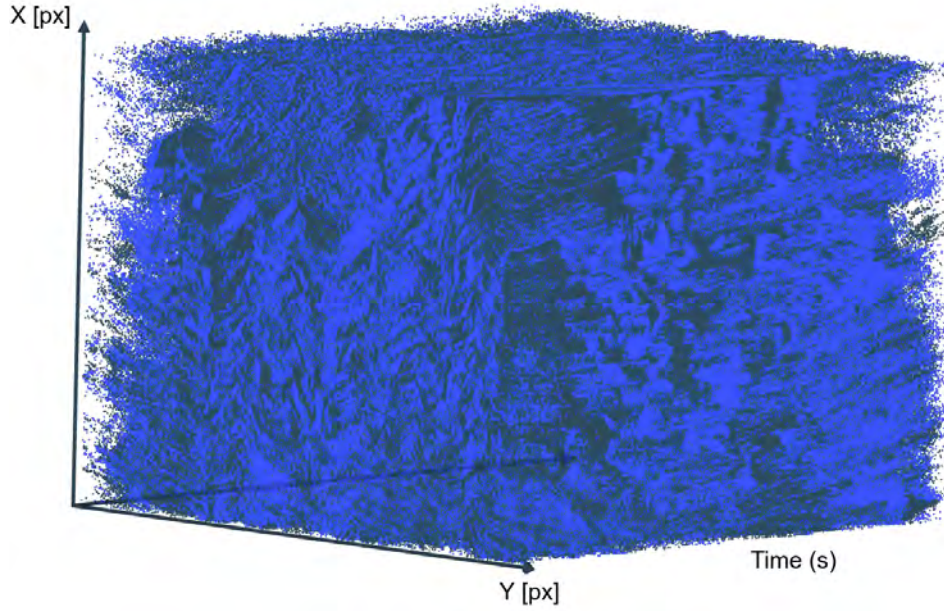
Each recording environment provides different types of challenges. For instance, in the lab recorded data, we can control the light source to adjust the events SNR, we can



(a)



(b)



(c)

Figure 4.1: Data collection setup. (a) Real world scenes at HIE. (b) Laboratory setup to control light and motion. (c) A dimetric projection of the events data from natural scenes which shows a dense spatio-temporal patterns.

control the object properties such as size, appearance, orientation, the degree of occlusion with other objects, as well as the speed of the platform. Real-world data consists of an unlimited number of non-ripened lemon fruits. It has all possible challenges that can help explore and exploit the robustness and effectiveness of the event-based algorithm, such as unstructured/noisy background, non-uniform contrast and illumination, non-linear camera motion, cluttered/occluded objects etc.

As shown in Figure 4.1(b), the laboratory setup comprised of a linear sliding platform (i.e. a small CNC robot capable of moving in 2D space), a 1304×984 , 12.48" red/black/white ink-based static display with a non-flickering LED. The 346×260

DAVIS camera is attached to the sliding platform, in which its position and velocity are controlled using G-Code commands. The camera is connected to a PC for data acquisition, and jAER¹ and DV² are used to write the events to an aedat4 file for post-processing. All the process is performed offline, which means that both data acquisition and recognition are separated.

The laboratory recorded data consists of six different recordings, and each recording has a minor variation such as overlap, object size and appearance. Below we provide a summary of the statistical analysis of the recorded data in Table 4.1 and Figure 4.2. The number of events is approximately equally distributed between both ON and OFF polarities and for both types of datasets, around 51% by 49%. A high imbalance class was observed by looking at the percentage of events for both classes. In this case, class 1 is the object of interest of the same type and class 2 is everything in the background. For instance, the classes ratio was an average of 85% by 15% for the lab recorded data and 97% by 3% for the actual world data.

Table 4.1: A summary of statistical properties for all laboratory recorded data.

		Duration (s)	Total Number of Events (millions)	ON Events (%)	OFF Events (%)	Percentage of events for Class 0 (%)	Percentage of events for Class 1 (%)
Shapes translation	Train	42.30	11.3	51.94	48.06	85.03	14.96
	Test	35.70	9.3	51.65	48.35	86.05	13.95
Two classes occlusion	Train	41.95	10.5	51.78	51.69	79.24	20.75
	Test	35.80	10.7	51.69	48.31	81.38	18.62
Multiple objects	Train	42.05	10.2	51.83	48.17	85.84	14.15
	Test	40.60	7.8	51.71	48.29	79.56	20.43
Different sizes	Train	42.35	10.6	51.75	48.25	79.51	20.48
	Test	39.35	9.1	51.77	48.23	83.82	16.17
One class occlusion	Train	42.35	8.5	51.75	48.25	80.47	19.52
	Test	40.05	7.8	51.65	48.35	82.70	17.29
Complex shapes	Train	9.05	8.7	49.31	50.69	94.75	5.24
	Test	12.65	11.7	49.59	50.41	94.19	5.80

For real-world scenes, the data was recorded by hand-holding the camera and scanning through the field consisting of several non-ripened lemon trees where the colour of the fruit is similar to the colour of the leaves. Unlike the laboratory recorded data that is fully controlled, the data bandwidth is affected by human motion and the soil floor in real-world scenes. Due to the uneven sensor motion, most of the dataset was occupied by the ego-motion background cluttering. Still, the realistic outdoor setting produces data properties that make developing high-level event-based computer vision solutions challenging. The recordings were performed at different points of view from the trees and at different directions to diversify the event’s output, which provides different semantic recording scenarios of the same area and includes artefacts from the environmental influences. Below we provide a summary of the statistical analysis of the recorded data in Table 4.2 and Figure 4.3.

¹<https://github.com/SensorsINI/jaer>

²<https://inivation.gitlab.io/dv/dv-docs/docs/getting-started.html>

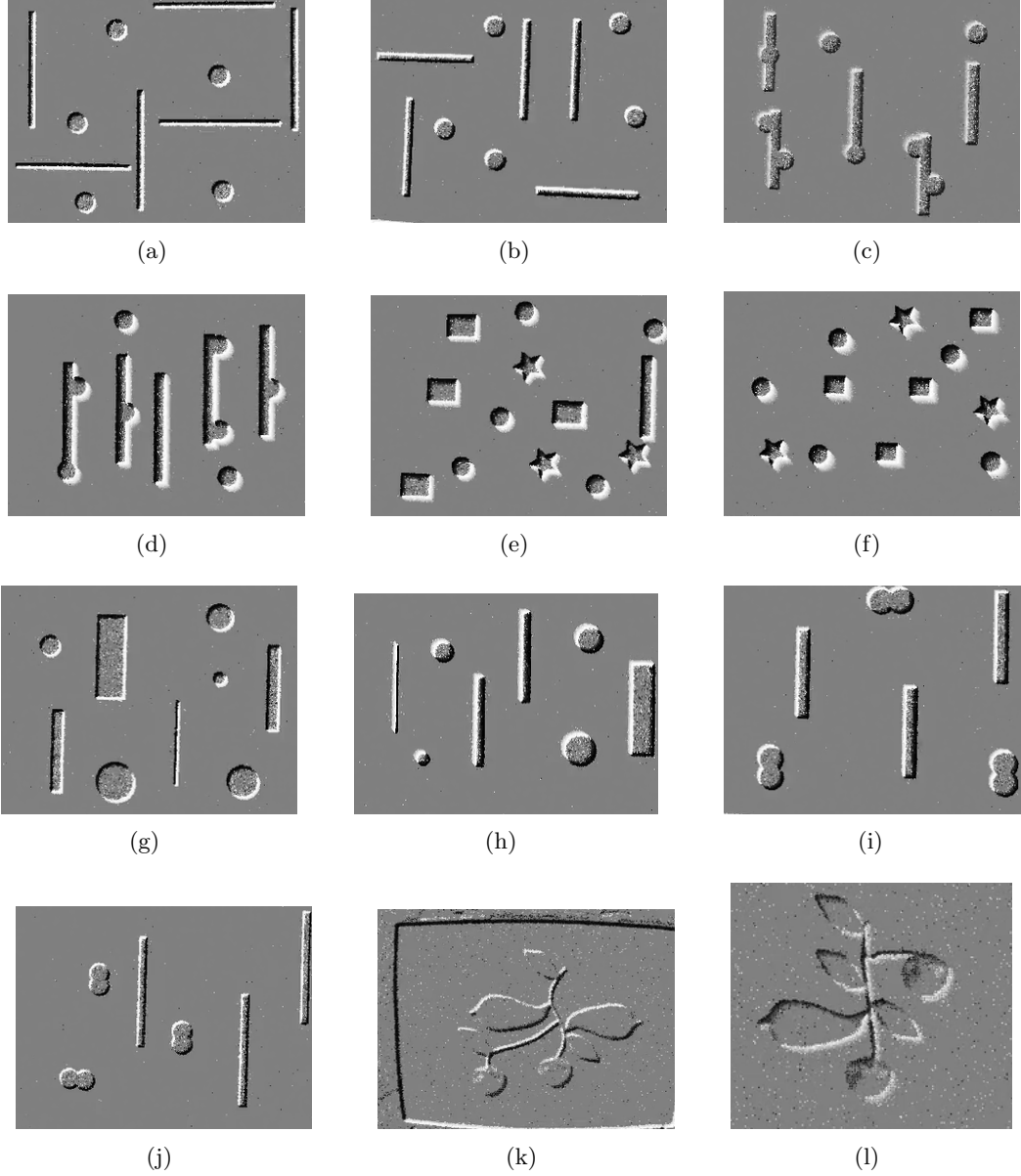


Figure 4.2: Laboratory recorded datasets showing images from training and testing sets. (a) and (b) train and test of Shapes translation. (c) and (d) train and test of Two classes occlusion. (e) and (f) train and test of Multiple shapes. (g) and (h) train and test of Different sizes. (i) and (j) train and test of One class occlusion. (k) and (l) train and test of Complex shapes.

Table 4.2: A summary of statistical properties for the real-world scenes.

		Duration (s)	Total Number of Events (millions)	ON Events (%)	OFF Events (%)	Percentage of events for Class 0 (%)	Percentage of events for Class 1 (%)
Sequence 1	Train	4.28	3.5	49.48	50.51	97.476	2.52
	Test	3.34	3.5	51.64	48.35	93.51	6.48
Sequence 2	Train	2.44	3.5	51.82	48.17	97.94	2.05
	Test	2.93	3.5	52.49	47.50	95.93	4.06
Sequence 3	Train	2.94	3.5	52.92	47.07	97.85	2.14
	Test	2.66	3.5	52.74	47.25	97.72	4.06
Sequence 4	Train	3.27	3.5	52.94	47.05	97.49	2.50
	Test	2.33	3.5	51.76	48.23	97.51	2.48
Sequence 5	Train	3.22	3.5	51.06	48.94	97.95	2.04
	Test	1.53	3.5	53.08	46.91	99.05	0.94
Sequence 6	Train	3.02	3.3	51.85	48.14	98.21	1.78
	Test	4.29	4.2	52.20	47.79	99.05	0.94

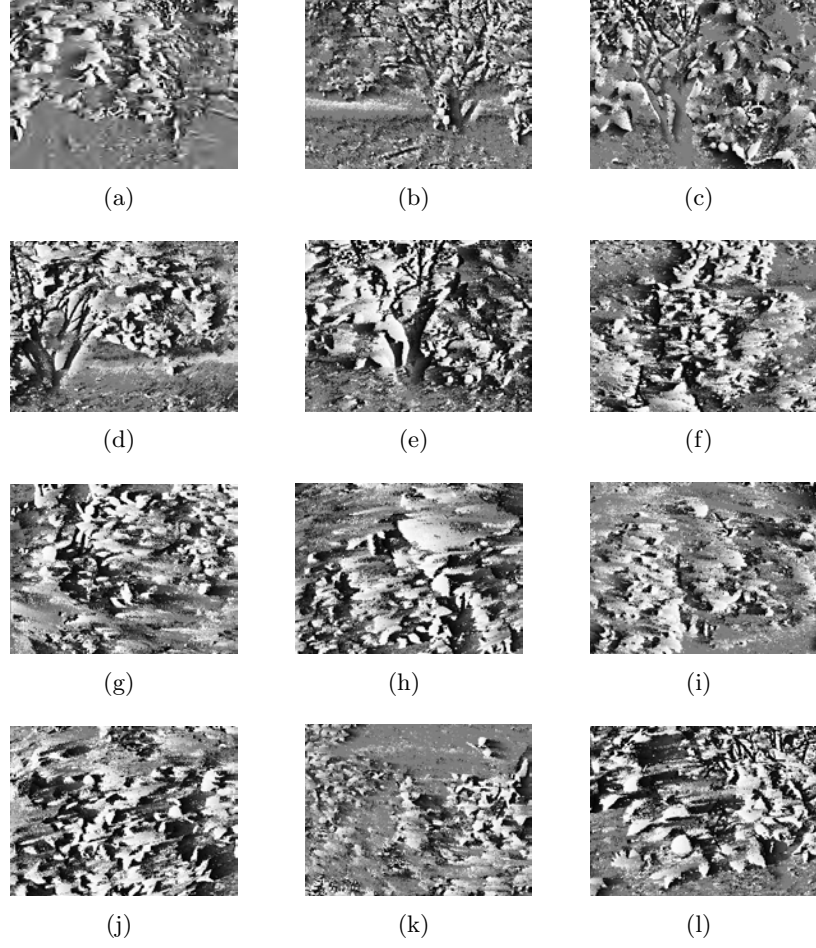


Figure 4.3: Real world datasets showing images from training and testing sets. (a) and (b) train and test of Sequence 1. (c) and (d) train and test of Sequence 2. (e) and (f) train and test of Sequence 3. (g) and (h) train and test of Sequence 4. (i) and (j) train and test of Sequence 5. (k) and (l) train and test of Sequence 6.

4.4.2 Labelling the Dataset

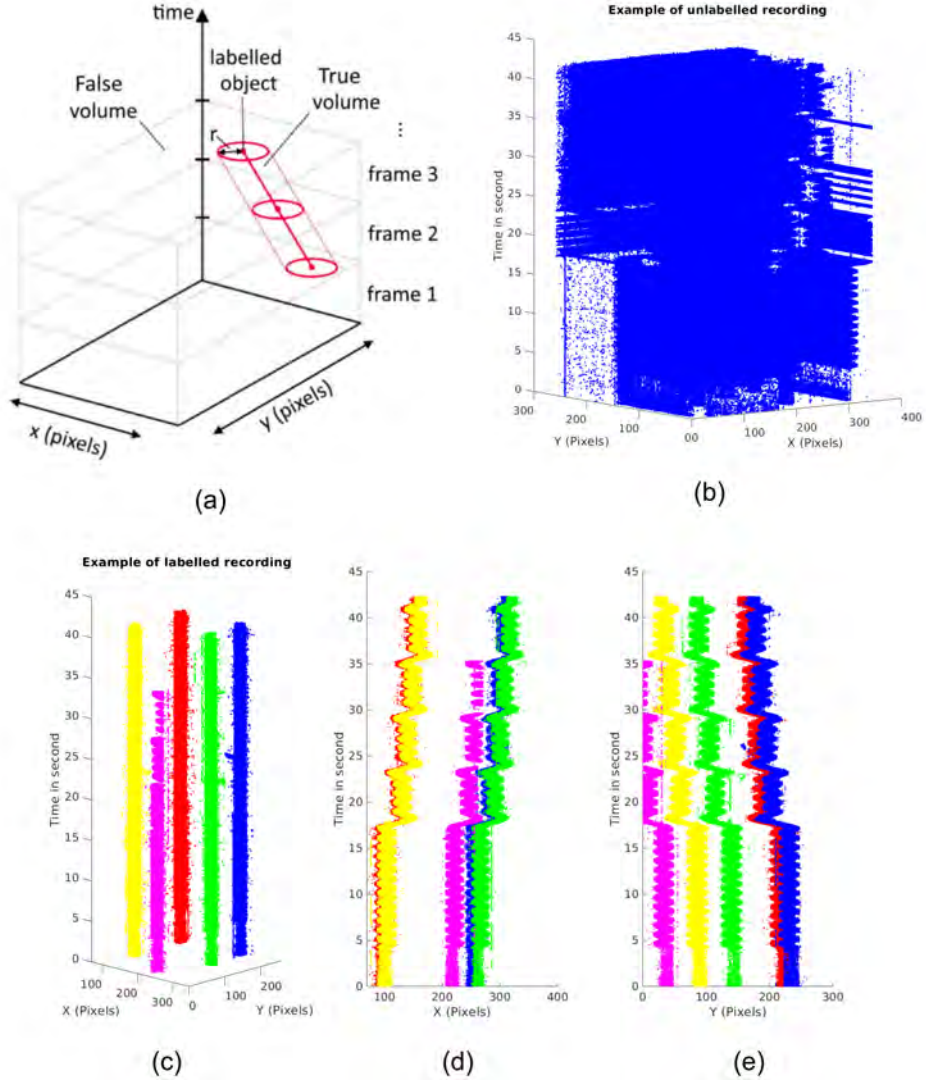


Figure 4.4: Data labelling. (a) Illustrates the method used to calculate event volumes' sensitivity and specificity around labelled data points. A volume of radius r around a line connecting the labelled points marks the boundary between true and false volumes. The event density of each sub-region designates its volume as a positive or negative volume depending on whether it is above or below the mean density of the recording as a whole (Source: Afshar et al. [2019b]). (b) Show the events data for the whole recording for both polarities. Panels (c), (d), (e) show the labelled objects for recording 1 in a dimetric projection and across the x and y-axis, respectively. Colour indicates the object ID of the same class.

The labels for the event-based data were obtained by building an image-like representation (i.e. time surface frames) from the events and then manually annotating them using an annotating graphical user interface tool built by Afshar et al. [2019b] as shown in Figure 4.5. The labelling was purely performed on the events stream without using any external sensor type (e.g. a conventional CCD or intensity frames). The labelled datasets were generated using a multi-stage labelling and editing procedure that involves viewing and labelling visible objects in each recording using a labelling interface

developed by [Afshar et al., 2019b], allowing the user to move forward or backwards through 2D time surface frames of the event stream at arbitrary frame rates with a maximum sampling frequency of 1000Hz. Target entry and exit points and segments of the trajectory exhibiting acceleration and the object’s radius at each timestep were all marked manually. These marked points were then linked programmatically via linear interpolation as illustrated in Figure 4.4.

During labelling, the centre of the object and its radius were labelled at each frame. The data points are linearly interpolated with each other to form a continuous stream in the event-based data.

This labelling scheme ensure that every incoming event has a single label. In our case, in each recording there are two labels, one that refers to the main class and one that refers everything else such as the entire background.

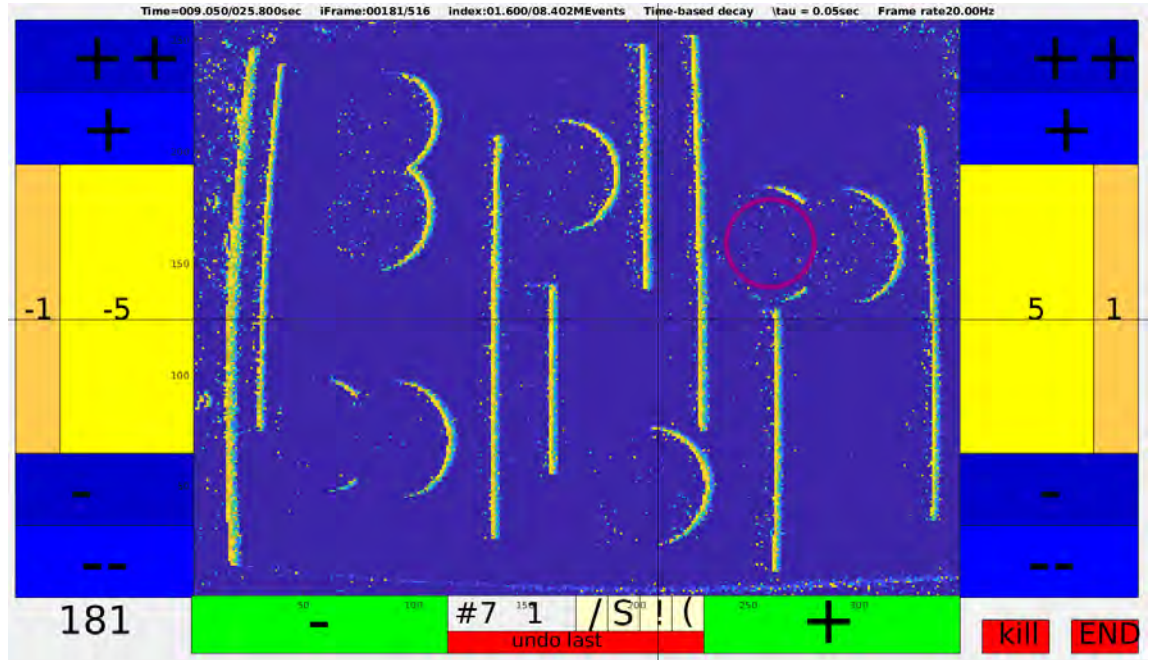


Figure 4.5: Data labelling graphical user interface (GUI).

4.4.3 Measuring Evaluation Metrics

The most direct measure of the utility of any processing pipeline is the classification accuracy achieved. However, this particular measure can be both misleading and computationally expensive in most cases. That is because the data is not always well balanced for each instance and obtaining a rigorous figure of merit for any classification system require repeated training of classifiers which can be time-consuming and resource-intensive.

In this work, the algorithms presented are designed to be operated entirely in the event-based domain from the sensors to the detectors. For this reason, the timestamp resolution (in μs) of the sensor is maintained in the whole architecture.

A robust evaluation metrics approach is required to quantify a given event stream sampled at 1MHz with the frame-based human labelled datasets (e.g. ground truth)

sampled at a lower frequency (e.g. 1KHz). For that reason, the evaluation metrics must account for the extreme differences in event rates produced by different recording conditions, to avoid bias and skewness in the data. The metrics must also assess the boisterous raw events of the sensor in the same manner as the highly sparse detection output event streams.

Event density in the event stream was used as a metric to evaluate the network performance. This metric separates the spatio-temporal events stream to distinctive volume with either a positive or a negative state. These states are then compared with the labelled dataset, which indicates whether the corresponding volume contains the correctly labelled objects or not. As illustrated in Figure 4.4(a), the spatio-temporal volume outside this region and in frames with no labelled object is designated as False, and the spatio-temporal volume slice surrounding the trajectory of a labelled object by radius r is designated as True for each frame.

The volume is set as positive when the event density is above the global event density of the whole recording for any spatio-temporal volume. Conversely, the volume is designated as negative if the event density in the volume falls below the global event density of the whole recording. Event streams with different event densities and noise characteristics can be directly evaluated and compared by calculating the mean True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN) volumes of each recording.

Using these volume-based measures, the event-based sensitivity and specificity of a particular event stream can be calculated using:

$$Sensitivity = \frac{\overline{TP}}{\overline{TP} + \overline{FN}} \quad (4.1)$$

$$Specificity = \frac{\overline{TN}}{\overline{TN} + \overline{FP}} \quad (4.2)$$

$$Informedness = Sensitivity + Specificity - 1 \quad (4.3)$$

$$Accuracy = 1 - \frac{missclassified}{allobservations} \quad (4.4)$$

The accuracy is a measure to compute how many instances were miss-classified across all the observations. Sensitivity (Equation 4.1) is the same as recall, it covers all TN, and it is appropriate when the focus is on minimising the FN. For example, understanding how many were correctly predicted as class 1 or not against the ground truth. Specificity (Equation 4.2) covers all TN instances. For instance, if we want to detect the correct labels for class 1, and we do not want the other features to be

detected as class 1, then, in this case, FP is intolerable. Informedness, as shown in Equation 4.3, provides a single statistic that captures the performance of a binary classification network and quantifies how informed a predictor is for the specified condition. Informedness measure helps in avoiding the biases of other common statistics which exist in the accuracy and precision metrics, as they are susceptible to population prevalence and label bias. For that reason, informedness is a more efficient accuracy measure that is also suitable for the highly imbalanced event-based datasets in which the vast majority of the spatio-temporal volumes are labelled as False regions.

4.4.4 Event-based Processing and Feature Extraction

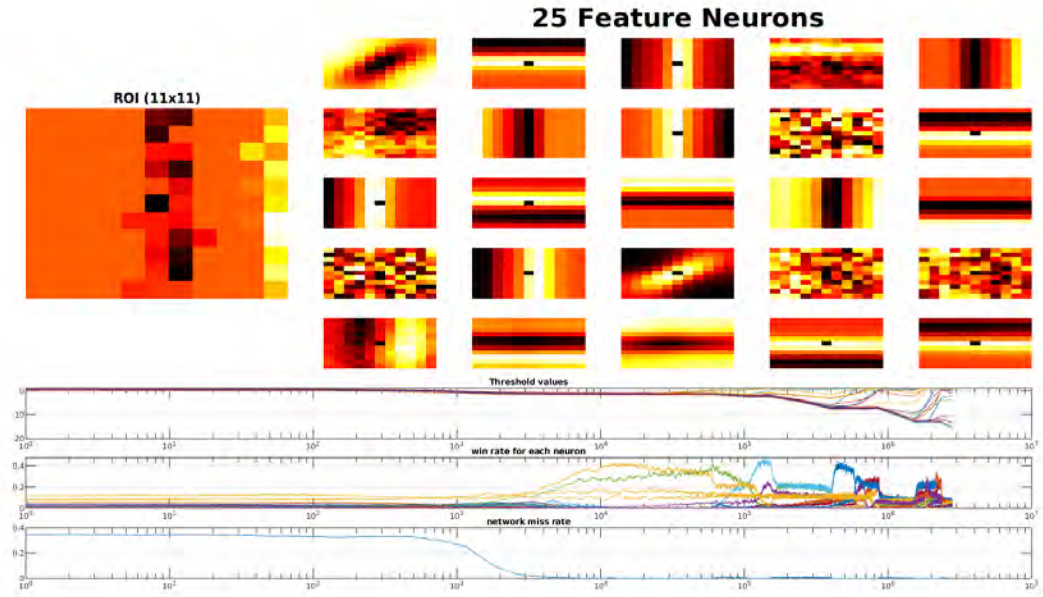


Figure 4.6: FEAST Training. **Top panels:** Shows the adaptation of various neural signals in the network over a single independent training cycle overtime. **Bottom panels:** Shows the progression of learning overtime such as the change in threshold adaptation per neuron, winner count for each neuron and the missing events (i.e. events filtered out by the network).

Event-based processing algorithms require memory of recent events as input which should be processed as a function of time. That is because event cameras output a continuous stream of events encoding the time, location, and polarity, making each event carry little information about the scene. An events memory can be generated via a range of methods which are investigated in Afshar et al. [2019a]. The method used in this section is the exponentially decaying event time-based surface, which outperforms other types of memory surfaces.

Figure 4.7 shows an illustration of the event context extraction from time surfaces in response to an incoming event e . The use of time surface reduces the data stream into two-dimensional representation, making it possible to generate frame-like representations that are continuous in time, as shown in Figure 4.7(b). The frames can be

Algorithm 1 FEAST Algorithm

Require: $e_i = [x_i, y_i, p_i, t_i]^T$, $i \in \mathbb{N}$
Ensure: $w_n(x, y)$, $n \in 1..Z$

- 1: where Z is the number of neurons
- 2: and $x = [-R..R]$ and $y = [-R..R]$
- 3: and R is the radius of the ROI
- 4: **Initialise:** $T_i \leftarrow -\infty$, $w \leftarrow w^0$, $\theta \leftarrow \theta^0$
- 5: where w^0 and θ^0 are random arrays with values between 0 and 1
- 6: **for** each event e_i **do**
- 7: $T_i(x_i, y_i) \leftarrow t_i$
- 8: $P_i(x_i, y_i) \leftarrow p_i$
- 9: $ROI(x, y) \leftarrow e^{t-T_i(x+x_i, y+y_i)/\tau_0}$
- 10: $d \leftarrow ROI(x, y) / ||ROI(x, y)||$
- 11: **for** each neuron $n \in 1..Z$ **do**
- 12: $\delta_n \leftarrow \langle d, vec(w_n(x, y)) \rangle$
- 13: **end for**
- 14: $q \leftarrow argmax(\delta_n)$
- 15: **if** $\delta_n < \theta_n$ for any $n \in 1..Z$ **then**
- 16: $w_m(x, y) \leftarrow (1 - \eta)w_m(x, y) + \eta ROI(x, y)$
- 17: $\theta_m \leftarrow \theta_m + \delta\theta^+$
- 18: **else**
- 19: $\theta_m \leftarrow \theta_m - \delta\theta^-$ for all $n \in 1..Z$
- 20: **end if**
- 21: **end for**

generated at regular time intervals making it possible to use conventional feature detection techniques. However, conventional approaches discard the sensor’s high temporal resolution and potentially result in non-optimal feature sets.

In this work FEAST algorithm, which was initially developed by [Afshar et al., 2019c], was used as the backbone for feature representation and extraction. FEAST algorithm is entirely event-based is an event-driven online unsupervised algorithm that reliably learns distinctive features and captures discriminative structures of the event-based data. It uses neurons or features with an individually adaptive selection threshold that updates iteratively using a competition control strategy. The simple adaptive selection threshold of FEAST maintain homeostasis between the activations of the majority of the neurons and the weight-update without the need to store the previous neural activity. Hence, it forgets the previously learnt features once new patterns appear in the incoming data. FEAST learning rule treats every input event with equal importance irrespective of them being detrimental to the network. The threshold is adapted for incoming events such that the features are consistently being contracted by current events and expanded by rejected or missed events. This event-based competition means that the thresholds of the neurons do not decay exponentially as a function of time but the only response to missed input data which represent information to the network.

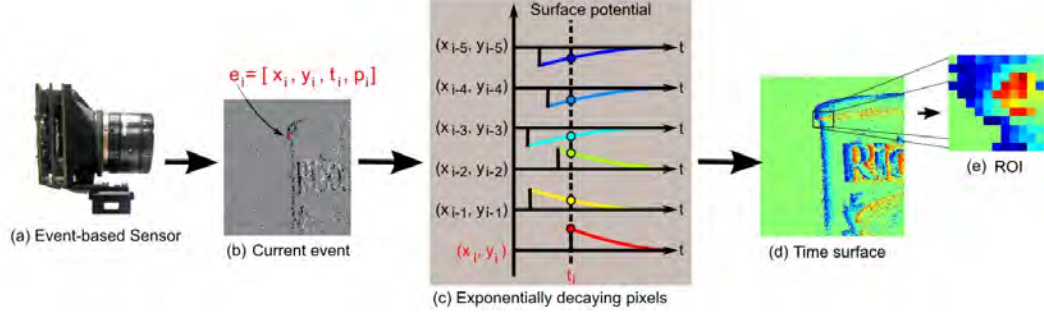


Figure 4.7: The construction of a spatiotemporal time surface and features from the event-based output of an EBC. (a) EBC generate a stream of events consisting of the pixel location, direction of the change in illuminance and time. (b) An example of the event stream. (c) An example of the decaying exponentials that generate the time surface. (d) The exponential decaying time surface. (e) The intensity of the colour encodes the time elapsed since the arrival of the last event from a particular pixel (Source: Afshar et al. [2019a])

As shown in Figure 4.7(b), the derived time surface is used to extract an ROI patch around every incoming event. This approach is beneficial because it does not process the entire image resolution, which can limit the feature information to the local spatial ROI and allows a computationally efficient event-based input to feature extraction. This method weighs each pixel as an exponentially decaying function of time. The method is implemented as described below:

$$T_i = \mathbb{R}^2 \Rightarrow \mathbb{R} \quad (4.5)$$

$$P_i = [-1, 1] \quad (4.6)$$

$$x : t \Rightarrow T_i(x) \quad (4.7)$$

$$S_i(x) = P_i * e^{(T_i(x) - t_i / \tau)} \quad (4.8)$$

where T_i contains the time-stamp of the most recent event at each pixel address x and y at the i^{th} event index. $S_i(x)$ is the corresponding exponentially decaying time surface which receives events for both polarities, and τ is the decay constant in seconds. In the case of active sensing (i.e. when the camera is actively moving and sensing the environment) and agriculture, the events rate is higher because all the objects in the field of view are salient and therefore depending on the circuitry biases and settings, the data bandwidth for both polarities is higher. Given the density and high volume in

the events readout, the design approach motivates more noise-robust object detection algorithms that also accounts for highly imbalanced classes. This is crucial because the network is strictly required to retain enough information about the object of interest where all the events triggered by this object are considered rare.

For every new incoming events (or observation), the time surface is updated, ROI of size $w * w$ around the event $e_i = [x_i, t_i, p_i]^T$ is selected for processing. The ROI_{*i*} associated with event is defined in Equation 4.9.

$$ROI_i = S_i(x_i + u_x, y_i + u_y) \quad (4.9)$$

where $u_x = [-R : +R]$ and $u_y = [-R : +R]$ are subject to the constraint in Equation 4.10.

$$\sqrt{x^2 + y^2} \leq R, \forall x \in u_x, \forall y \in u_y \quad (4.10)$$

Consequently the ROI at the i^{th} event index contains the time surface values S_i at time t_i from all pixels within the receptive field of size $2 * R + 1$, where R is the radius. Then the ROI_{*i*} is processed if Equation 4.11 is satisfied.

$$L < \sum_{x=x_i-R}^{x_i+R} \sum_{y=y_i-R}^{y_i+R} (T_i(x, y) > \Phi) \quad (4.11)$$

Where Φ is the event activation time interval, L is the number of activated pixels required, and x and y are subject to the distance constraint in Equation 4.10. The ROI will be accepted only when the number of recently activated pixels on the time surface within the radius R around the current event e_i is above L . In this case, events recentness is defined as a pixel that has received an event within Φ seconds. The ROI size was selected around the neighbouring 11x11 pixels. The ROI region is converted into a descriptor d in the form of a one-dimensional vector in order to perform further processing on the event, as shown in Equation 4.12.

$$d = vec(I) = [I_{1,1}...I_{w,1}, I_{1,2}...I_{w,2}, I_{1,w}...I_{w,w}]^T. \quad (4.12)$$

Following Equation 4.12, the descriptor is normalised through a division by its norm to achieve invariance to temporal scaling.

$$d = \frac{vec(I)}{\|vec(I)\|} \quad (4.13)$$

The normalisation of the d as shown in Equation 4.13 result in a time scale invari-

ance that can effectively discard velocity information in favour of feature robustness. It also means that the d generated from each event inject an equal amount of information into the network by Equation 4.14. For example, a faster-moving object in the scene with a higher magnitude would significantly affect the learned feature weights compared to slow-moving features. In this case, the normalisation step makes the network speed invariant. FEAST algorithm captures the most dominant spatiotemporal patterns observed in an event stream using two main techniques such as adaptation of all feature neurons toward the observed context allowing the features to evolve to match the incoming data continuously, and the learning rate balance across all neurons through the adaptive selection threshold that contract and expand to make each neuron more or less selective.

During feature extraction, a dot product is applied between the normalised descriptor d and each feature neurons w_n such as $\sigma_n = d \cdot w_n$. In this case, each neuron is compared to its selection threshold θ_n . The threshold is dynamic and given i features. The thresholds change based on two rules: (1) If the cosine distance between its weights and the input ROI is within the feature threshold, then the threshold is decreased by a fixed amount ΔI then the neuron with the largest dot product is selected as the winner, (2) If multiple features match the input, the best matching feature is selected, (3) If there is no match between the ROI and any of the feature neurons, then all the threshold are increased by a fixed amount ΔE then no neuron wins, and the network misses the event. To perform this weight update, a small mixing rate η is used to move the winning neuron toward the d slightly:

$$w_n = (1 - \eta) \cdot w_n + \eta d \quad (4.14)$$

Where w_n denote the weights of the winning neuron to the current input ROI and η is the mixing rate used to update the features. There are two types of mixing rates: one to move the threshold up and one to move it down. In this work, $\mu_{up} = 0.001$ and $\mu_{down} = 0.003$, and the number of neurons depend is a variable parameter that depends on the types of dataset.

In this case, FEAST learns unsupervised representations from the incoming events and each neuron is updated according to the number of events being fed to the system. The neurons will learn representation for the most salient objects. Initially, the features training is initialised to random points on the unit hypersphere. Given that the selection thresholds are also initialised at random, the threshold will increase for some neurons where every input causes a neuron to fire, and some neurons will not be activated and will not learn prominent features from the incoming events. Given that the scene is continuously changing and the camera is in constant motion, at least one neuron will learn a specific feature from the incoming data. For instance, some neurons will learn a feature for each class, others will learn noise and hot pixels, and few will learn rare events. Neurons with great receptivity capture all input events such that there are no missed spikes and no change in the selection thresholds of the more selectively

initialised neurons.

As shown in Figure 4.6 by the bottom panel, during the initial stage of learning, the relative magnitude of change in the threshold and miss event rate were low since only the threshold of a few neurons are adapting and becoming more selective. The rate of change in feature weight update is also low due to the network’s imbalanced activity, which only allows a few neurons to learn and get updated. The standard deviation of the spike rate is also low. As the training continues, the highly sensory neurons become more selective and more neurons with a more selective initialised threshold become activated and begin learning new features. This method of learning results in an increase in the rate of change in the feature weights and the selection threshold over time.

The number of activated neurons with decreasing thresholds increases to a tipping point, and the change in threshold begin to decline as fewer highly receptive neurons are left for adaptation. Simultaneously, a change in neurons weights and the variance in the spike rate of the neurons also decreases. As the training advances, the neurons weights and threshold begin catching the statistics and the dominant representation in the data such that the neurons orient toward the centroid of the most common spatiotemporal pattern clusters while the threshold takes on values in proportion to the spread of the patterns around these centroids.

Thus, all neurons eventually become more selective. The missed spike rate would cause the final most selective neurons to respond to input and begin adapting their weights. Once training is completed, the trained weights are stored in the memory. The trained weights are used during inference. During inference, FEAST adaptive threshold and weight update rule are disabled, the feature with the smallest cosine distance to the input is assigned to the incoming event regardless of the absolute value of the adapted selection threshold.

4.4.5 Per Pixel Spatial Downsampling

This section examines the effect of downsampling operations on the input test patterns during inference. The advantages of downsampling are significant [Cohen et al., 2018]. Firstly, it makes the network less computationally expensive and reduces resource requirements, resulting in increased accuracy for networks with the same hidden layer size. It requires fewer time-steps or input channels, speeding up the process and reducing computation time. Also, downsampling inherently reduces the input data size, requiring less bandwidth to transmit between the input and hidden layer and storing data in the memory.

Although the DVS output is already spatially sparse in contrast to frame-based sensors, our work focus on exploring whether the event rate can be further reduced, as the power consumption of an EBC is a function of the event rate both in terms of sensing and the processing of the data. Moving further up in our network pipeline and the processing chain, the size of the attributes of the network is proportional to the dimensionality of the input data resulting in further power and resource benefits when

reducing the resolution of the events data.

Reducing the spatial dimension of the data before the classification network can significantly impact the size, power consumption, and speed of the event-driven processing operations. However, it is essential to investigate how much information the system can afford to lose without harming the network performance during classification. The spatial downsampling implemented in this work is computationally inexpensive and easy to implement with one parameter to tune: the downsampling factor, which reduces data rate (i.e. total number of events) before embarking on computationally expensive operations, such as classification. As downsampling is a lossy process, the results containing the full resolution pattern should achieve the same accuracy bound as any of the downsampled variants.

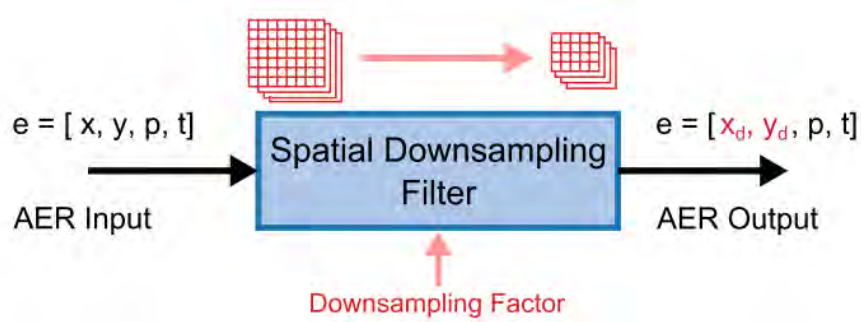


Figure 4.8: Diagram of the structure and function of the event-based per pixel downsampling filters. The spatial downsampling filter reduces the spatial resolution of the input data by a fixed downsampling factor, resulting in an output stream that maintains the temporal resolution of the input (Source: Cohen et al. [2018]). AER refers to Address Event Representation.

The downsampling mechanism is performed during inference and in between the STP filter and the classification network. In this case, the spatial downsampling affects the mapping from x and y addresses to input channels. As shown in Figure 4.8, the downsampling operates directly on each incoming event, and the pixels values are adjusted according to the downsampling factor while keeping the polarities and timestamps the same throughout the operation. The information contained within each event is an integer in nature.

The timesteps possess microsecond resolution encoded with a 32-bit integer value. The x and y pixel addresses are integers within the range of the camera frame, and the polarity values contain only a boolean value indicating the direction of change in brightness. As the output of the downsampling filter must conform to the same conventions as the input, the output events must contain only integer values. The range of permissible downsampling factors is limited only to integer values, and the chosen downsampling factors range between 1 and 28.

At minimum downsampling factor 1, the events spatial resolution is $346 * 260$, and at maximum downsampling factor 28, the resolution becomes $12 * 9$, any downsampling value greater than 28 will not result in a significant change in the spatial resolution or the dynamic of the moving object. As shown in Figure 4.9 we displayed the feature

map through a specific neuron and applied the downsampling mechanism across all the downsampling ranges.

In this case, the objects in the scene become much more prominent, and all the information become much more visible as the size of the spatial resolution reduces. For example, for a downsampling factor $\sigma = 10$, every ten neighbour events within the 10×10 window are combined into a single event where the resolution after downsampling is ten less than the original resolution (i.e. $34 \times 26 \Leftarrow 346 \times 260$).

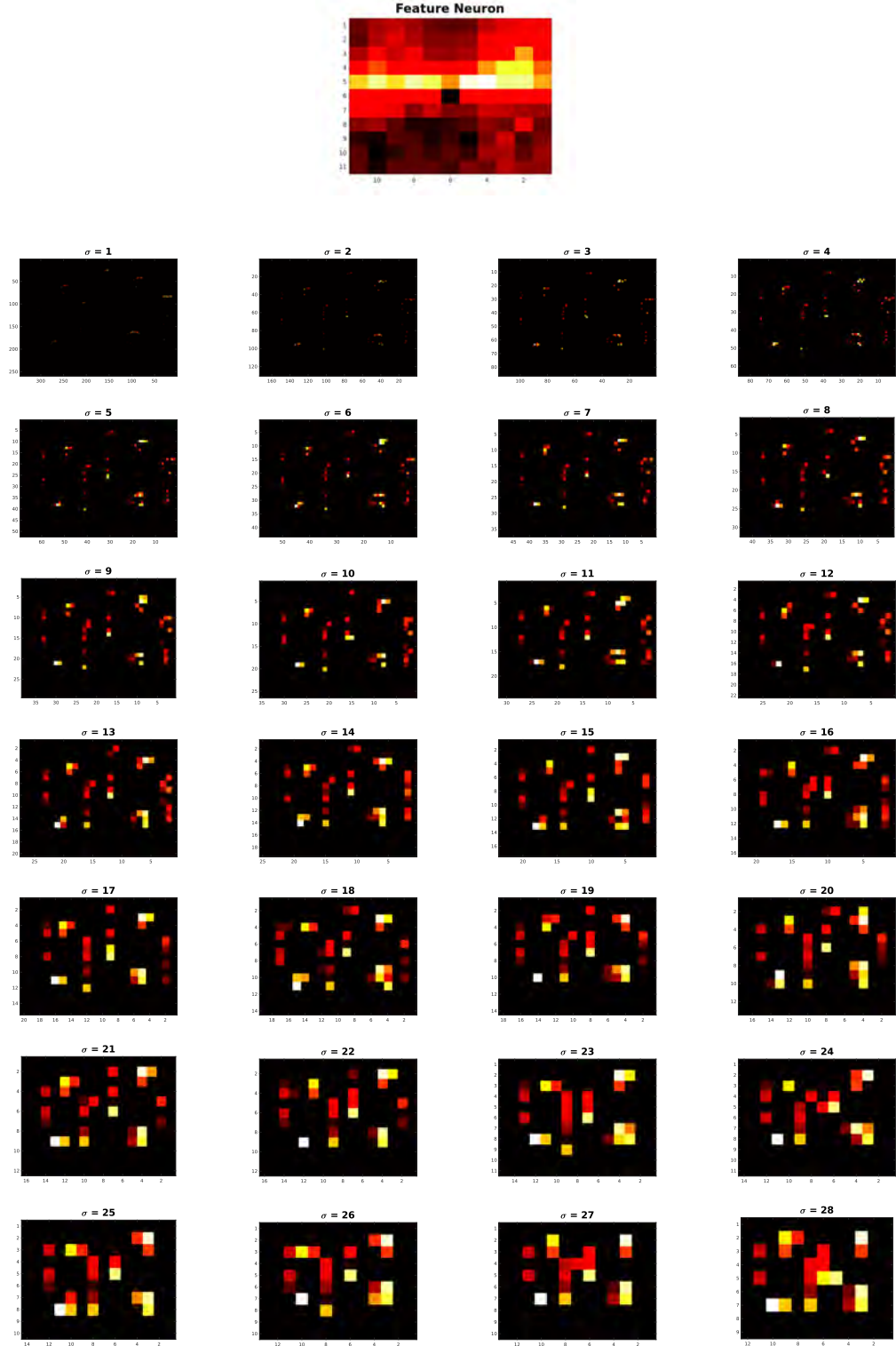


Figure 4.9: The exponential time surface of the feature map was generated at 28 different spatial resolutions alongside their associated feature neuron. All surfaces are all updated in an event-based fashion for each incoming event.

In this work, we show that spatial downsampling improves the accuracy of the classification system under different circumstances while still reducing the effective data rate and that there is a strong correlation between the size of the object and the selected downsampling factor. These findings are particularly significant for designing processing systems and algorithms that deal with many event data in a complex and dense environment. Results of the downsampling mechanism are reported in Section 4.5.5.

4.4.6 Spatial Temporal Pooling STP Filter

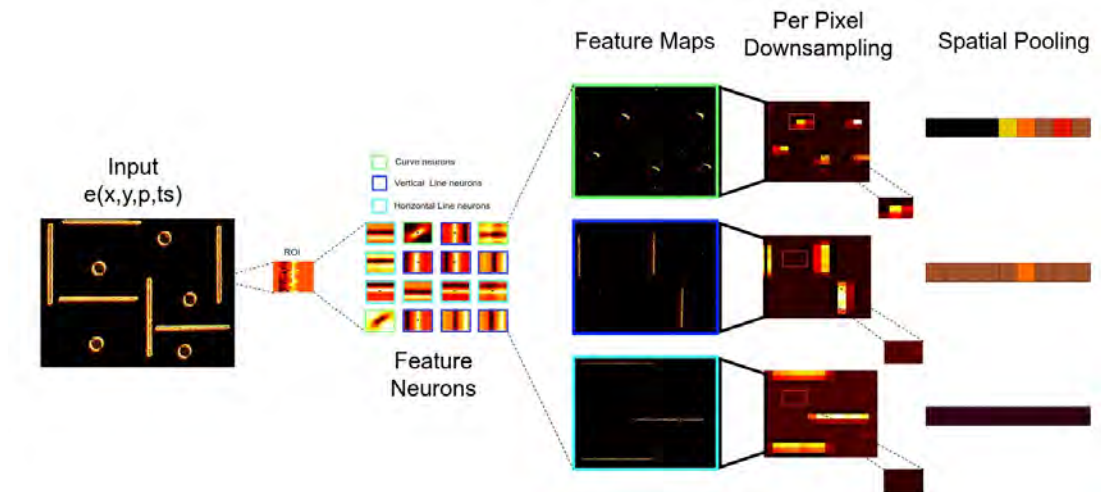


Figure 4.10: FEAST network with feature neurons. Shows the procedure from extracting features from the data to spatial pooling/flattening the features. The process starts with the input event stream followed by an 11x11 ROI associated with 16 neurons, each neuron is colour coded representing which part of the data was learned, a feature map was created for each neuron, and then a per pixel downsampling approach was applied to reduce the resolution, finally, the last layer pool the features from the feature maps using a 3x3 receptive field.

In conventional computer vision algorithms, STP usually involves aggregating frame-level features into video-level features. Typical pooling operation (e.g. max or mean-pooling layer) aims to compress the information of two consecutive frames into one frame. However, the pooling layer processes the entire image using the sliding window technique, adding redundancy and unnecessary information to the network.

To take full advantage of the compact and sparse representation of DVS data, an event-based STP filter was designed to eliminate all the unchanging values, processing only the salient object. That means fewer kernels are required to represent DVS events than regular images in CNN. Additionally, since there is a large portion of image regions that are zeros in DVS events, pooling features on those empty regions are non-informative. A larger pooling receptive field at frontal layers helps to get a more

compact and meaningful description of sparse DVS image at an early stage of the network, instead of spending much computation resource on the noisy details.

Afshar [2020] has shown that 2D pooling outperforms 1D pooling. This is because the event generation methods involve the Pooling of raw sensor data over either time or space or both, significantly increasing the information content of each event. In this work, the STP filter is designed and applied in an event-driven way over the downsampled feature maps on both space and time to increase the information retrieved for each incoming event as described in Algorithm 26. Thus, the pooling operation is only performed when the events occur within the field of view, disregarding the inactive pixels. The feature extraction operation projects the raw event stream onto many sparsely populated feature surfaces, which shows where a particular feature occurs over the entire pixel array. For instance, a neuron that activates for vertical lines will only show lines on the feature map. Similarly, for the neurons that only activate for circle and horizontal lines, as shown in Figure 4.10, it can also act as events denoiser like an autoencoder [Ma et al., 2020]. A 3D of size $3 * 3$ receptive field is selected as a pooling window, performed after the per-pixel spatial downsampling.

Applying the 3D pooling layer overall feature maps avoids information loss as the size of the feature extraction layer expands, the effect of information loss due to Pooling becomes less significant. The 3D Pooling around the recent events for each feature map simultaneously is flattened to form a 1D vector including all the pixels information for the most recent pixel and the neighbouring pixels around it. The resultant is a pooling vector of size $\text{Pooling} = 3 * 3 * N$ where N is the number of neurons. The process is repeated for every incoming event producing the final STP_{filter} matrix of size $nEvents * \text{pooling_window}$, where $nEvents$ is the total number of events in the sequence. This matrix contains all the relevant information for each event, and it is then used in the cross-validation and classification network.

Investigating the effect of pooling layer window size was out of the scope of this work, as an increase in the window size will dramatically increase the dimension of the output data by a minimum of 10X-100X, resulting in several computational overheads.

4.4.7 Event-based Feature Selection

In machine learning, feature selection is vital for extracting meaningful and relevant features to build an efficient classification model, reduce computation complexity, and improve its generalisation ability. Generally, feature selection methods are divided into three categories: (1) filter methods Sánchez-Marono et al. [2007], the selection process is independent of classifiers and rank features according to the intrinsic properties, (2) the wrapper methods, which utilise the model’s predictive power to rank subsets of features, and (3) the embedded methods, where feature selection interacts with the machine learning process.

Unsupervised learning deals with finding hidden structures in unlabelled data. However, when the data is highly imbalanced, an unsupervised feature extraction algorithm learns the dominant class structure and under-prioritises the non-dominant class, es-

Algorithm 2 STP Filter Algorithm at Inference

Require: $e_i = [x_i, y_i, p_i, t_i]^T$, $i \in \mathbb{N}$

Ensure: $w_n(x, y)$, $n \in 1 \dots Z$

```
1: where  $Z$  is the number of neurons
2: and  $x = [-R..R]$  and  $y = [-R..R]$ 
3: and  $R$  is the radius of the  $ROI$ 
4: Initialise:  $S_i \Leftarrow 0$ ,  $S_{di} \Leftarrow 0$ ,  $T_i \Leftarrow -\infty$ ,  $T_{di} \Leftarrow -\infty$ ,  $P_i \Leftarrow -\infty$ ,  $w \Leftarrow w^0$ 
5: where  $w^0$  is the trained weight with values between 0 and 1
6: and  $d$  is the downsampling factor
7: for each event  $e_i$  do
8:    $T_i(x_i, y_i) \Leftarrow t_i$ 
9:    $P_i(x_i, y_i) \Leftarrow p_i$ 
10:   $ROI(x, y) \Leftarrow e^{t - T_i(x+x_i, y+y_i)/\tau_0}$ 
11:   $d \Leftarrow ROI(x, y) / ||ROI(x, y)||$ 
12:  for each neuron  $n \in 1..Z$  do
13:     $\delta_n \Leftarrow \langle d, vec(w_n(x, y)) \rangle$ 
14:  end for
15:   $q_i \Leftarrow argmax(\delta_n)$ 
16:   $x_{di} \Leftarrow x_i/d$ 
17:   $y_{di} \Leftarrow y_i/d$ 
18:  if  $T_{di}(x_{di}, y_{di}, q_{di}) < 0$  then
19:     $T_{di}(x_{di}, y_{di}, q_i) \Leftarrow t_i$ 
20:     $P_{di}(x_{di}, y_{di}, q_i) \Leftarrow p_i$ 
21:  else
22:     $T_{di}(x_{di}, y_{di}, q_i) \Leftarrow (1 - \beta)T_{di}(x_{di}, y_{di}, q_i) + \beta * t_i$ 
23:     $P_{di}(x_{di}, y_{di}, q_i) \Leftarrow (1 - \beta)P_{di}(x_{di}, y_{di}, q_i) + \beta * p_i$ 
24:  end if
25:   $STP_{filter} \Leftarrow P_{di} * T_{di}((x_{di} - 1 : x_{di} + 1), (y_{di} - 1 : y_{di} + 1), :)$ 
26: end for
```

pecially when the data from the non-dominant class is considered rare. The problem becomes more severe and challenging in dense and cluttered environments. To solve this problem, we implemented an event-driven feature selection technique based on the filtering method that makes use of the supervisory signals from the ground truth as input to FEAST and compared the performance with the same architecture where the supervisory signals were not used. Section 4.4.7.1 and Section 4.4.7.2 describes in detail the procedure with examples and provide an overview for the whole network pipeline.

4.4.7.1 Without Supervisory Signals

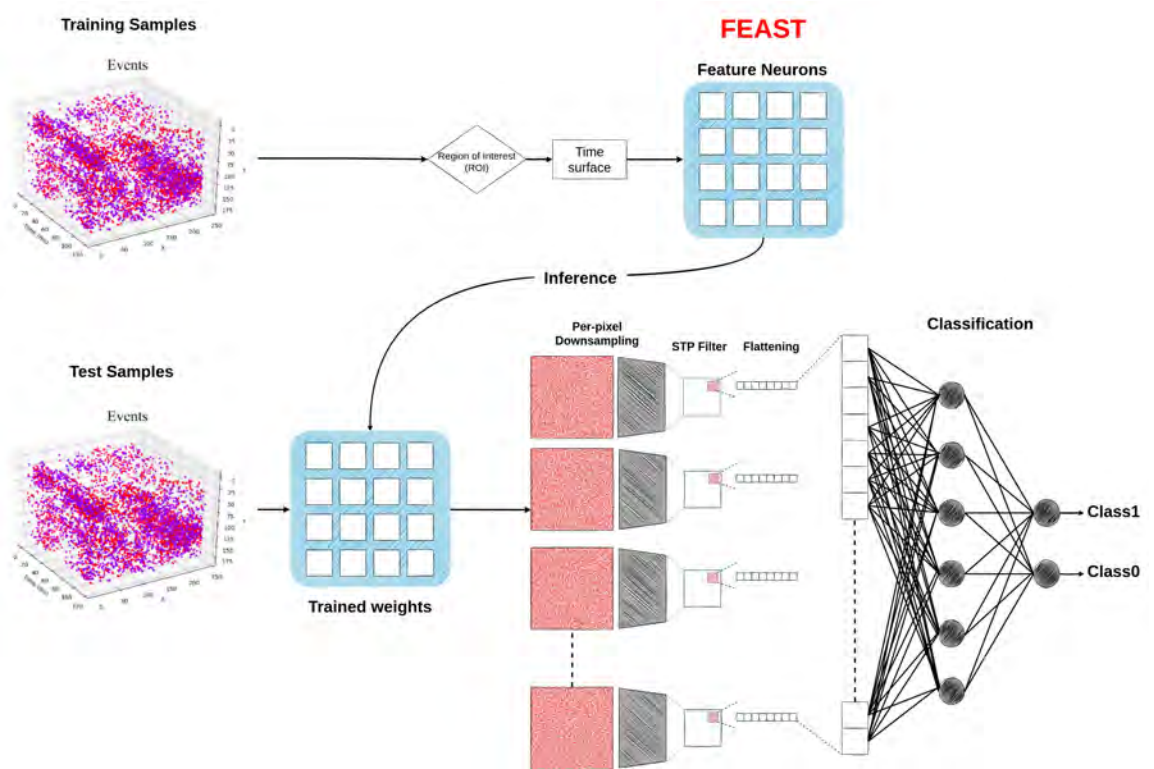


Figure 4.11: First network pipeline which is called "Standard architecture". The network is unsupervised from feature extraction to inference to classification.

A diagram of the whole network architecture used in this work is shown in Figure 4.11. Here we present the baseline architecture without applying feature selection. The events from the EBC are used to generate a time surface [Afshar et al., 2019a] with an exponential kernel on which FEAST) operates. An example of the features generated for one of the tests sequences from the lab recorded dataset is presented in Figure 4.15. In this case, all the events are fed to FEAST to extract unsupervised features based on the presence of the objects in the scene. It was apparent that more features represented the lines than the circles.

The lab recorded data are controlled and more structured regarding lights, motion, and sensor biases, with fewer noise events than real-world scenarios. Therefore, the network does not generate any variant for the noise features. However, in real-world scenes, the network produces many variants of the noise features. Since the output

of these features does not correlate with any particular class, they effectively act as naturally evolved noise detectors, reducing the event density for the rest of the network. FEAST algorithm extracts multiple variants of each object, including a variant of the noise feature from the event stream.

These noise features point to subtle statistical structure in the noise, which likely depends on the dynamic in the environment and the sensor bias settings. For that reason, it is impossible to hard-code all the variants in noise features to cover all biases and scene conditions. Therefore, FEAST account for as many noise features as possible automatically. After the convergence of the feature detector, the training data are converted to feature space (i.e. feature map), and the weights (i.e. feature neurons) are frozen to be used at inference.

At inference, the unseen test sequences were passed through the same FEAST layer in which the adaptive threshold and the weight update mechanism were disabled. In this case, each incoming event from the test set is projected into a feature map. The information was then pooled from each feature map using the STP filter with a 3×3 receptive field. A label is assigned for each pooled event and presented to the classifier through a supervised training regime. After each STP operation, the 3×3 receptive field is flattened to a 1-D vector containing information about the recent event and its neighbours.

The process is repeated until the end of the test set is reached, resulting in a matrix of size $9 * N_{neurons} * N_{events}$ where $N_{neurons}$ is the total number of neurons and N_{events} is the total number of events in the test set. This matrix is then used as input to the classifier network. The training set consisted of random segments from each recording sequence of about 50%, and the remaining 50% made up the test set.

There was a slight variance in the spatiotemporal patterns in each recording category, such as changes in velocity, pose, and the periods of partial occlusion as the objects enter and exit the field of view. This intra-recording variance significantly adds to the dataset’s complexity and makes it more diverse and challenging.

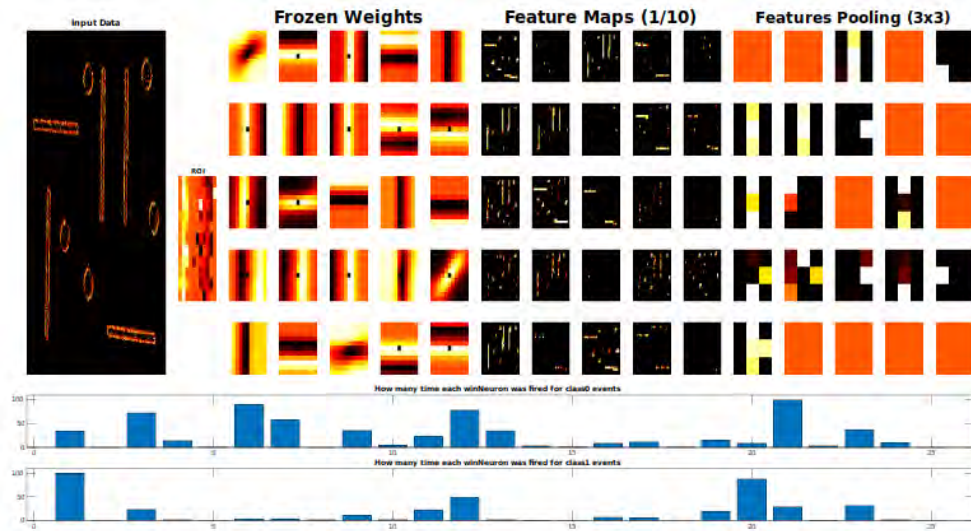


Figure 4.12: Inference with imbalanced features. Shows the uses of the trained weight during inference to construct a feature map along with the feature pooling layer. The bottom panels show the number of winner neurons for each class over time.

4.4.7.2 With Supervisory Signals

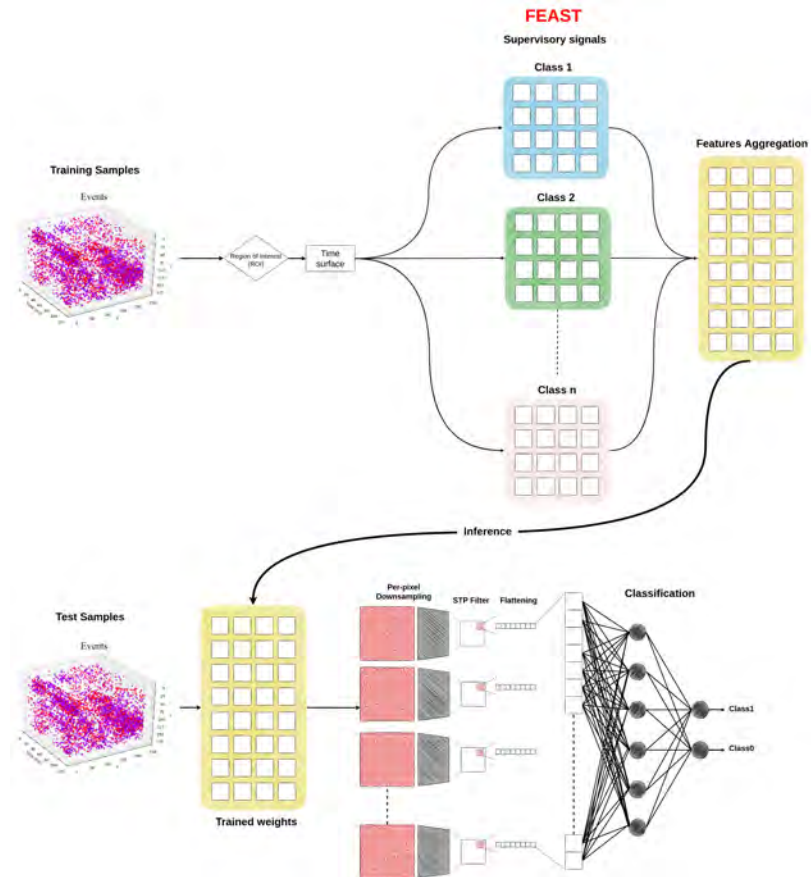


Figure 4.13: Second network pipeline which is called "Mixed weights architecture". Supervisory signals were fed as input to FEAST to extract robust and discriminative features. The features were aggregated before using them during inference.

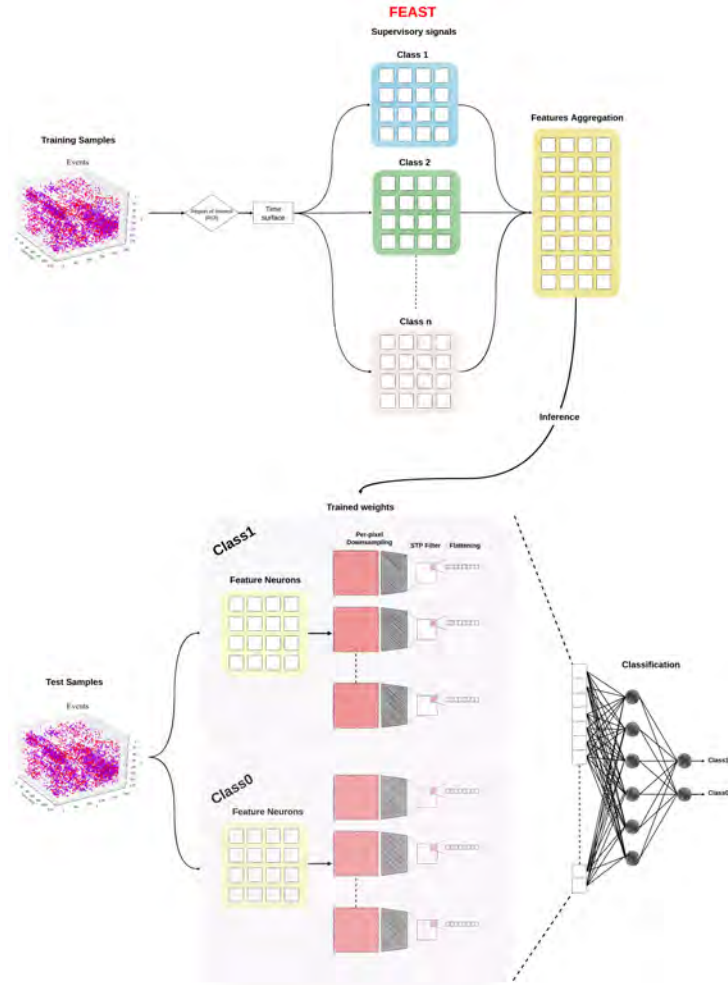


Figure 4.14: Third network pipeline is called "Dedicated weights architecture". In this network architecture the supervisory labels were used during training and inference.

In Figure 4.13 and Figure 4.14 we present two different architectures where the supervisory labels are used as part of the training in FEAST. Here we make use of feature selection in the feature extraction phase. The motivation behind using the supervisory signals is to solve feature biases in conditions where the majority of the feature neurons belong to the dominant class and under-representing the non-dominant one, resulting in feature skewness toward one particular class. Given that our datasets were generated by moving the camera in a dynamically changing environment, the entire field of view becomes salient, making the events stream noisy and highly imbalanced. For instance, in conditions where the camera is moving, the events stream are imbalanced and the events polarities and noise patterns due to the non-ideality of the sensor arbiters.

In this condition, it was evident that during feature extraction, the model became more selective for the class that has more events and less than 10% of the neurons end up firing for the non-dominant class, making the network a good background and noise detector but not a suitable to detect the objects of interest. It was then apparent that using supervisory labels is a great potential to solve data biases on the feature extraction level.

This has the advantage of forcing the network to learn features specific to one class, resulting in many robust representations and producing an equal amount of features per class, avoiding feature bias. Instead of learning unsupervised features from all the input streams, FEAST training was divided into two phases, one for each class independently. Thus, two weight templates (i.e. neuron population) and two threshold arrays were initialised with random points. For example, a population of neurons is assigned for each class weight, such as n neurons for class1 and m neurons for class0. The exact mechanism of threshold adaptation and weight update rule is applied as described in detail in Section 4.4.4.

During the initial training phase, if an incoming event belongs to class1, this event activates the population of neurons for class1 resulting in weights update only for this group of neurons and an increase in selectivity toward one class. The exact mechanism is applied for events that belong to class0. After the convergence of the feature detector, an equal number of unsupervised features are produced for each class, including more information about the object’s class in the scene and noise features.

For the second architecture in Figure 4.13, the trained weights for each class were combined into one unified template, unlike during training where the supervisory signals are used to produce better features, at inference the whole process of pooling information using the mixed weight template is entirely unsupervised.

On the contrary, the third architecture, as shown in Figure 4.14, is divided into two phases during training and during inference, segregating the features for training and test dataset, giving the model additional supervision over the incoming signals in each processing layer. At inference, the selection threshold is discarded such that the features with the largest dot product to the input are assigned to the incoming event, regardless of the absolute value of the adapted selection threshold. Thus, the unseen test sequences are passed through the same FEAST layer in which the adaptive threshold and the weight update mechanism are disabled. Every trained weight acts as an events filter, allowing only events from specific event contexts that correlate with a specific weight matrix to be passed through and projected to a feature map.

The following layers are the same as the first network pipeline described in Section 4.4.7.1. The main difference is that the features are more robust and balanced in this network. Similarly, after each STP operation, the $3 * 3$ pooled receptive field is flattened to a 1-D vector which contains all the information about the recent event and its neighbours. The process is repeated until the end of the test set is reached, resulting in a matrix of size $9 * N_{neurons} * N_{events}$ where $N_{neurons}$ is the total number of neurons and N_{events} is the total number of events in the test set. A supervisory label is assigned to each flattened vector in the final matrix denoting its category.

The number of firing neurons for each class was monitored during training and during inference to ensure each neuron activated to a particular class. In addition to pooling the information, the spatial coordinate (x, y) of each pixel is recorded, and the timestamp is used later to visualise the output of the classifier networks. Similarly, the training set consisted of random segments from each recording sequence of about 50%,

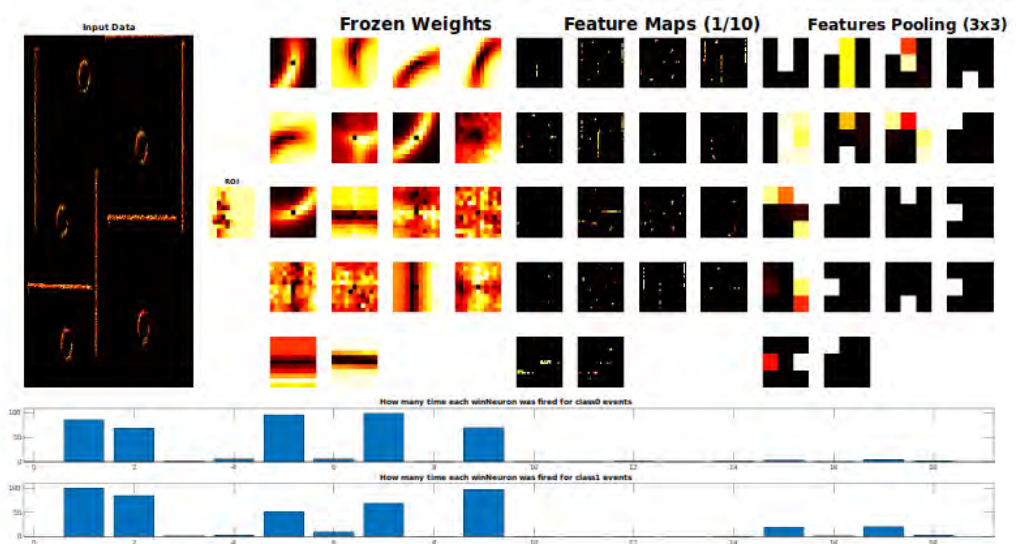


Figure 4.15: Inference with balanced features. Shows the uses of the trained weight during inference to construct a feature map along with the feature pooling layer. The bottom panels show the number of winner neurons for each class.

and the remaining 50% made up the test set.

4.4.8 An Event-based Noise Filter

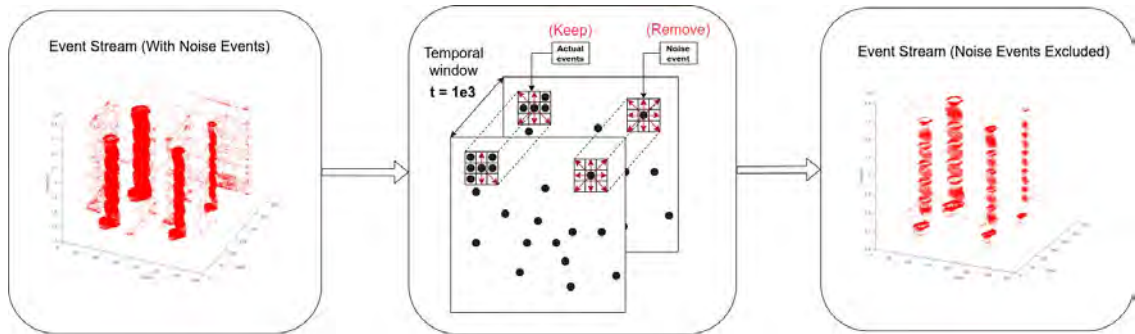


Figure 4.16: Event noise processing showing the spatio-temporal events with and without applying a noise filter.

Noise in the data from the DAVIS sensor typically has two broad characteristics, such as high-frequency burst events and low-frequency random events that occur throughout the scene. There are multiple noise filtering methods for event-based data proposed in the literature such as [Liu et al., 2015, Linares-Barranco et al., 2015, Ieng et al., 2014, Czech and Orchard, 2016]. The task of filtering away the unwanted noise outside the classifier is an important post-processing step that helps extract a clear spatio-temporal pattern of the object. An efficient filter must retain most of the signal and remove most of the noise events presented in the stream.

A high-frequency movement generates most of the events in our data due to the camera's motion, making removing low-frequency events (i.e. noise) a trivial task. In this case, the filter was considered as a low pass filter. Here, the main focus is to

apply a noise filter after the classification as the network's last layer to remove the false positives and extract the moving objects. In this work, a signal was defined as the events captured by the classifier that belong to class 1 (i.e. True Positive).

The noise was considered the events or activities presented outside the True Positive. Note that the EBC not only captures the change in the light intensity at a location due to moving objects but also produces noise events from the scene at various pixels due to the movements of background objects and the sensor's internal noise. A few noise events pass through the classifier network (i.e. miss classified events), producing an increased number of False Positives. The filter has to deal with removing noise events and miss-classified events.

The noise filter has two parameters: receptive field size and temporal window. For each incoming event, the filter looks at the most recent event and its neighbouring pixels and check whether this pixel is noise or an actual signal. For instance, if multiple events occurred within the receptive field within the defined temporal window, then this event is considered an actual signal. Otherwise, it will be removed as a noise event. These parameters depend on the types of events the classifier gives, and a parameter search needs to be performed to find the appropriate parameters.

4.4.9 GLS for Highly Imbalanced Classes

This section proposes a statistical method that considers all the information in the input event stream without applying undersampling or oversampling to overcome the problem of highly imbalanced classes and to address the problem of heteroskedasticity.

Due to the nature of our dataset, which is inherently dense and complex, the class labels become imbalanced and lead to an unequal distribution of data for the train and test dataset. For example, the events for class 0 were significantly more numerous than class1, creating a biased towards the dominant class. In this case, the problem of heteroskedasticity arises, because the variance of the observations was not constant, as shown in Figure 4.17 ³. We addressed this problem by applying a GLS estimator method Aitken [1936] which is equivalent to applying ordinary least squares to a linearly transformed version of the data and takes into account the inequality of variance in the observations.

GLS assumes that we have the following model:

$$Y = X\beta + \varepsilon \quad (4.15)$$

$$E[\varepsilon] = 0 \quad (4.16)$$

³<http://halweb.uc3m.es/esp/Personal/personas/durban/esp/web/notes/gls.pdf>

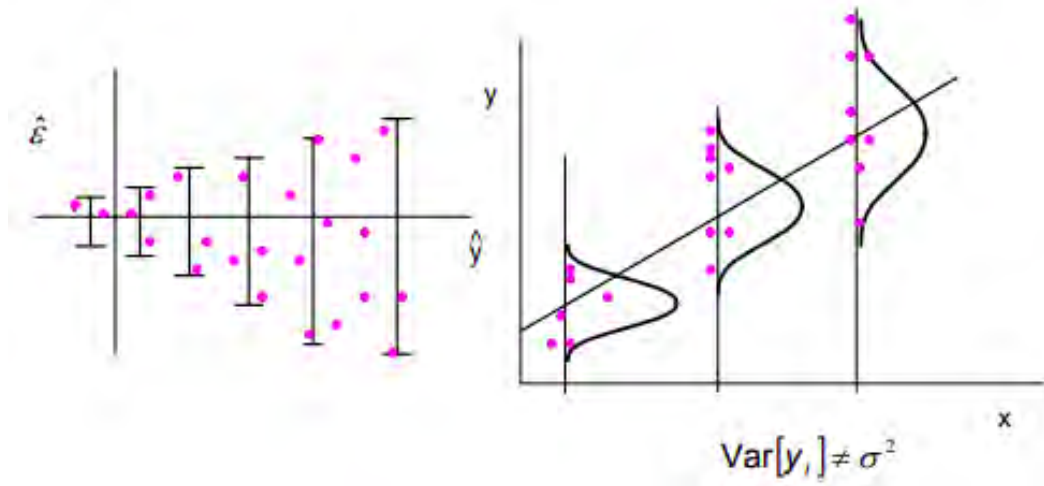


Figure 4.17: Heteroskedasticity. It shows the skewness of the dataset where the variance of the observations is unequal to the mean. Upon visual inspection of the residual errors, the tell-tale sign is that they will tend to fan out over time.

$$Var[\varepsilon] = \sigma^2 \Omega \quad (4.17)$$

Where Ω is a known $n * n$ matrix if Ω is diagonal but with unequal diagonal elements, the observations y are uncorrelated but have unequal variance, while if Ω has non-zero off-diagonal elements, the observations are correlated. The optimal solution is to transform the model to a new set of observations that satisfy the constant variance assumption and use the least square to estimate the parameters. Since $\sigma^2 \Omega$ is a covariance matrix, Ω is a non-singular matrix. The generalised least squares estimator of β is:

$$\beta = (X' \Omega^{-1} X)^{-1} X' \Omega^{-1} y \quad (4.18)$$

The value of Ω dictates how far or how close the class labels are from each other, which also adjust the distribution between the dataset.

4.4.10 Classification Algorithms

In this work, the choice of a classifier plays a crucial role in the performance of the feature extractor. For that reason, two classifiers were used to perform the learning and classification tasks on the feature events generated from the network pipeline. The baseline test was performed using a linear classifier to measure how linearly separable the underlying data is after processing. In addition to this baseline classifier, an ELM was used, which not only has a large number of random hidden layer neurons but also projects the non-linearities of the dataset into a linearly separable higher dimensional feature. In this work, we present the results from both classification algorithms to show

the model performance on different classifiers and show the linearity of the data after feature extraction.

The linear classifier is the first algorithm used to classify data into labels based on a linear combination of input features. It separates data using a line or plane, and it is used to classify data that is linearly separable. The classifier consists of only one layer that uses Online Pseudo-inverse Update Method (OPIUM) van Schaik and Tapson [2015] to iteratively update the linear output weights which project from the input layer to the output neurons. An iterative method of solving the pseudo-inverse for the linear classifier allows the classification network to be updated in response to each event. Because this method classifies each event individually, the bigger the size of the event-based data and the scale of input channels, the more computational resources needed, which make the classification prohibitively challenging, which was the primary motivation to apply the event-driven downsampling as described in Section 4.4.5.

The second classifier is an iterative implementation of the ELM Huang et al.. It consists of a standard three-layer configuration and uses random weights to project from the input layer to a hidden layer. This hidden layer input is passed through a nonlinear activation function, typically a sigmoid function. A set of linear output weights are learned to map the hidden layer output to the output classes, thus performing classification. The ELM network also makes use of the OPIUM to learn the input/output weight mapping.

In this work, both of the classifiers are backpropagation-free as they only require a single feed-forward pass to compute the weight mapping between the input and output and then predict the labels for the test samples. Both classifiers were applied to all test sequences. For instance, the linear classifier was used to investigate the linearity of the data and understand its complexity in high dimensional space. However, the only drawbacks of the linear classifier are that it gives the same outcome at every run because there is no random weight initialisation, and at each feed-forward pass, the same outcome is generated with no further improvement. Therefore, the linear classifier was used as a baseline for the performance of the other classifier. In contrast, the ELM classifier was used with different hidden layer neurons and different random weight initialisation to help to compute the accuracy error over multiple runs.

4.5 Results

In the sections, we evaluated the performance of the event-based detector and classifier algorithm on the events dataset described in Section 4.4.7.1 and Section 4.4.7.2. The architectures were first evaluated on a well-known dataset such as N-MNIST and DVS gestures in Section 4.5.1 and Section 4.5.2, respectively in a view to forming a baseline for the model performance. While we are targeting per-pixel classification in this thesis, it is worth to note that the structure of the N-MNIST and DVS gestures is different from our agricultural data mainly because we deal with multiple objects of the same class per recording, whereas N-MNIST and DVS gestures involve single object per recording. The performance of the algorithm on the lab recorded data was inves-

tigated in detail in Section 4.5.3 and was evaluated on the real-world scene in Section 4.5.4. We investigated the effect of per-pixel downsampling and event-based feature selection in Section 4.5.5 and Section 4.5.6 respectively. We showed how applying a noise filter after the classification can reduce the false positives for the predicted signal and potentially lead to better results. Finally, we proposed a solution for the problem of highly imbalanced data by using GLS method, which was described in Section 4.5.8. All networks in this work were trained with the same parameters such as the number of neurons, learning rate, time constant, STP filter size and the number of hidden neurons in the ELM and train/test cross-validation in a view to providing a valid comparison of the network performance.

4.5.1 N-MNIST Digit Classification Results

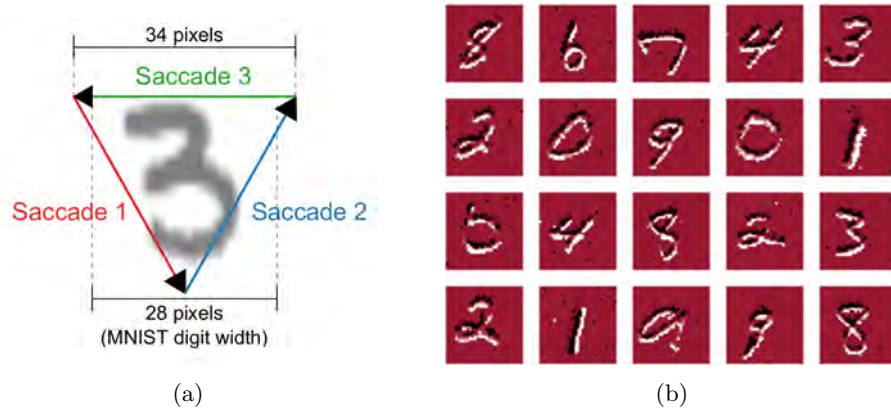


Figure 4.18: N-MNIST Dataset. (a) Showing the three saccade-inspired motions across each digit (Source: Afshar et al. [2019b]). (b) N-MNIST patterns are represented in the time dimension to static images with pixel intensity proportional to the event rate of the pixel.

To prove and demonstrate the feasibility and reliability of the network for classification tasks, we first evaluated the network performance on well-known datasets, which can be considered the baseline performance for the architectures proposed in this thesis.

This section examines several approaches to performing recognition and classification tasks on the N-MNIST dataset. The N-MNIST dataset [Orchard et al., 2015] as shown in Figure 4.18(b)⁴ contains only 10 different classes, the digits 0–9. The digits were recorded using a neuromorphic vision sensor by moving it in a pre-defined saccadic motion in front of a display screen. Through the heuristic examination described in Section 4.4.7, this approach primarily makes use of the same network architecture as a mechanism of learning and classifying spatio-temporal patterns. This section characterises the performance of a linear and ELM classifier using two methods of pooling features from feature maps and investigates the model performance with regularisation.

⁴<http://greg-cohen.com/project/datasets/>

Table 4.3: Parameters used during training and inference.

Parameter	Value
Number of features	120
Time constant	1e4
Learning rate	0.001
ROI	11x11
Threshold open	0.001
Threshold close	0.003
Downsampling factor	No Downsampling
STP filter	3x3
Regularisation	1e-3 to 1e3
ELM hidden neurons	1-1e3

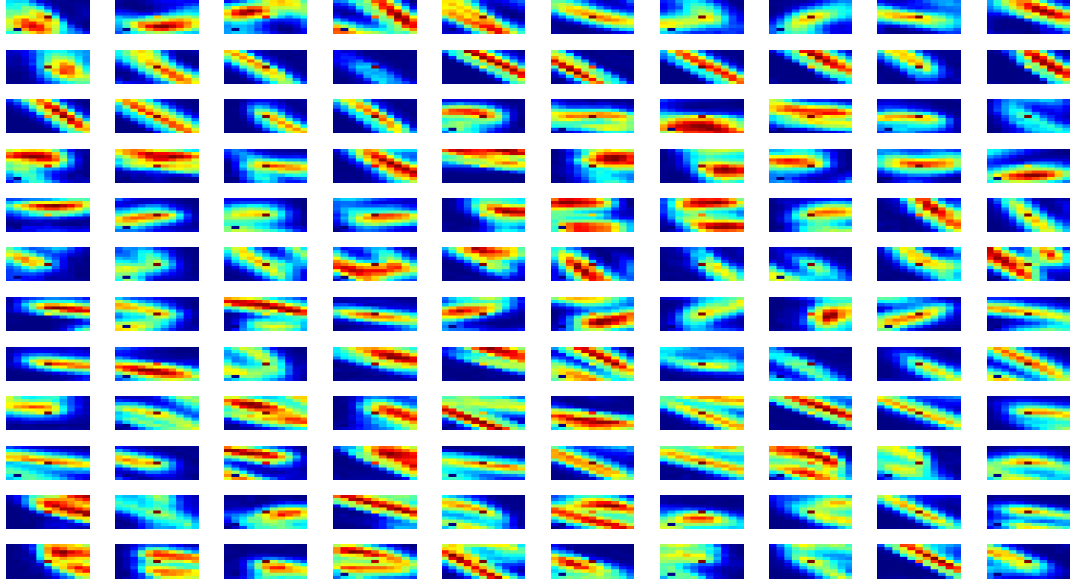
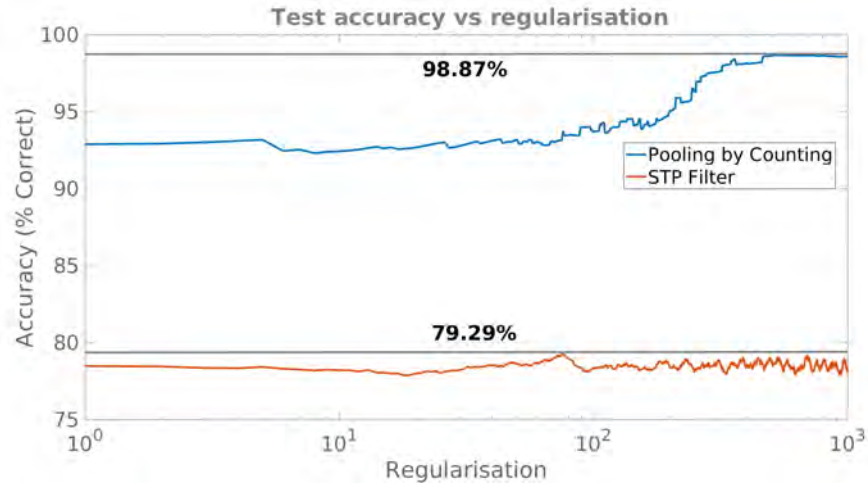


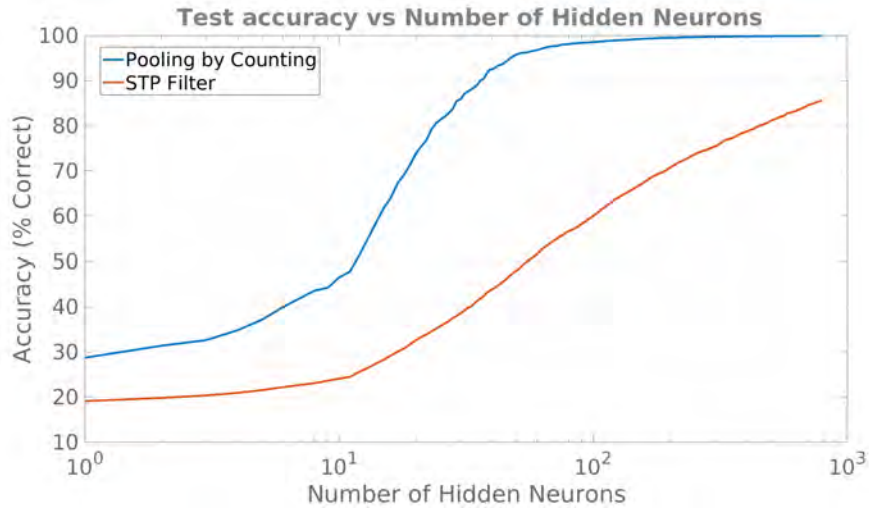
Figure 4.19: Learned weights during FEAST training. 11x11 feature learned from the ON and OFF events of the N-MNIST dataset. Each feature represents a normalised vector reshaped to match the size of the incoming feature patches.

For each digit, 12 feature neurons were selected using both polarities with a size of 11x11. When the training for all digits was completed, the features were aggregated to form 120 features for all the digits similarly to the "mixed weight" architecture in Figure 4.13. Only the training samples were used to generate the features and made use of the feature extraction parameters configured as shown in the Table 4.3. The first pooling layer is the same described in Section 4.4.6 where a spatio-temporal filter of size 3x3 is selected for each incoming event to pool all the features from all features surfaces. For example, for a network containing 120 neurons with a 3x3 STP filter with the image size of 34×34 pixels and the 5500 events (average of the total number of events for digit 0) results in a required input size of 5.94 million events per digit.

The second pooling layer counts the pixel through the whole image (i.e. pooling by counting). It reduces the layer's size to a single feature per time-step, reducing the input layer by more than an order of magnitude. For example, a network with 120 neurons will result in an input pattern size of 660000. Both polarities were processed and pooled from the feature map and then fed to a linear and an ELM classifier. Given the computational resources needed to classify all test digits in an event-driven fashion, only a portion of the test was selected to reduce the computation cost and time. In this case, 250 event files were randomly chosen for each digit producing a large matrix of size $250 \times \text{Nevents} \times \text{Nneurons}$ for the pooling by counting and $250 \times \text{Nevents} \times \text{Nneurons} \times 3 \times 3$ for the STP filter. After applying cross-validation and shuffling, this matrix is then fed to the classifier to randomise the order.



(a)



(b)

Figure 4.20: Comparison of the classification test accuracy. (a) Test accuracy with regularisation. (b) Test accuracy with different hidden layer size.

Figure 4.20(a) presents the model accuracy using both pooling operations using the ELM classifier. The results show that the pooling by counting operation outperforms

the STP filter. The pooling by counting operation achieved 98.87% in accuracy when the regularisation factor was 478, while the STP filter achieved 79.29% and then it started to reduce. The results in Figure 4.20(b) show that the model accuracy gradually increases with the size of the hidden layer. Similarly, the pooling by counting operation shows superior performance and achieved higher accuracy with a smaller number of hidden neurons in comparison with the STP filter, which seems that it requires more hidden neurons to achieve higher accuracy.

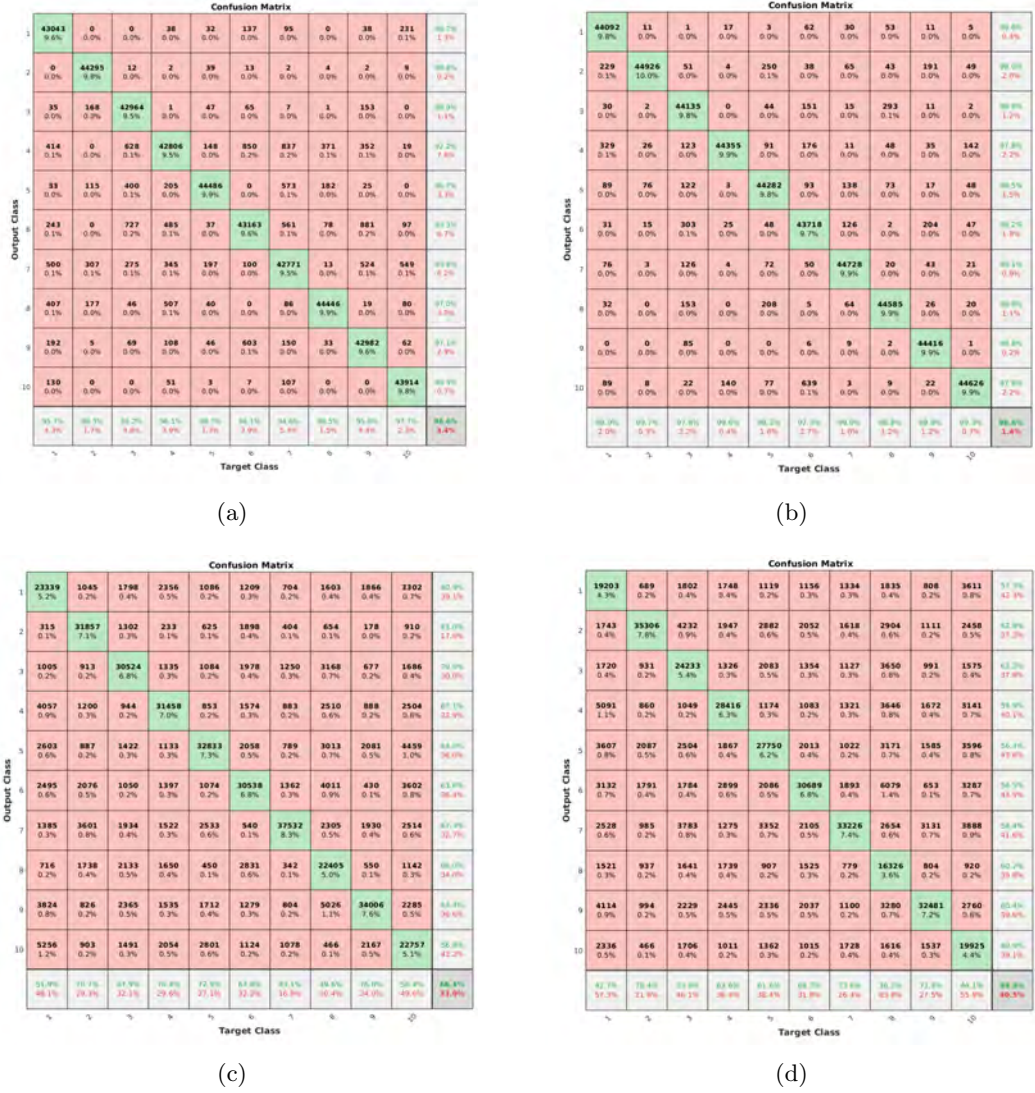


Figure 4.21: Confusion matrices for the 10-category classification problem. (a) and (b) Showing the results using the pooling by counting operation using the linear and ELM classifier, respectively. (c) and (d) Showing the results using the STP filter operation on the linear and ELM classifier, respectively.

These results show both the importance of regularising and increasing the classifier's hidden neurons. Also, it shows that providing a single value per time-step (i.e. pooling by counting) can provide sufficient information for the classifier. In contrast, the STP filter operation shows that the data became non-linearly separable and made the input attribute for the classifier much larger, which affect the computational and

memory limits. Thus, counting the values through the whole image is a well-suited operation to the nature of the events produced by FEAST, as they are inherently sparse spatio-temporal patterns. Figure 4.21 shows the confusion matrices for both pooling operations and classification networks. For comparison purposes, the pooling by counting also had a higher accuracy across all digits for both the linear and ELM classifier as high as 99.8% for digit one and 99.3% for digit 92.2% for digit three the linear classifier, which increased to 98.8% using an ELM classifier. In contrast, using the STP filter show much lower performance in the range of 56.8% and 83.0% for digit 9 and 1, respectively, and the linear classifier showed to have superior performance than the ELM classifier.

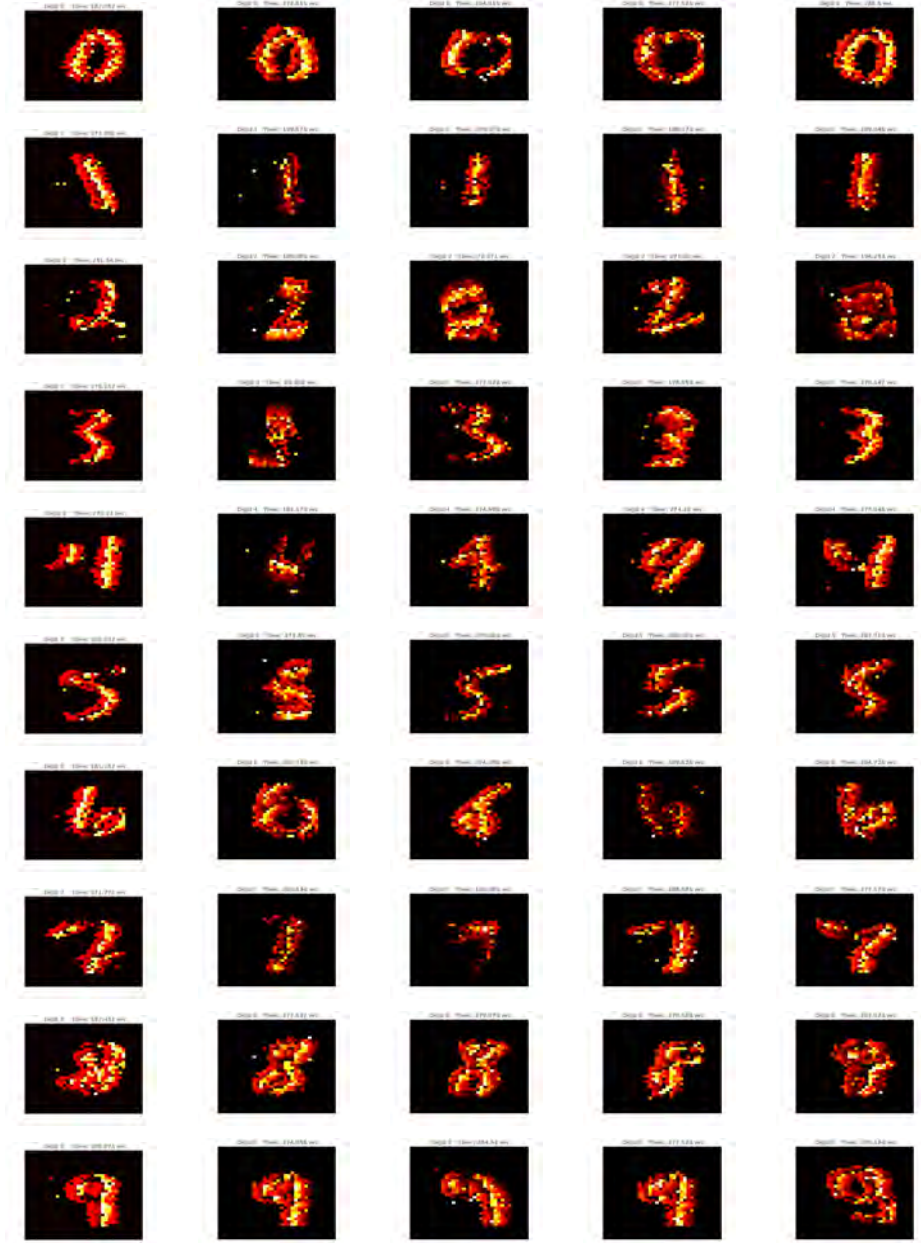


Figure 4.22: Correctly classified events for each digit using the Pooling by Counting operation. The digits were randomly selected for visualisation purposes. The digits patterns are collapsed in the time dimension to static images with pixel intensity proportional to the spike time of the pixel using exponential time surface.

In Figure 4.22 and Figure 4.23 the results of the classification output for each pooling operation are projected into a time surface to show the event patterns for each digits. Given the higher accuracy of the pooling by counting in terms of hidden layer size and regularisation, this pooling operation demonstrates better and clearer spatio-temporal patterns for each digit in comparison with the STP filter operation, which was spatially sparse.

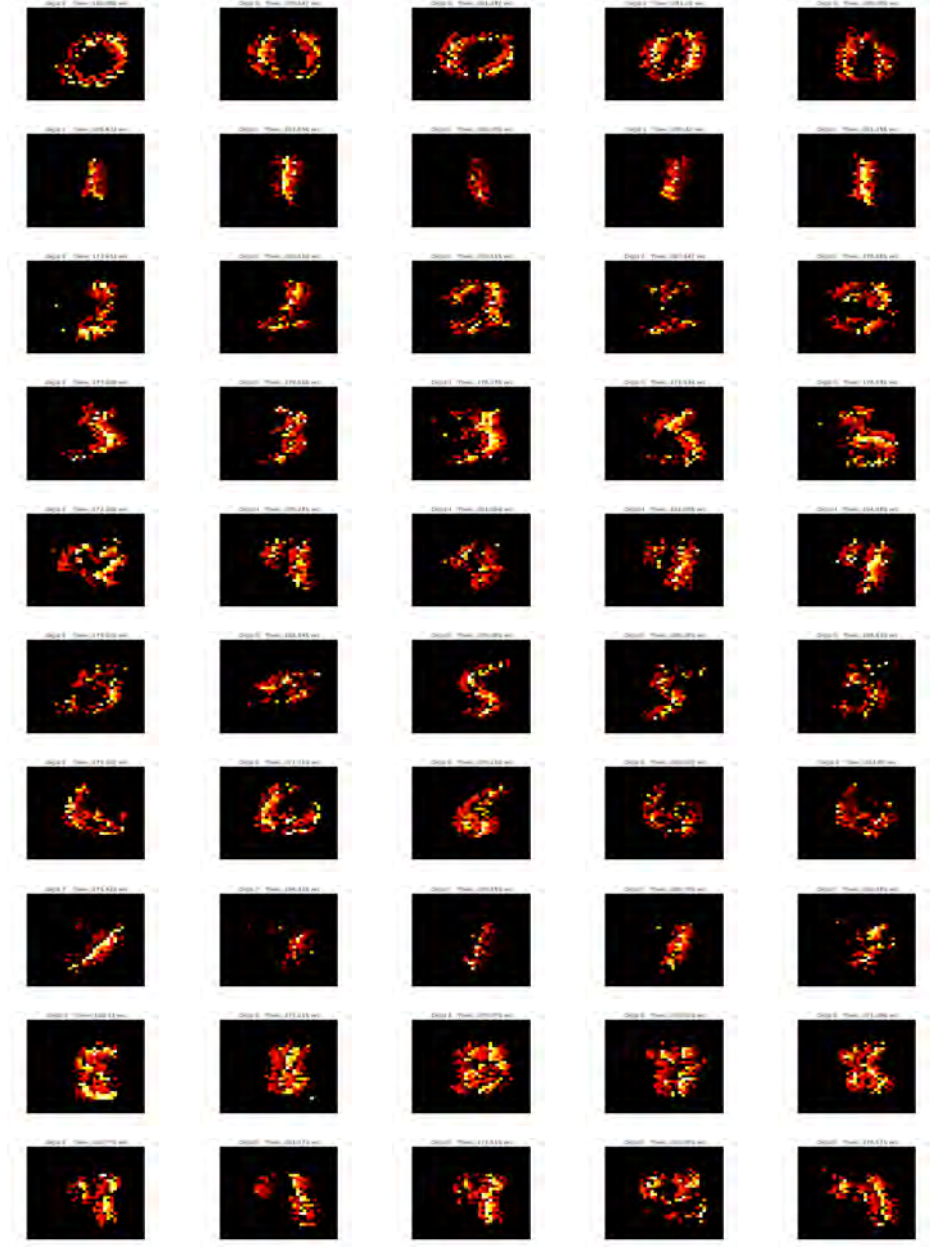


Figure 4.23: Correctly classified events for each digit using the STP filter operation. The digits were randomly selected for visualisation purposes. The digits patterns are collapsed in the time dimension to static images with pixel intensity proportional to the spike time of the pixel using exponential time surface.

4.5.2 DVS Gesture Dataset Results

To test our network architecture on more complex datasets and instead of creating new datasets from scratch, we utilised the DVS gestures datasets from IBM [Amir et al., 2017], which were recorded using a DVS128 sensors. This data presents several fundamental properties, such as being recorded using natural motion rather than simulating movements used in the generation of DVS-converted datasets like the N-MNIST and Caltech101 [Orchard et al., 2015]. This dataset has 11 distinct human gestures executed by one subject in each trial, providing 1342 samples divided into 122 trials. Each

gesture has an average duration of 6 seconds combining all the polarity events recorded by the sensor. Three lighting conditions, including LED light, natural light, and fluorescent light, are selected to control the effects of the flicker on the DVS camera and the background shadow, providing a bias improvement for the data distribution. In this case, spatial and temporal information are essential components.

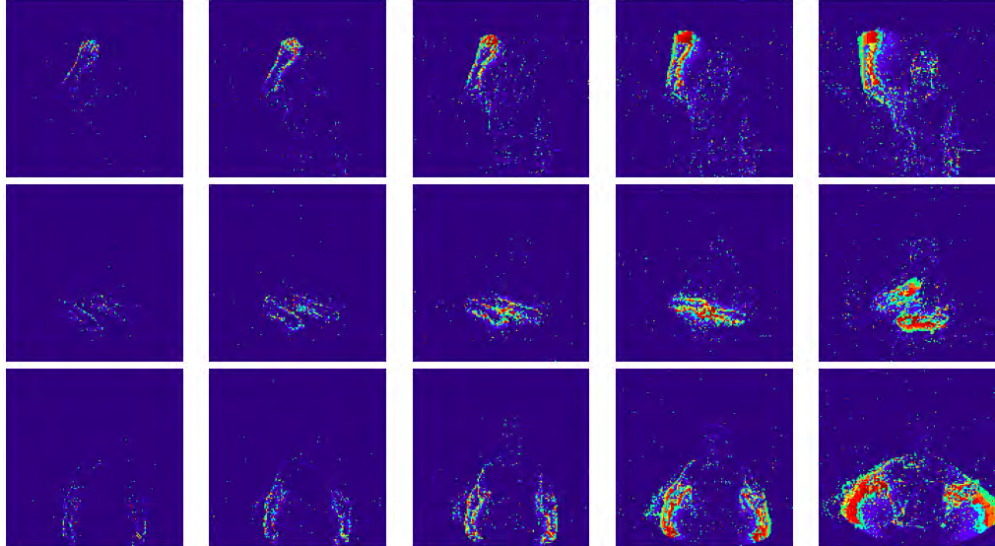


Figure 4.24: DVS Gesture dataset. Three different gestures are shown where each frame represents all the events at different time interval. *Right Hand Clockwise* (top); *Arm Roll* (middle); *Other Gesture* (bottom). Pixel are colour-coded according to the pixel time.

For DVS gesture dataset, we used the "standard weight" and "mixed weights" architectures described in Section 4.4.7.1 and Section 4.4.7.2 to investigate the effect of using the supervisory signals during the initial features training. The method of processing the DVS gesture dataset was the same as that used for the N-MNIST dataset. However, we only used the pooling by counting operation as it is less computationally expensive compared with the STP filter, which requires rigorous optimisation for pseudo-inverse weights multiplication (i.e. OPIUM) and GPU support due to the immense data size. The investigation was based on the ROI size that ranged from 7×7 to 31×31 , the number of features, and the downsampling σ size using both linear and ELM classifier networks. As with the N-MNIST dataset, the system was trained on a subset of the gesture dataset. The training set consisted of 98 sequences of human gestures, with the remaining 23 making up the test set. The gestures' spatio-temporal pattern significantly varies for each recording due to the change of the gesture velocity, pose, and the activity's periods. An example of the features generated for the gesture for different ROI size is presented in Figure 4.25 which shows the resulting features set learned at five trials on a randomly selected train set with different ROI sizes. The trained feature shows the consistency in learning discriminative features with some features coding the human hand from a different point of view and the variation in noise events. One of the main advantages of the feature extraction network is the dynamic learning of the noise variants in the events, which can vary with the sensor bias, removing the need to

hard-code a noise filter. The features that learn the noise can filter out these events for the rest of the network. Once the training is completed, the trained neurons (i.e. features) are used to project the latent features back to the original event pixel space (i.e. feature space) to remap the learned features over to the shape generation process to unravel the latent space the classifier. Here we show the events captured by each winner neuron during inference at different network sizes. At inference, each neuron act as a spatio-temporal filter. For example, if a particular neuron learned the shape of the human hand, then at inference, this neuron will only allow events from the human hand to be passed through to the feature space, and the same applies to noise events.

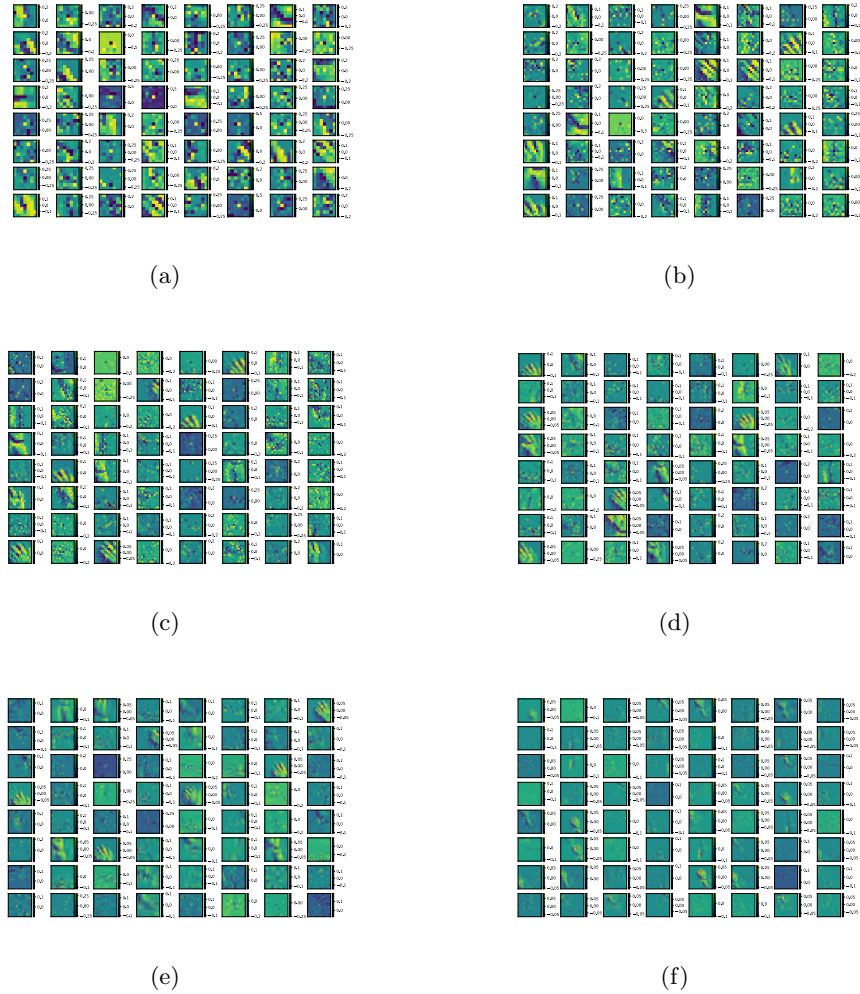


Figure 4.25: Features generation at multiple network scales showing the network consistency in learning features. Panels (a) are features generated using an ROI of size 7×7 . Panels (b) are features generated using an ROI of size 11×11 . Panels (c) are features generated using an ROI of size 15×15 . Panels (d) are features generated using an ROI of size 19×19 . Panels (e) are features generated using an ROI of size 23×23 . Panels (f) are features generated using an ROI of size 31×31 .

As shown in Figure 4.26 the magnitude of the change in the features thresholds and weights are similarly low at the beginning of the training due to the random initialisation of the weights and thresholds as well as the early unbalanced activity of

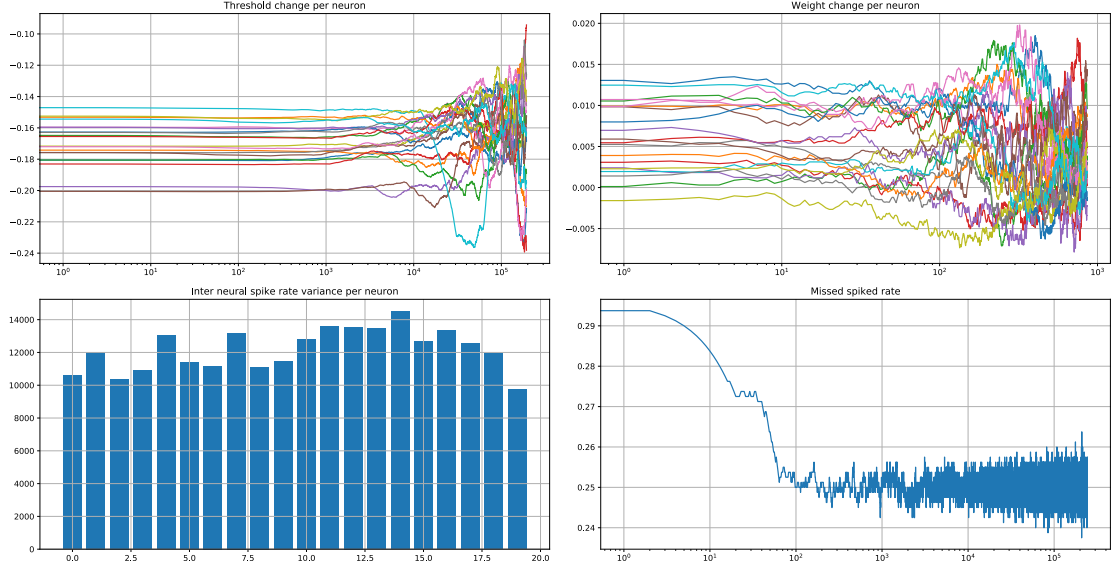


Figure 4.26: Evolution of the FEAST neural signals and parameters during training using 20 neurons. Top panels show the network’s adaptation of various neural signals, such as the change in the threshold (Left) and the neuron weight (Right) over three trials and a randomly selected test set. The bottom panels show the interneural spike rate variance (Left), which demonstrate the neuronal homeostasis showing how the network maintained a target level of spike activity and the missing spike rate over time (Right).

the network. As the learning progressed, the neurons became more selective and active, which increased/decreased the rate of change in the neuron weight and threshold. Besides, the missing events rate is reduced once the rate of the change increases, indicating that fewer events are being filtered out over time, and the majority of the events are being processed. The network has also maintained a homeostasis state between all the neurons resulting in a relatively equal firing rate per feature as shown in Figure 4.26(Bottom left panel). That means that the firing rates remain relatively constant, ensuring that the network treats each neuron equally and dynamically adjust the synaptic strengths in the correct direction to promote stability.

In Figure 4.27, the trained weights are used during inference using randomly selected test sets to convert the events to feature space. The network demonstrated that it is possible to segregate discriminative events even at different scales. However, the number of output events slightly increase as the ROI increase which was expected because a larger ROI contain more information about the scenes. At inference, the downsampling factor σ was varied between 1 and 10. Two classifiers were used, a linear classifier and an ELM classifier with 100 hidden neurons, to compare the network performance. The process was repeated for "standard weights" and the "mixed weights" architecture. For evaluation, the accuracy measure was used for both classifiers. These networks results are shown in Table 4.4 and Table 4.5.

As the results in Table 4.4, the highest classification accuracy is achieved using a 15x15 ROI, 50 FEAST neurons, without downsampling ($\sigma=1$) using a linear classifier, resulting in 92.95%. Overall, we observed the following: (i) the highest accuracy was

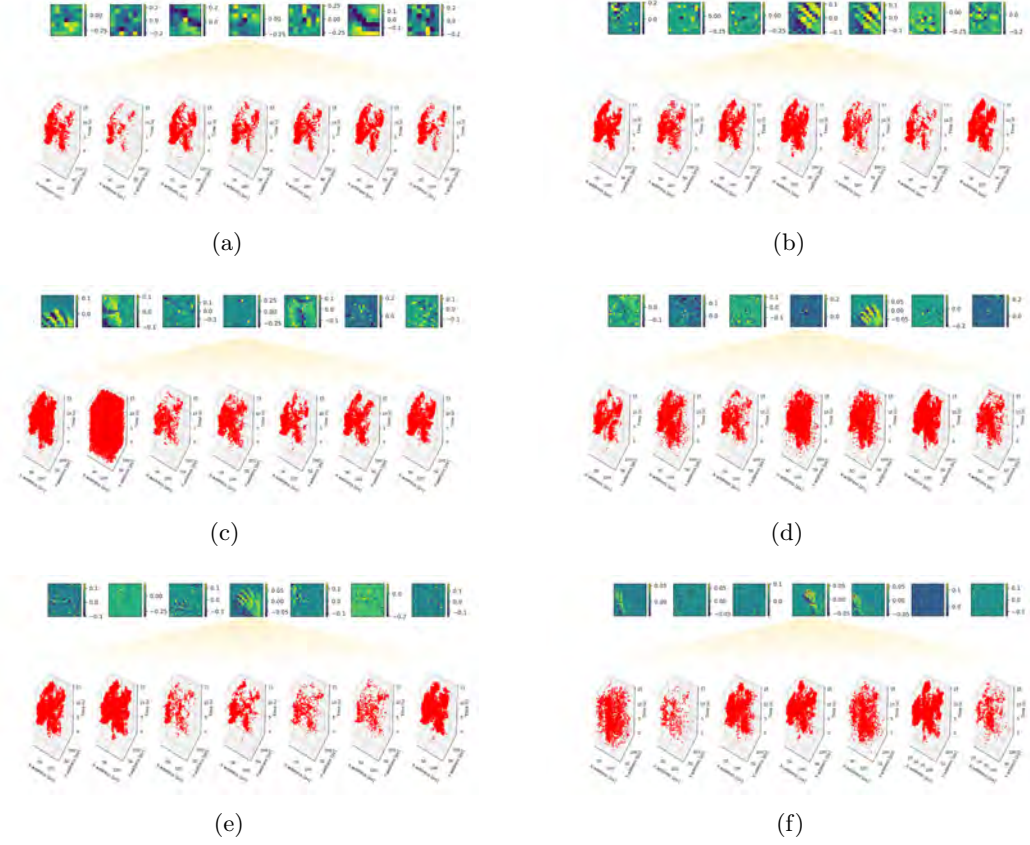


Figure 4.27: Filtering mechanism at inference. The spike events output generated by each winner neuron shows that each neuron selects events based on the learned feature. Only a subset of neurons is shown here for visualisation purposes. Panels (a) are features generated using an ROI of size 7x7. Panels (b) are features generated using an ROI of size 11x11. Panels (c) are features generated using an ROI of size 15x15. Panels (d) are features generated using an ROI of size 19x19. Panels (e) are features generated using an ROI of size 23x23. Panels (f) are features generated using an ROI of size 31x31.

achieved without downsampling for most of the cases, (ii) having more features led to better classification performance, (iii) in most cases, the linear classifier outperformed the ELM network with a tiny margin, (iv) the network performance showed to be consistent at different ROI sizes and (v) the ELM had lower classification error with larger ROI size over 20 trials indicating a consistent performance. Table 4.5 show the results using the "mixed weights" architecture. The highest accuracy was achieved using a 31x31 ROI, 50 FEAST neurons, without downsampling ($\sigma=1$) using a linear classifier, resulting in 94.51%. The aggregation of the features during inference increased the classifier's performance by an additional 1.56% slightly. Almost all results were higher using the linear classifier than the ELM classifier showing that FEAST managed to provide enough information to the linear classifier even without the use of downsampling. Overall, the results from the "standard weights" and the "mixed weights" architectures were not significantly different. That was due to the high degree of similarity between the features where the human arms and torso were dominant in the visual field,

making the neurons learn wide variations of the same subject. For that reason, the output activation of FEAST neurons has already provided a linearly separable mapping to the output classes for both architectures. However, the feature similarity and the high variance in the activity velocity and the recording condition (e.g. lighting condition, background, etc.) impacted the algorithm’s perfect classification performance. Figure 4.28 shows the correctly classified events for each activity using the ”*user28_natural*” recording and Figure 4.29 shows the confusion matrices using the model architecture that had the highest performance. Computational and memory limits prevented further experiments.

Table 4.4: Comparison of classifier performance on DVS gestures with different hyper-parameters using ”standard weights” architecture as described in section 4.4.7. Twenty trials of each network configuration were performed using the ELM classifier consisting of 100 neurons. σ is the downsampling factor. LC is the linear classifier.

ROI	Features	σ	Test Accuracy LC (% Correct)	Test Accuracy ELM (% Correct)
7x7	20	1	91.67	91.26 ± 1.24
		10	86.46	81.77 ± 2.56
	50	1	92.39	90.68 ± 2.41
		10	87.35	81.24 ± 3.12
11x11	20	1	82.47	86.98 ± 2.87
		10	86.42	78.36 ± 2.34
	50	1	91.82	88.83 ± 2.15
		10	88.00	83.84 ± 4.23
15x15	20	1	88.22	84.08 ± 1.26
		10	78.98	77.58 ± 3.25
	50	1	92.95	87.55 ± 2.56
		10	90.17	82.48 ± 1.19
19x19	20	1	89.85	89.75 ± 1.69
		10	83.00	79.97 ± 3.58
	50	1	92.09	87.99 ± 2.59
		10	90.66	83.22 ± 2.58
23x23	20	1	88.05	88.47 ± 1.44
		10	84.53	81.49 ± 1.67
	50	1	90.37	90.23 ± 0.98
		10	86.56	84.48 ± 1.63
31x31	20	1	86.81	86.92 ± 1.35
		10	85.54	81.16 ± 2.33
	50	1	92.54	88.73 ± 1.31
		10	92.91	83.31 ± 1.75

Table 4.5: Comparison of classifier performance on DVS gestures with different hyper-parameters using "mixed weights" architecture as described in section 4.4.7. Twenty trials of each network configuration were performed using the ELM classifier consisting of 100 neurons. σ is the downsampling factor.

ROI	Features	σ	Test Accuracy LC (% Correct)	Test Accuracy ELM (% Correct)
7x7	20	1	90.27	89.56 ± 1.54
		10	89.72	83.50 ± 2.00
	50	1	91.96	90.04 ± 0.97
		10	92.67	85.61 ± 1.62
11x11	20	1	90.07	91.27 ± 0.92
		10	88.38	83.31 ± 1.89
	50	1	91.99	90.02 ± 1.22
		10	91.43	85.40 ± 1.66
15x15	20	1	88.10	89.01 ± 0.98
		10	83.73	82.04 ± 1.77
	50	1	92.54	90.87 ± 0.91
		10	90.74	85.50 ± 1.35
19x19	20	1	90.27	88.84 ± 1.39
		10	88.61	83.06 ± 2.07
	50	1	93.60	91.17 ± 0.96
		10	92.44	86.27 ± 1.55
23x23	20	1	87.24	90.53 ± 1.22
		10	83.05	82.09 ± 1.55
	50	1	93.42	90.83 ± 0.83
		10	91.54	85.45 ± 1.93
31x31	20	1	91.82	90.88 ± 1.30
		10	86.93	81.62 ± 2.58
	50	1	94.51	92.34 ± 0.77
		10	93.32	83.53 ± 3.09

4.5.3 Performance on Laboratory Recorded Data

The lab recorded datasets were primarily recorded to simulate each real-world challenge individually to understand the network behaviour in specific scenarios more efficiently. We conducted several experiments to test the robustness of the feature extraction algorithm against high dimensional noisy events and test the classification network using balanced and imbalanced data. Below are the experiments conducted on each test sequence:

1. Linear classifier using standard network approach with imbalanced data
2. Linear classifier using standard network approach with balanced data
3. Linear classifier using dedicated weights network approach with imbalanced data

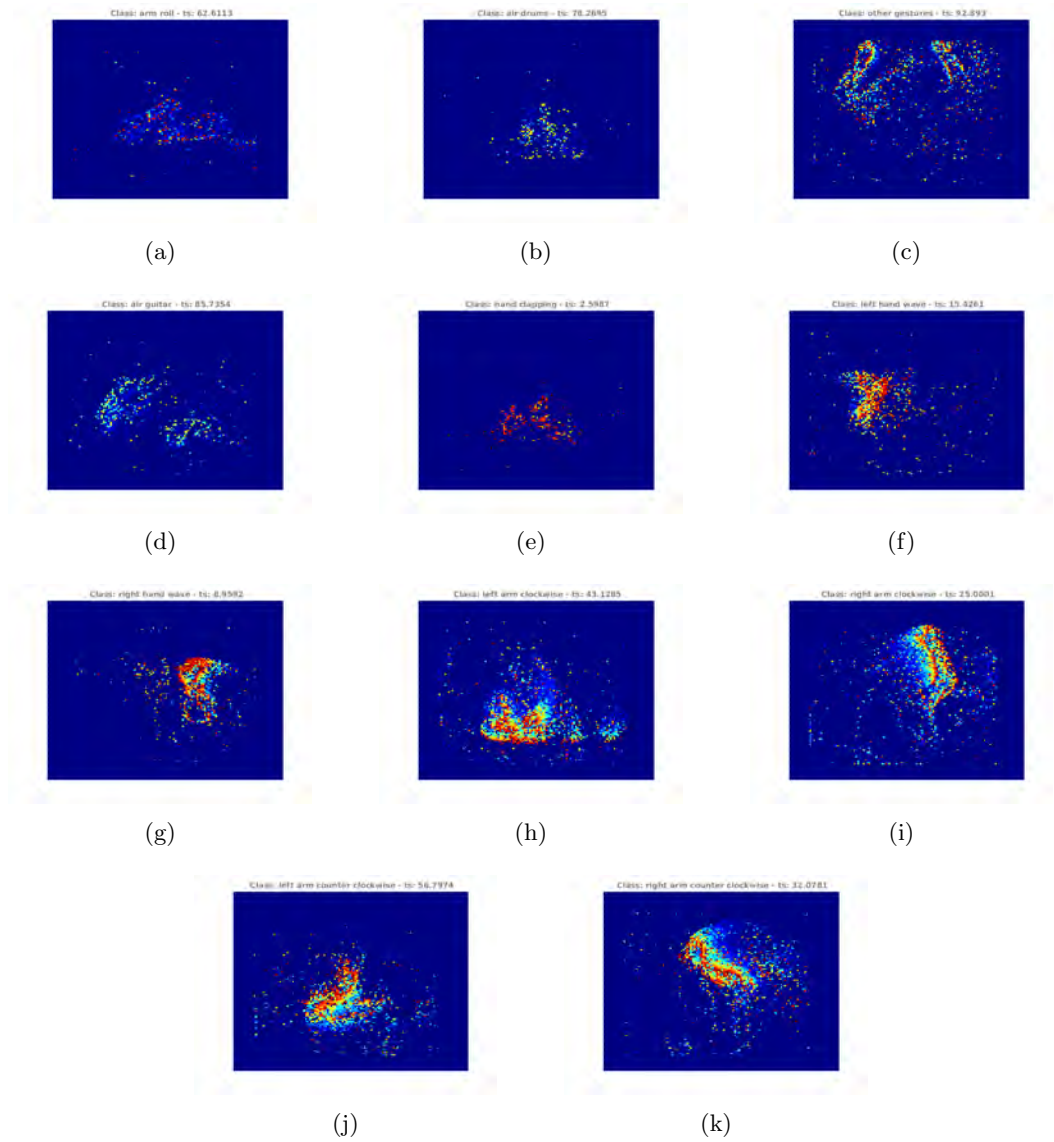


Figure 4.28: Correctly classified events for each activity using *user28_natural* recording. The activity patterns are collapsed in the time dimension to static image representation with pixel intensity proportional to the spike time of the pixel using the time surface. (a) Arm roll. (b) Air drum. (c) Other gestures. (d) Air guitar. (e) Hand clapping. (f) Left-hand wave. (g) Right-hand wave. (h) Left-arm clockwise. (i) Right arm clockwise. (j) Left-arm counter clockwise. (k) Right arm counterclockwise.

4. Linear classifier using dedicated weights network approach with balanced data
5. Linear classifier using mixed weights network approach with imbalanced data
6. Linear classifier using mixed weights network approach with balanced data
7. ELM classifier using standard network approach with imbalanced data
8. ELM classifier using standard network approach with balanced data
9. ELM classifier using dedicated weights network approach with imbalanced data
10. ELM classifier using dedicated weights network approach with balanced data

Confusion Matrix												
Output Class	1	2	3	4	5	6	7	8	9	10	11	12
	17411 0.9%	153 0.0%	2894 0.2%	60 0.0%	2709 0.1%	755 0.0%	1974 0.1%	15563 0.8%	4014 0.2%	2360 0.1%	2642 0.1%	4789 0.3%
	21 0.0%	311 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	93.7% 6.3%
	1077 0.1%	155 0.0%	24380 1.3%	3167 0.2%	50 0.0%	3 0.0%	974 0.1%	0 0.0%	232 0.0%	743 0.0%	2628 0.1%	796 0.0%
	4463 0.2%	2904 0.2%	22070 1.2%	60277 3.2%	857 0.0%	743 0.0%	544 0.0%	2231 0.1%	1859 0.1%	2107 0.1%	3721 0.2%	3837 0.2%
	53126 2.8%	9058 0.5%	22566 1.2%	12209 0.6%	225952 12.0%	16437 0.9%	11130 0.6%	10992 0.6%	43050 2.3%	4994 0.3%	7342 0.4%	14597 0.8%
	33282 1.8%	5295 0.3%	14489 0.8%	10844 0.6%	18766 1.0%	213301 11.3%	5715 0.3%	13464 0.7%	16630 0.9%	5840 0.3%	7350 0.4%	9139 0.5%
	6623 0.4%	82 0.0%	251 0.0%	12898 0.7%	17905 0.9%	2477 0.1%	101085 5.4%	12179 0.6%	11258 0.6%	11469 0.6%	2272 0.1%	7852 0.4%
	15816 0.8%	1565 0.1%	8650 0.5%	7446 0.4%	3642 0.2%	4288 0.2%	2971 0.2%	73054 3.9%	3638 0.2%	918 0.0%	2408 0.1%	6918 0.4%
	42603 2.3%	1328 0.1%	654 0.0%	1674 0.1%	13651 0.7%	23871 1.3%	45544 2.4%	27371 1.4%	199018 10.5%	56034 3.0%	8703 0.5%	21345 1.1%
	1678 0.1%	82 0.0%	0 0.0%	61 0.0%	0 0.0%	0 0.0%	137 0.0%	78 0.0%	909 0.0%	5728 0.3%	2603 0.1%	0 0.0%
	6704 0.4%	11619 0.6%	3419 0.2%	3523 0.2%	0 0.0%	0 0.0%	1886 0.1%	343 0.0%	3940 0.2%	10450 0.6%	43324 2.3%	0 0.0%
	6457 0.3%	251 0.0%	45 0.0%	34 0.0%	239 0.0%	1372 0.1%	0 0.0%	3694 0.2%	11079 0.6%	1217 0.1%	139 0.0%	27587 1.5%
	9.2% 90.8%	0.9% 99.1%	24.5% 75.5%	53.7% 46.3%	79.6% 20.4%	81.0% 19.0%	58.8% 41.2%	46.0% 54.0%	67.3% 32.7%	5.6% 94.4%	52.1% 47.9%	28.5% 71.5%
Target Class												

Figure 4.29: Results and confusion matrices using the network with a linear classifier and an 11x11 ROI and 20 features using the "standard weights" architecture.

11. ELM classifier using mixed weights network approach with imbalanced data
12. ELM classifier using mixed weights network approach with balanced data

In Figure 4.30, each row represents the classification output for each test sequence across different timestamps. Events were represented using a time surface with an exponential decay kernel, and red events belong to class1, and black events belong to class0. Due to the skewness and imbalanced nature of the data, the accuracy metric was not used to evaluate the network's performance. Instead, sensitivity, specificity, informedness and correlation coefficient were used for the final model evaluation. This section shows the model's performance in detail using the balanced and imbalanced data to show (i) the effect of highly imbalanced data vs perfectly balanced classes and (ii) form a baseline comparison for each network architecture. Given that all datasets were recorded with the same movement patterns and translation speed, the same time constant $\tau = 0.5$ was applied in data labelling, feature extraction, and inference. The time constant value was chosen by inspection during the data labelling process to ensure the noise events fade quickly without dominating the signal from the true objects.

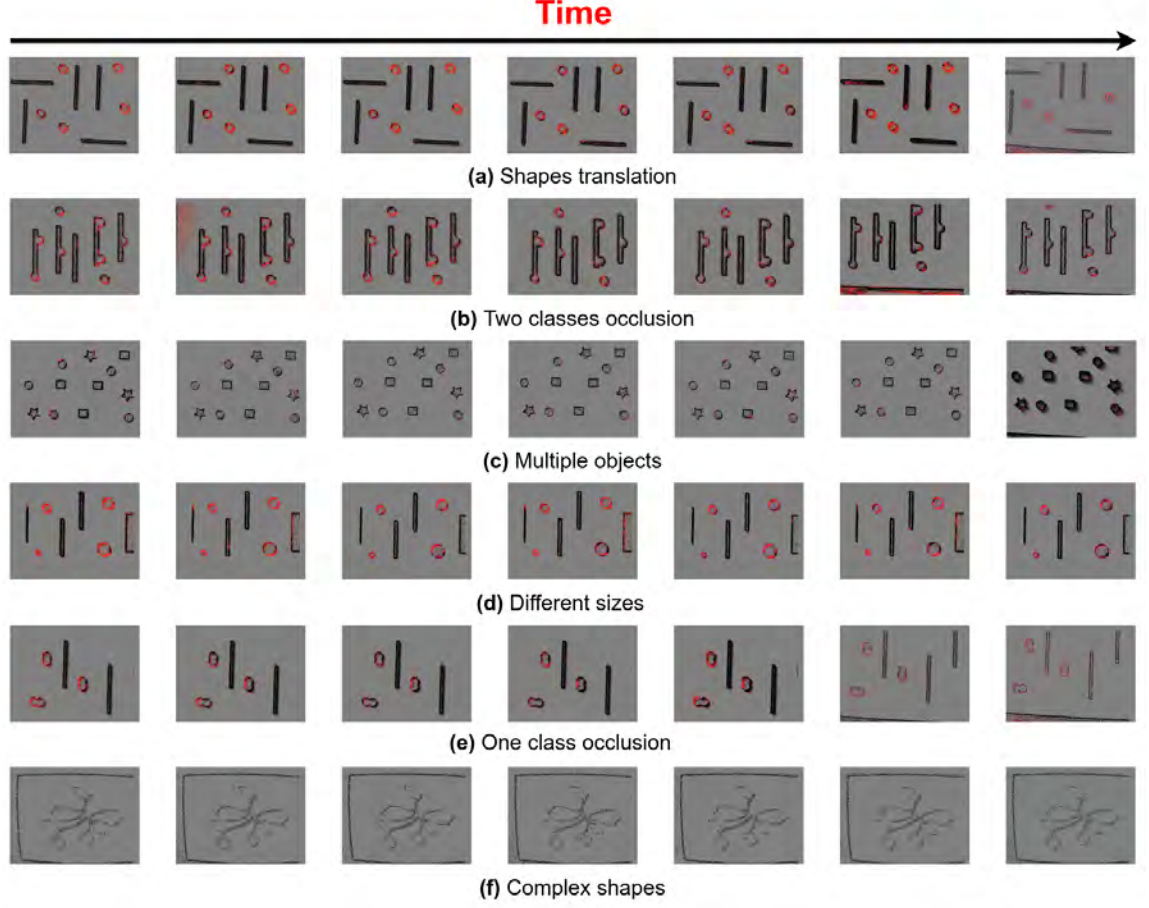


Figure 4.30: Performance of the network on the laboratory recorded dataset from the best performing architecture. Correctly classified instances as red dots, black dots are other classes. (a), (b), (c), (d), (e), (f) show the spatio-temporal patterns of the classification output for each test sequences. Original datasets are described in Section 4.4.1.

Each test sequence in Figure 4.30 presents a unique challenge to the classifier network, such as occlusion, variation in size and complex shape. In "Shape translation", the events were well segregated after the classification, showing a clear stream of events for each class. However, due to the uneven motion of the platform and the presence of noise events, there were few noticeable miss-classified events in each class. As shown in Table 4.6 and Table 4.7, the model achieved the highest informedness of 82.35% on the imbalanced data and 88.24% using the imbalanced data, both using the "mixed weights" architecture with an ELM classifier with 100 neurons indicating the significance of the mixed weights method and the effectiveness of the ELM classifier network.

In "Two classes occlusion", it was evident that the border between both classes became no longer visible due to the overlap, changing the circles' appearance. For that reason, the classification task became more challenging, resulting in a high rate of miss-classified events (i.e. false positive). The model achieved high informedness using the mixed weights method of 86.32% with the imbalanced data using the "mixed weights" architecture, followed by 69.42% using the "dedicated weights" method with the bal-

Table 4.6: Summary of classification performance for all laboratory test sequences using three network architecture and two types of the classifier on the imbalanced dataset. LC refers to the linear classifier, and ELM refers to the extreme learning machine classifier.

		True Positive (%)		False Positive (%)		Sensitivity (%)		Specificity (%)		Informedness (%)	
		LC	ELM	LC	ELM	LC	ELM	LC	ELM	LC	ELM
Shape translation	Standard Weights	65.6	72.5	30.5	5.3	80.76	92.90	96.46	96.17	77.25	77.03
	Dedicated Weights	15.3	15.6	25.3	24.5	10.1	17.85	99.66	98.58	15.5	16.44
	Mixed Weights	55.6	63.2	24.5	35.4	72.60	85.00	98.08	97.34	70.68	82.35
Two classes occlusion	Standard Weights	23.5	15.5	35.8	25.6	38.82	37.75	90.60	90.44	29.42	28.19
	Dedicated Weights	30.5	15.7	27.5	26.5	19.34	41.30	97.78	95.26	17.12	36.56
	Mixed Weights	10.2	25.6	50.5	45.2	28.55	27.16	86.32	85.69	86.32	12.85
Multiple objects	Standard Weights	45.5	34.9	36.5	23.2	13.05	8.37	98.12	98.37	11.17	6.74
	Dedicated Weights	32.5	28.5	52.5	45.2	53.36	52.37	93.75	93.22	47.11	45.60
	Mixed Weights	10.2	31.2	23.5	31.8	25.37	19.43	96.49	96.36	21.86	15.79
Different sizes	Standard Weights	45.3	35.6	32.5	47.5	64.69	62.49	93.64	93.70	58.33	56.19
	Dedicated Weights	21.2	15.3	10.2	10.9	8.26	27.37	98.99	97.29	7.25	24.66
	Mixed Weights	50.3	45.5	32.5	45.5	66.54	62.25	93.41	93.39	59.96	55.64
One class occlusion	Standard Weights	40.2	45.5	32.5	15.5	46.25	47.52	95.02	94.23	41.27	41.75
	Dedicated Weights	10.7	5.7	15.6	14.5	4.55	17.03	99.74	99.03	4.30	16.07
	Mixed Weights	25.5	28.5	26.3	30.5	54.76	37.90	95.35	92.92	50.11	30.81
Complex shapes	Standard Weights	20.5	15.3	12.5	12.9	28.15	28.17	99.80	99.71	27.98	27.81
	Dedicated Weights	20.6	15.8	18.5	24.5	13.85	23.42	99.99	99.79	13.79	23.22
	Mixed Weights	50.6	65.5	69.5	15.5	65.21	73.52	97.46	96.83	62.66	70.33

anced data, both using a linear classifier which was well above the chance (e.g. 50% in this case). This was due to two main reasons: (i) the network has to account for the lines edges, circles curves and the edges between the overlapped objects, which diversifies the learned features and adds another dimension of complexity to the network and (ii) labelling the overlapped objects was challenging as it became harder to separate the events between two overlapped objects during labelling. Since it is a binary classification problem, a particular feature has to be classified as either line or circle, which made the network struggle to classify the overlapped surface. For that reason, fewer events were mislabelled, resulting in a bias in the data toward the dominant class. In the "Multiple objects" recording, there were more objects with different shapes in a view to test the robustness of the network. In this scenario, the network has to make more discriminative representations about each object to differentiate between the classes. The model achieved 45.6% informedness on the imbalanced data using the "dedicated weights" method with the ELM classifier, and a slight improvement to 47.58% using the balanced data with the "mixed weights" method ELM classifier. There were two classes with different sizes in the "Different size" recording to test the model robustness against a different object with different scales. The highest informedness was achieved using the mixed weights method with 59.96% using the "mixed weights" method and a linear classifier, and a noticeable increase in the model informedness with 74.46% on the imbalanced data using the same architecture and classification algorithm. This shows that in cases where the objects have different sizes, it is beneficial to have dedicated weights for each class to help the model better generalise the object of interest and be size invariant. Another type of occlusion was introduced in the "one class occlusion" sequence, where only the circles were overlapped. This overlap changes the shape and appearance of the circle. Thus, two occluded circles resemble the shape eight instead of

Table 4.7: Summary of classification performance for all laboratory test sequences using three network architecture and two types of the classifier on the balanced dataset. LC refers to the linear classifier, and ELM refers to the extreme learning machine classifier.

		True Positive (%)		False Positive (%)		Sensitivity (%)		Specificity (%)		Informedness (%)	
		LC	ELM	LC	ELM	LC	ELM	LC	ELM	LC	ELM
Shape translation	Standard Weights	65.3	45.5	56.5	35.8	94.70	91.51	80.31	76.93	75.00	68.44
	Dedicated Weights	34.5	35.5	21.7	38.9	91.87	85.82	77.44	82.35	69.31	68.17
	Mixed Weights	42.9	35.6	11.5	12.5	98.42	96.12	89.82	87.89	88.24	84.02
Two classes occlusion	Standard Weights	32.5	25.5	15.5	10.8	73.48	70.58	66.54	68.35	40.01	38.93
	Dedicated Weights	57.5	59.8	45.5	39.8	92.39	86.78	77.03	77.76	69.42	64.55
	Mixed Weights	12.5	32.5	18.5	20.5	67.67	64.48	67.67	60.11	25.62	24.60
Multiple objects	Standard Weights	35.6	34.7	23.5	24.8	74.41	72.26	70.54	72.26	44.95	42.48
	Dedicated Weights	10.5	9.5	25.5	17.5	10.84	16.30	99.99	99.93	10.79	16.23
	Mixed Weights	47.8	35.8	39.5	45.5	74.30	70.14	73.29	73.32	47.58	43.46
Different sizes	Standard Weights	65.8	55.4	35.4	25.8	87.73	86.20	84.46	85.63	72.19	71.83
	Dedicated Weights	45.3	45.2	25.7	17.8	82.02	79.84	81.54	85.34	63.56	65.18
	Mixed Weights	65.5	45.6	23.5	15.9	87.33	85.44	87.13	86.94	74.46	72.38
One class occlusion	Standard Weights	45.5	48.9	12.5	18.9	80.10	73.54	76.91	81.16	62.36	54.70
	Dedicated Weights	45.8	23.5	55.6	39.7	83.24	76.76	71.26	79.37	54.51	56.13
	Mixed Weights	51.8	37.4	34.8	38.4	84.03	76.91	85.18	85.29	69.21	62.19
Complex shapes	Standard Weights	12.5	19.5	38.4	34.5	41.07	43.52	76.35	73.68	17.42	17.20
	Dedicated Weights	55.2	59.5	45.3	38.5	99.19	94.93	70.85	74.81	70.04	69.74
	Mixed Weights	12.5	16.5	35.5	38.6	42.38	44.78	77.50	73.24	19.88	18.01

two separate circles. The highest informedness was achieved using the "mixed weights" architecture on the balanced data with 69.21% using the linear classifier followed by 50.11% using the same architecture and classifier but on the imbalanced data. The last sequence, "Complex objects", consists of a 2D representation of a plant with two hanging fruits and a few leaves. Complex non-uniform shapes were passed to the event-based classifier pipeline in this scenario. The network achieved 81.3% informedness using the mixed weights method on the balanced dataset using the linear classifier, followed by an informedness of 76.8% using the standard network using the linear classifier.

To improve the model performance during classification, we regularised OPIUM by slightly changing the regularisation factor during the feedforward pass. The weights mapping between the input and output was regularised within a specified regularisation range. In this work, we experimented with a regularisation range from 1 to 1000 and showed the classification informedness for linear and ELM classifiers as shown in Figure 4.31 and Figure 4.32 respectively. The regularisation effect on informedness was somewhat mixed for both classification methods and network architectures since each condition present a unique challenge to the classifier. For instance, for test sequence 1, we observed an increase in informedness using the imbalanced data using the "dedicated weights" method with the linear classifier, and a decrease in performance using the balanced data, showing that balancing the classes can change the distribution and statistics of the data and lead to different network behaviour. We noticed that the impact of regularisation was more evident using the linear classifier than with the ELM. For instance, all the informedness results on the test sequences showed a fluctuation in the model performance for different regularisation values with a slight increase in the informedness at different regularisation values. This was reasonable and expected because, in the ELM classifier, the weights in the first layer are randomly initialised, which change the statistics on the input attribute before computing the output weight with

OPIUM. The investigation of the regularisation helped identify the highest and lower performance of the classification network in a view to select the correct regularisation value for the suitable dataset.

These results demonstrate the effectiveness of the end-to-end system in transforming boisterous raw input events into sparse, highly informative noise-free event streams in various scenarios. Overall the mixed weights approach achieved superior performance compared to the other methods. This demonstrates the benefit of having feature selection during the initial training. On the other hand, balancing the dataset by undersampling the dominant class improved the model performance. More detailed results about the model performance were summarised from Figure C.8 to Figure C.13 in Appendix C.

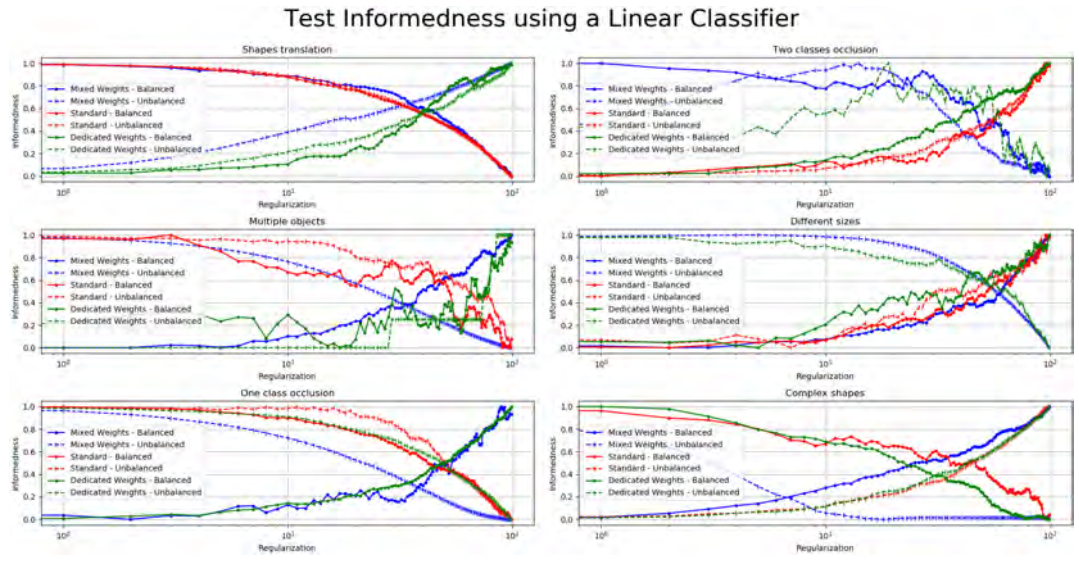


Figure 4.31: Performance comparison of the linear classifier for each architecture and for balanced and non-balanced classes on the laboratory recorded datasets.

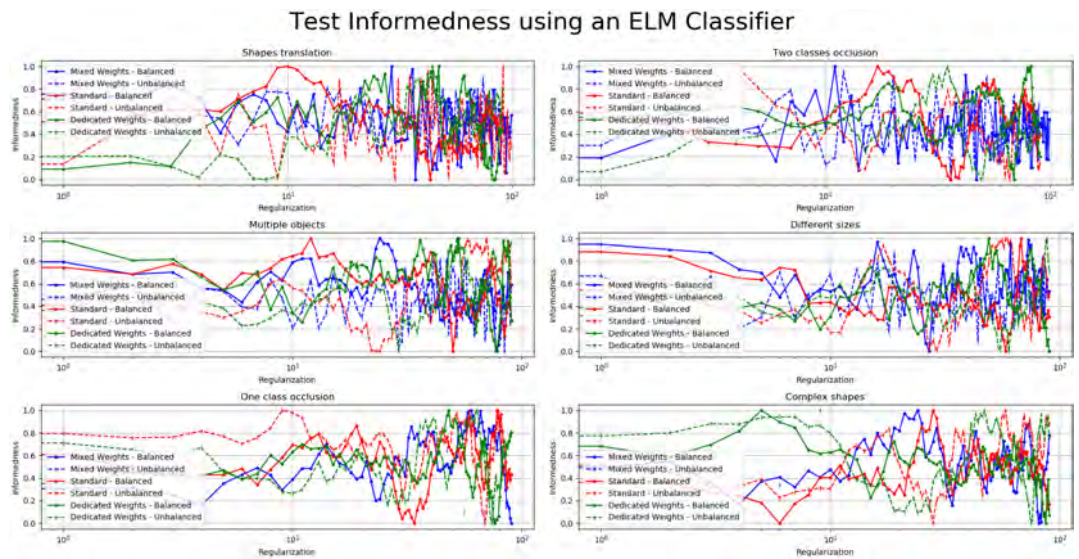


Figure 4.32: Performance comparison of the ELM classifier for each architecture and for balanced and non-balanced classes on the laboratory recorded datasets.

4.5.4 Performance on Real World Scenes

Unlike N-MNIST and DVS gesture which is more structured, the real-world datasets pose an exciting challenge for the event-based processing pipeline. The structure of this dataset represents the actual real world and provides more spatially and temporally rich information that does not exhibit the same consistency between samples of the same class. One major obstacle in real-world processing scenes was the highly imbalanced classes and the non-uniformity of the input data, as shown in Table 4.2, the percentage of the average event between class 1 and class 0 was 98% by 2% making the problem extremely imbalanced. Events from class1 can be considered rare events. To tackle this extremely challenging problem, we explored the use of GLS in section 4.5.8 which utilised the same framework with only minor modifications made to support the differing nature of the input events and showed the model performance without under-sampling or oversampling. All the work performed on the real-world dataset attempted to use as much of the same processes and methodology as used in the classification of the lab recorded dataset detailed in Section 4.5.3. The detailed results for all real-world recordings were summarised in Figure 4.33.

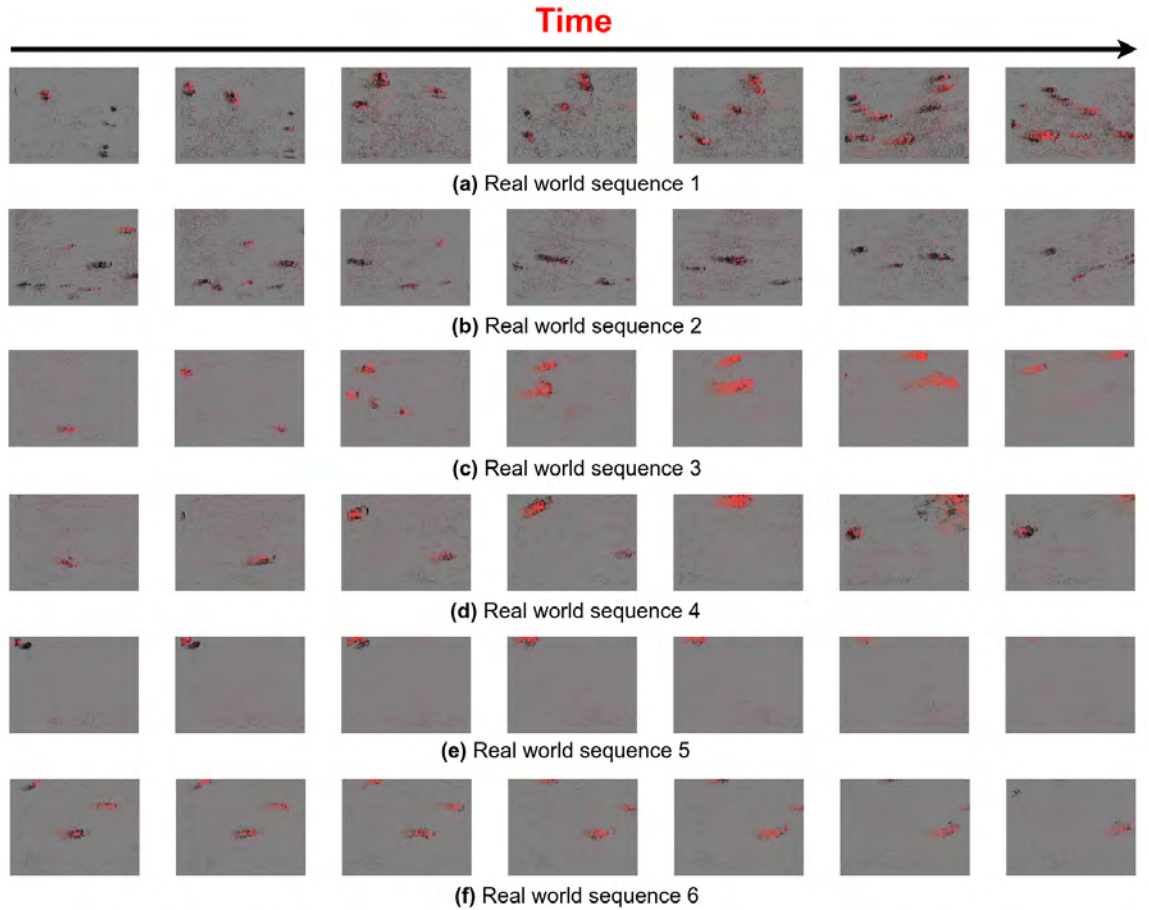


Figure 4.33: Performance of the network on the real-world recorded dataset. Correctly classified instances as red dots, black dots are other class. (a), (b), (c), (d), (e), (f) show the spatio-temporal patterns of the classification output for each test sequences. Original datasets are described in Section 4.4.1.

Table 4.8: Summary of classification performance for all real-world test sequences using three network architecture and two types of the classifier on the imbalanced dataset. LC refers to the linear classifier, and ELM refers to the extreme learning machine classifier.

		True Positive (%)		False Positive (%)		Sensitivity (%)		Specificity (%)		Informedness (%)	
		LC	ELM	LC	ELM	LC	ELM	LC	ELM	LC	ELM
Real world Sequence 1	Standard Weights	0.1	0.5	0.2	1.2	1.3	2.3	99.99	98.78	0.8	1.08
	Dedicated Weights	0.2	3.5	3.2	12.3	2.5	0.4	98.85	99.99	1.35	0.39
	Mixed Weights	1.3	2.5	5.6	11.2	1.1	1.4	99.85	99.99	0.95	1.39
Real world Sequence 2	Standard Weights	2.3	6.8	8.6	25.3	9.67	9.68	95.31	96.45	4.98	6.13
	Dedicated Weights	0.1	4.7	3.9	5.3	1.4	1.23	99.99	99.99	1.39	1.22
	Mixed Weights	0.1	2.7	5.6	10.2	2.3	2.4	98.52	98.85	1.82	1.25
Real world Sequence 3	Standard Weights	0.6	6.5	0.2	12.3	2.3	4.5	99.99	99.99	2.29	4.49
	Dedicated Weights	0.2	5.9	0.6	10.2	2.7	4.5	99.99	99.78	2.69	4.37
	Mixed Weights	0.8	2.4	0.9	18.6	1.7	2.8	98.85	99.99	0.55	2.8
Real world Sequence 4	Standard Weights	0.6	3.5	0.1	2.3	0.2	0.8	99.99	99.97	0.19	0.77
	Dedicated Weights	1.2	4.5	0.3	1.8	1.6	4.3	99.55	99.99	1.15	4.29
	Mixed Weights	2.3	8.7	1.1	5.6	1.57	0.53	99.95	99.54	1.52	0.07
Real world Sequence 5	Standard Weights	0.5	6.5	0.5	7.5	0.5	0.9	99.99	99.97	0.49	0.87
	Dedicated Weights	0.3	3.5	0.6	8.7	5.6	0.5	98.32	99.99	3.92	0.49
	Mixed Weights	0.7	7.8	0.6	8.9	2.6	5.6	99.99	99.99	2.59	5.59
Real world Sequence 6	Standard Weights	1.2	5.6	0.9	4.5	2.3	4.6	99.99	99.75	2.29	4.59
	Dedicated Weights	2.5	8.9	1.8	10.5	3.5	0.5	96.57	99.99	0.7	0.49
	Mixed Weights	0.2	3.5	1.1	24.5	0.9	5.3	99.97	99.99	0.87	2.29

Table 4.9: Summary of classification performance for all real-world test sequences using three network architecture and two types of the classifier on the balanced dataset. LC refers to the linear classifier, and ELM refers to the extreme learning machine classifier.

		True Positive (%)		False Positive (%)		Sensitivity (%)		Specificity (%)		Informedness (%)	
		LC	ELM	LC	ELM	LC	ELM	LC	ELM	LC	ELM
Real world Sequence 1	Standard Weights	26.7	15.9	24.8	17.7	26.69	24.78	88.18	84.62	14.87	9.40
	Dedicated Weights	27.3	3.3	26.7	4.1	27.26	26.72	93.35	91.82	20.60	18.54
	Mixed Weights	17.4	11.3	16.4	13.6	37.43	46.44	87.40	92.79	24.83	39.23
Real world Sequence 2	Standard Weights	19.7	5.9	25.1	11.7	39.67	25.11	88.15	86.51	27.82	11.62
	Dedicated Weights	39.2	3.8	36.8	6.2	59.17	36.77	92.45	87.53	51.62	24.30
	Mixed Weights	13.2	8.7	14.7	11.2	33.20	44.73	82.62	77.56	24.18	37.71
Real world Sequence 3	Standard Weights	40.3	31.8	28.4	21.2	91.23	58.36	95.52	77.60	86.75	35.95
	Dedicated Weights	37.7	5.0	35.8	6.4	37.72	35.79	90.05	87.29	27.78	23.08
	Mixed Weights	44.4	36.5	35.5	26.6	44.40	35.50	77.08	86.71	21.52	22.21
Real world Sequence 4	Standard Weights	31.3	22.2	27.6	19.6	41.30	57.63	85.58	80.89	56.88	38.52
	Dedicated Weights	39.0	9.6	40.6	12.0	38.96	40.65	80.78	76.09	19.74	16.73
	Mixed Weights	27.7	21.2	34.3	24.2	47.67	54.27	77.52	51.54	25.19	5.81
Real world Sequence 5	Standard Weights	37.1	25.8	30.7	23.7	77.12	50.75	48.32	72.60	25.44	23.35
	Dedicated Weights	39.4	32.7	32.7	5.5	39.36	32.72	89.32	89.02	28.69	21.74
	Mixed Weights	38.6	28.1	39.5	28.5	78.65	89.53	73.90	42.96	52.55	32.49
Real world Sequence 6	Standard Weights	20.6	12.0	27.8	16.7	60.59	37.80	76.10	76.57	36.69	14.37
	Dedicated Weights	36.8	2.8	39.8	4.0	36.79	39.77	94.30	91.96	31.09	31.73
	Mixed Weights	26.9	16.2	22.6	13.7	39.90	52.55	77.64	72.65	17.54	25.24

Figure 4.33 shows the classification output on the real-world dataset where the classes were equally balanced before the fully connected layer. The undersampling was applied to the data because the background contains no valid objects, which can produce unexpected and erratic results if trained with the existing network architectures. The network performance can be dramatically deteriorated by including background events as they occupy most input signals. The challenge was primarily a practical one relating to mapping pixels to the classifier input channels. The network performance increased when class0 was undersampled. Each row in Figure 4.33 represented the classification results on a single test sequence from the real-world environment. Each test

sequence has a different event structure (e.g. temporally and spatially), recorded with different non-linear motion, in different directions, and has different points of view toward the objects of interest. Although the undersampling was applied on the dominant class, the noise rate was extremely high due to the low SNR in the events data. The network segregated the events that belong to the fruits vs the ones that belong to the background. However, the results were less significant compared with the lab recorded datasets.

Figure 4.34 and Figure 4.35 presented the results of all the classification architectures. The same parameters were used for the balanced and imbalanced classes to allow a valid comparison between the methods. It was obvious from the results that the classification was not far from a chance using the balanced dataset, which would be 50%, thereby displaying little predictive power especially using "mixed weights" architecture. On the other hand, the classification shows poor performance on the imbalanced data due to the high skewness of the data toward the dominant class, making the data non-linearly separable. The highest informedness was achieved on the test sequence 3 with 86.75% using the "standard weights" architecture on the balanced dataset with a linear classifier as shown in table 4.9. Overall, it was evident that by using the "dedicated weights" architecture, the network produced better results using both the ELM and linear classifier. However, the results were just below chance for some test sequences. As shown in Table 4.8 the classifier failed to classify the highly imbalanced data resulting in informedness resulting in a below chance overall architectures, which indicates the limitation of the network to classify highly imbalanced data. Given that the lab recorded datasets provides a far easier challenge due to the consistent motion and the lack of background or clutter in the data, it is likely that a network trained on more complex data, such as the sequences in the real world scenes, will require a higher number of training samples before the accuracy will converge. It is also likely that the equations for the number of hidden layer nodes will change, primarily due to the increased number of input channels.

To validate the performance of all the network pipelines, we regularised the classifiers using a regularisation range from 1 to 1000 to find the value that gives the lowest classification error and highest informedness, similarly to the one performed in Section 4.5.3. Regularisation results for linear classifier and ELM classifier are illustrated in Figure 4.34 and Figure 4.35 respectively. There was a dramatic increase in the informedness for the linear classifier using the "mixed weights" and "standard weights" architecture, which peaked at a higher regularisation value using the balanced dataset. With the "dedicated weights" architecture, the informedness increased only for test sequences 3 and 6 with the balanced data. All the imbalanced data showed a reduction in the network performance at a higher regularisation value. On the other hand, the results for the ELM classifier were very similar to the results in Figure 4.31. In this case, regularising the ELM slightly improves the network classifier due to the random weight initialisation in the hidden layer, which changes the weight distribution with every regularisation value. However, this regularisation remains essential to help to

select the correct regularisation value for the suitable dataset.

The object classification task on the real-world dataset is a difficult problem. Implementing such a task using the networks proposed in Section 4.4.7 presents additional challenges due to the immense size of the problem, the data and the input channels. The size of the problem also demands significant computational power, and a full classification pipeline can take a longer time to complete and a higher memory capacity. The number of input channels required to cater for patterns in real-world scenes is one of the major contributing factors. As described in Section 4.4.5 and Section 4.4.6, the size of the input channels can exceed a few million attributes depending on the chosen downsampling factor to fully cater for all data sizes. In the case of ELM classifier, the number of hidden neurons was limited to only 100 neurons to prevent memory overload during weight mapping. This can increase as the number of data samples increases, which presents a difficult problem for optimisation as these input channels need to interact with a similarly large matrix of random weights. The essence of this operation is a large multiplication requiring either sequential execution on the CPU or to incur the penalties involved with transferring the entire input pattern vector to the GPU for parallel execution. More detailed results about the model performance were summarised from Figure C.14 to Figure C.14 in Appendix C.

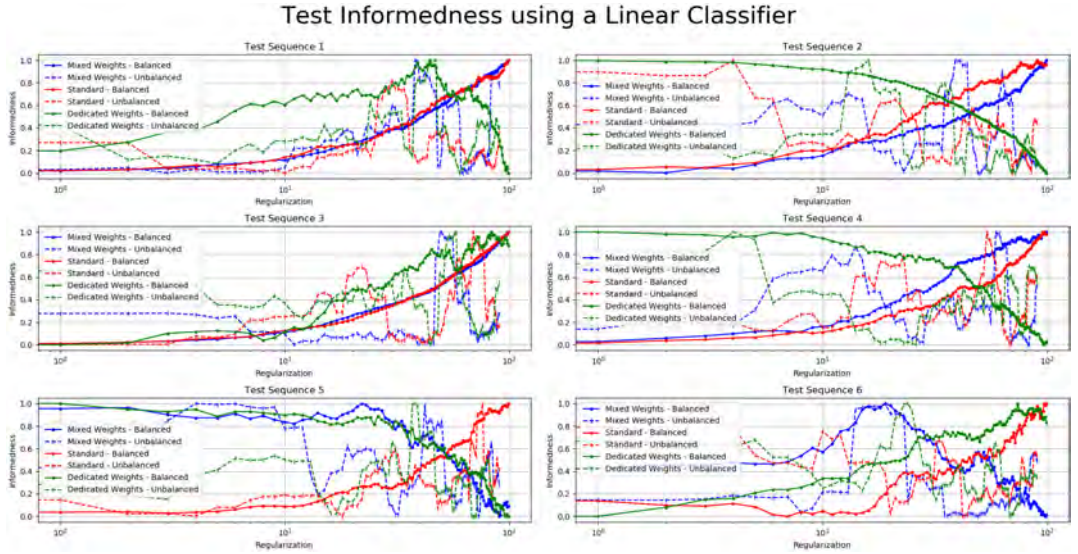


Figure 4.34: Performance comparison of the linear classifier for each architecture and for balanced and non-balanced classes on the real world datasets.

4.5.5 Investigation of Spatial Downsampling

This section explores the effects of performing per pixel downsampling operations on the input patterns before learning them with the event-based object classifier described in Section 4.5.5. A full downsampling sweep from $\sigma = 1$ to $\sigma = 28$ was investigated to evaluate its contribution to the network performance. The downsampling method was performed on the x and y addresses using the same downsampling factor to maintain the original aspect ratio of the events. The same trained weights were used at inference,

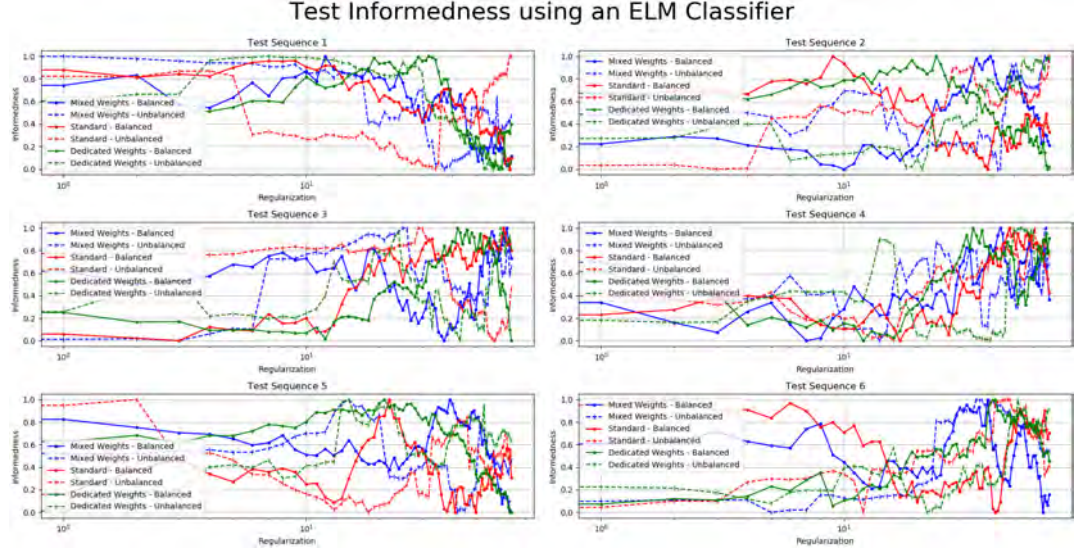


Figure 4.35: Performance comparison of the ELM classifier for each architecture and for balanced and non-balanced classes on the real world datasets.

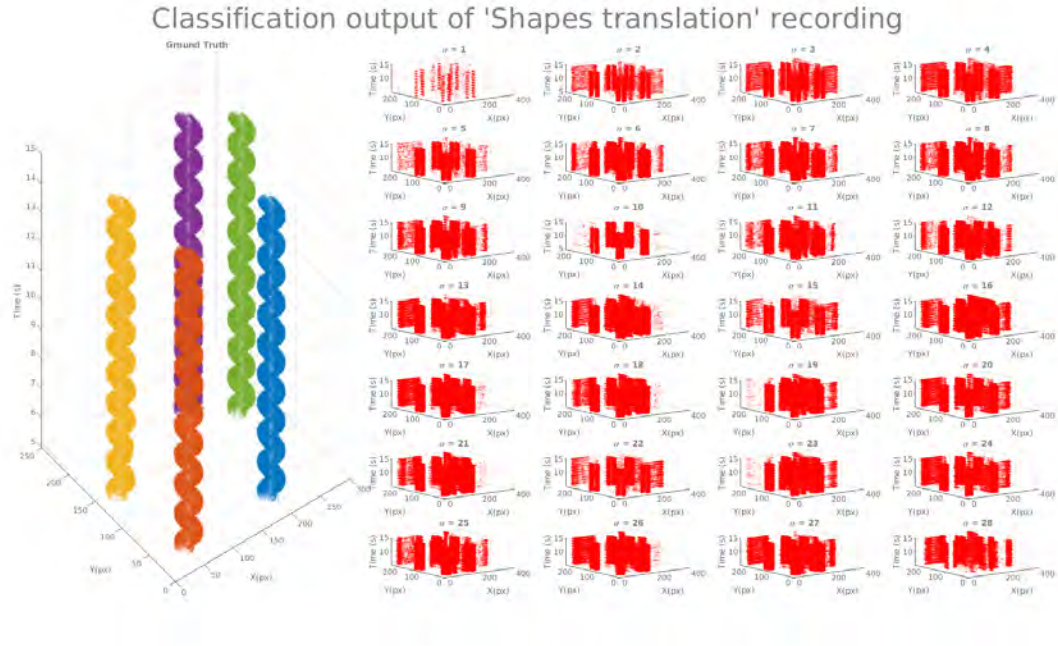
and the only variable in the network was the downsampling factor σ .

The results revealed a strong correlation between the size of the object and the selected σ using the laboratory recorded datasets. In this case, the downsampling technique slightly improved the model results. As shown in Figure 4.36(a), for $\sigma = 10$, the spatio-temporal pattern from the classification output produced fewer false-positive and had a peak performance of 0.75 for the correlation coefficient metric. The downsampling factor " σ " matched the object's size, leading to better generalisation in the classification network. The same results were observed on other laboratory recorded test sequences, especially when the object shape was uniform as shown in Appendix C from Figure C.24 to Figure C.28. In addition, increasing σ has increased the percentage of the event correctly predicted as class1, showing that a higher downsampling factor leads to a higher true positive. In contrast, the number of events classified as class0 decreased with a higher σ , which indicates less false positive as shown in Table 4.10.

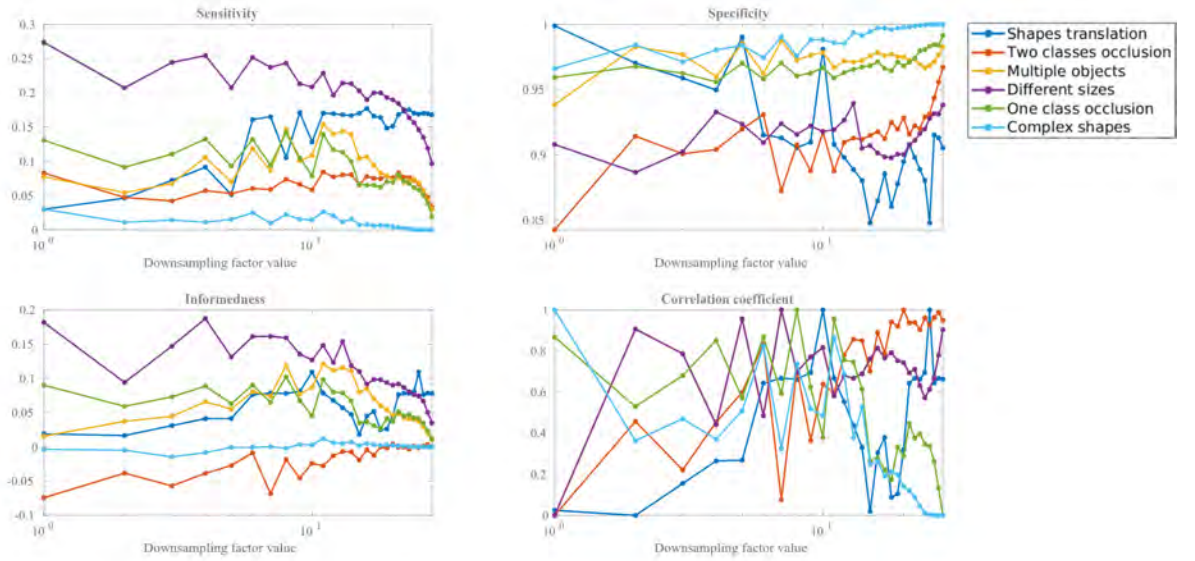
The same downsampling sweep was repeated on all real-world test sequences. The results vary from those achieved on the laboratory recorded dataset in that the effect of the downsampling sweep was more visible in space-time. For example, it was evident that higher σ lead to fewer events from the classification output without having any significant reduction in the classification performance as shown in Figure 4.37(a), the same behaviour was also observed for other test sequences from Figure C.29 to Figure C.32. The changes in the event's output may result from the non-uniform object size as opposed to the laboratory recorded data, where the objects were uniform in shape and size. In this case, the spatial downsampling showed that the model could produce the same performance with less data than the classifier as illustrated in Figure 4.37(b). Hence, the resolution of high dimensional data can be reduced without harming the classification performance, which proved the advantage of this technique. On the other hand, it was observed that with a higher σ the percentage of true positive decrease for

Table 4.10: Results summary of the per-pixel downsampling method on the laboratory recorded datasets. Showing the percentage of events correctly classified as class1 (true positive) and class0 (false positive). σ is the downsampling factor.

Downsampling factor	Percentage of events correctly classified for class1										Percentage of events correctly classified for class0													
	Shapes translation		Two classes occlusion		Multiple objects		Different sizes		One class occlusion		Complex shapes		Shapes translation		Two classes occlusion		Multiple objects		Different sizes		One class occlusion		Complex shapes	
$\sigma=1$	0.38	5.83	14.39	4.33	6.15	17.18	5.19	6.29	2.60	0	99.61	94.17	95.67	85.61	97.40	100								
$\sigma=2$	7.02	8.14	17.18	6.15	5.19	6.29	2.60	0	99.61	94.17	95.67	85.61	97.40	100										
$\sigma=3$	10.54	10.10	18.77	8	7.20	20.14	7.20	0.02	86.80	89.90	90.67	79.86	92.80	99.98										
$\sigma=4$	13.19	10.10	21.77	9.33	7.77	22.75	8.41	0.21	77.04	86.22	87.32	78.23	92.23	99.94										
$\sigma=5$	10.03	12.68	21.77	9.99	7.77	22.75	8.41	0.21	77.04	86.22	87.32	78.23	92.23	99.94										
$\sigma=6$	22.95	13.78	22.75	9.51	8.41	22.75	8.41	0.21	77.04	86.22	87.32	78.23	92.23	99.94										
$\sigma=7$	23.55	13.66	24.29	9	9.23	24.29	9.23	15.92	76.44	86.34	87.32	78.23	92.23	99.94										
$\sigma=8$	24.89	14.81	25.41	9.76	10.45	25.41	10.45	0.55	75.10	85.19	86.34	87.32	92.23	99.94										
$\sigma=9$	24.44	13.13	26.47	9.14	9.51	26.47	9.51	0.63	75.55	86.87	87.32	78.23	92.23	99.94										
$\sigma=10$	14.38	13.99	26.79	9.83	9.37	26.79	9.37	0.86	73.21	85.61	86.01	73.21	90.63	99.14										
$\sigma=11$	24.49	13.25	27.64	10.06	9.13	27.64	9.13	1	75.50	86.75	87.32	78.23	92.23	99.94										
$\sigma=12$	25.24	14.26	27.59	11.30	9.24	27.59	9.24	0.83	74.75	85.74	86.01	73.21	90.63	99.14										
$\sigma=13$	25.87	14.11	26.34	12.33	8.84	26.34	8.84	1.04	74.12	85.89	87.32	78.23	92.23	99.94										
$\sigma=14$	26.45	13.12	27.21	12.30	9.23	27.21	9.23	1.25	73.54	86.88	87.32	78.23	92.23	99.94										
$\sigma=15$	11.80	14.73	28.33	15.98	12.56	28.33	12.56	2.37	88.19	85.27	86.01	73.21	90.63	99.14										
$\sigma=16$	29.48	14.69	25.88	16.56	14.04	25.88	14.04	1.69	70.51	85.31	86.01	73.21	90.63	99.14										
$\sigma=17$	25.99	14.63	25.02	16.09	14.55	25.02	14.55	3.37	74.00	85.45	86.01	73.21	90.63	99.14										
$\sigma=18$	27.89	16.97	28.70	17.77	17.02	28.70	17.02	3.84	72.10	83.03	84.02	71.67	87.44	97.63										
$\sigma=19$	25.06	12.22	26.90	12.53	10.53	26.90	10.53	2.53	74.93	87.78	88.70	73.10	89.47	97.47										
$\sigma=20$	23.92	15.27	26.95	11.90	13.43	26.95	13.43	2.61	76.07	84.73	85.59	70.99	88.21	98.15										
$\sigma=21$	24.56	14.41	30.28	16.64	17.10	30.28	17.10	4.27	75.52	85.59	86.01	73.21	90.63	99.14										
$\sigma=22$	22.78	15.84	29.01	9.63	11.79	29.01	11.79	1.85	84.16	73.25	75.56	68.55	83.72	95.31										
$\sigma=23$	21.75	11.34	31.45	14.59	16.28	31.45	16.28	4.69	75.56	88.66	89.90	79.86	92.80	99.98										
$\sigma=24$	26.89	11.55	26.15	8.15	11.60	26.15	11.60	2.95	72.24	88.45	89.90	79.86	92.80	99.98										
$\sigma=25$	22.58	13.06	30	13.40	16.41	30	16.41	2.79	71.51	86.94	87.32	78.23	92.23	99.94										
$\sigma=26$	24.65	11.89	31.15	8.53	13.83	31.15	13.83	3.89	65.75	88.11	89.90	79.86	92.80	99.98										
$\sigma=27$	23.57	11.44	28.70	6.81	11.66	28.70	11.66	2.51	88.56	88.56	89.90	79.86	92.80	99.98										
$\sigma=28$	24.56	20.34	33.56	12.27	15.91	33.56	15.91	5.73	79.66	69.85	73.10	89.47	97.47	97.47										



(a)



(b)

Figure 4.36: Results of the downsampling sweep on the shapes translation sequence. (a) The output spatio-temporal patterns were generated from the classification output at 28 different spatial resolutions. It shows that for $\sigma = 10$ the network classified events with less noise and fewer false positives. (b) The results of the evaluation matrices used 28 different downsampling factors for all test sequences.

class1 and leading to an increase in the false positive for class0 as shown in Table 4.11, however, this had minimal effect on the classification performance.

These results were insightful, as the chosen spatial resolutions were based on logical extensions from the camera's parameters, and the number of input channels was derived

Table 4.11: Summary of per-pixel downsampling method on the real-world recorded dataset. Showing the percentage of events correctly classified as class1 (true positive) and class0 (false positive). σ is the downsampling factor.

Downsampling factor	Percentage of events correctly classified for class1										Percentage of events correctly classified for class0									
	Test sequence 1	Test sequence 2	Test sequence 3	Test sequence 4	Test sequence 5	Test sequence 6	Test sequence 1	Test sequence 2	Test sequence 3	Test sequence 4	Test sequence 5	Test sequence 6	Test sequence 1	Test sequence 2	Test sequence 3	Test sequence 4	Test sequence 5	Test sequence 6	Test sequence 1	Test sequence 2
	22.23	24.93	32.47	39.59	52.22	48.60	77.76	75.06	67.52	60.40	47.77	51.39	77.76	75.06	67.52	60.40	47.77	51.39	77.76	75.06
$\sigma=1$	22.23	24.93	32.47	39.59	52.22	48.60	77.76	75.06	67.52	60.40	47.77	51.39	77.76	75.06	67.52	60.40	47.77	51.39	77.76	75.06
$\sigma=2$	22.76	18.40	33.60	55.72	63.63	48.17	77.23	81.59	66.39	44.27	36.36	51.83	77.23	81.59	66.39	44.27	36.36	51.83	77.23	81.59
$\sigma=3$	27.91	15.89	37.77	57.88	66.30	53.07	72.08	84.10	62.22	42.11	33.69	46.92	72.08	84.10	62.22	42.11	33.69	46.92	72.08	84.10
$\sigma=4$	27.51	14.99	42.00	58.05	67.14	53.96	72.48	85.00	57.99	41.94	32.85	46.03	72.48	85.00	57.99	41.94	32.85	46.03	72.48	85.00
$\sigma=5$	21.01	13.28	43.18	53.52	69.59	58.98	78.98	86.71	56.81	46.47	30.40	41.01	78.98	86.71	56.81	46.47	30.40	41.01	78.98	86.71
$\sigma=6$	16.71	10.53	46.46	56.40	66.65	57.09	83.28	89.46	53.53	43.59	33.340	42.90	83.28	89.46	53.53	43.59	33.340	42.90	83.28	89.46
$\sigma=7$	12.84	9.39	49.76	55.25	70.61	51.13	87.15	90.60	50.23	44.74	29.38	48.86	87.15	90.60	50.23	44.74	29.38	48.86	87.15	90.60
$\sigma=8$	10.30	9.02	50.96	48.48	67.92	50.66	89.69	90.97	49.03	51.51	32.07	49.33	89.69	90.97	49.03	51.51	32.07	49.33	89.69	90.97
$\sigma=9$	10.39	9.51	50.41	48.08	70.01	46.92	89.60	90.48	49.58	51.91	29.98	53.07	89.60	90.48	49.58	51.91	29.98	53.07	89.60	90.48
$\sigma=10$	6.25	9.06	60.67	43.19	65.18	45.43	93.74	90.93	39.32	56.80	34.81	54.56	93.74	90.93	39.32	56.80	34.81	54.56	93.74	90.93
$\sigma=11$	6.59	7.65	60.26	39.64	63.98	44.67	93.40	92.34	39.73	60.35	36.01	55.32	93.40	92.34	39.73	60.35	36.01	55.32	93.40	92.34
$\sigma=12$	4.94	7.16	62.62	35.08	63.63	43.33	95.05	92.83	37.37	64.91	36.36	56.66	95.05	92.83	37.37	64.91	36.36	56.66	95.05	92.83
$\sigma=13$	4.71	9.04	62.53	38.87	66.53	43.51	95.28	90.95	37.46	61.12	33.46	56.48	95.28	90.95	37.46	61.12	33.46	56.48	95.28	90.95
$\sigma=14$	4.87	8.62	65.31	35.39	67.32	47.20	95.12	91.37	34.68	64.60	32.67	52.79	95.12	91.37	34.68	64.60	32.67	52.79	95.12	91.37
$\sigma=15$	4.91	7.95	54.11	40.10	63.80	56.37	95.08	92.04	45.88	59.89	36.19	43.62	95.08	92.04	45.88	59.89	36.19	43.62	95.08	92.04
$\sigma=16$	4.44	9.35	60.59	43.23	60.01	43.98	95.55	90.64	39.40	56.76	56.01	56.01	95.55	90.64	39.40	56.76	56.01	56.01	95.55	90.64
$\sigma=17$	3.61	8.98	62.62	42.48	62.75	45.24	96.38	91.01	37.37	57.51	37.24	54.75	96.38	91.01	37.37	57.51	37.24	54.75	96.38	91.01
$\sigma=18$	4.32	7.88	43.74	48.47	61.98	58.37	95.67	92.11	56.25	51.52	38.01	41.62	95.67	92.11	56.25	51.52	38.01	41.62	95.67	92.11
$\sigma=19$	6.32	7.12	44.58	38.17	50.02	53.88	93.67	92.87	55.41	61.82	49.97	46.11	93.67	92.87	55.41	61.82	49.97	46.11	93.67	92.87
$\sigma=20$	4.04	6.55	36.75	53.23	59.08	48.78	95.95	93.44	63.24	46.76	40.91	51.21	95.95	93.44	63.24	46.76	40.91	51.21	95.95	93.44
$\sigma=21$	5.85	6.33	41.39	57.64	67.85	48.02	94.14	93.66	58.60	42.35	32.14	51.97	94.14	93.66	58.60	42.35	32.14	51.97	94.14	93.66
$\sigma=22$	4.76	9.49	42.38	52.62	60.19	44.62	95.23	90.50	57.61	47.37	39.80	55.37	95.23	90.50	57.61	47.37	39.80	55.37	95.23	90.50
$\sigma=23$	4.45	8.11	28.28	59.55	50.84	50.24	95.54	91.88	71.71	40.44	49.75	54.75	95.54	91.88	71.71	40.44	49.75	54.75	95.54	91.88
$\sigma=24$	3.91	7.50	38.19	66.34	62.47	45.22	96.08	92.49	61.80	33.65	37.52	54.77	96.08	92.49	61.80	33.65	37.52	54.77	96.08	92.49
$\sigma=25$	3.40	8.13	29.06	68.72	63.53	34.81	96.59	91.86	70.93	31.27	36.46	65.18	96.59	91.86	70.93	31.27	36.46	65.18	96.59	91.86
$\sigma=26$	3.35	8.27	43.68	66.85	49.48	47.54	96.64	91.72	56.31	33.14	52.45	52.45	96.64	91.72	56.31	33.14	52.45	52.45	96.64	91.72
$\sigma=27$	4.30	9.95	48.04	66.10	40.19	36.89	95.69	90.04	51.95	33.89	59.80	63.10	95.69	90.04	51.95	33.89	59.80	63.10	95.69	90.04
$\sigma=28$	3.25	6.73	38.26	49.35	34.34	35.31	96.74	93.26	61.73	50.64	65.65	64.68	96.74	93.26	61.73	50.64	65.65	64.68	96.74	93.26

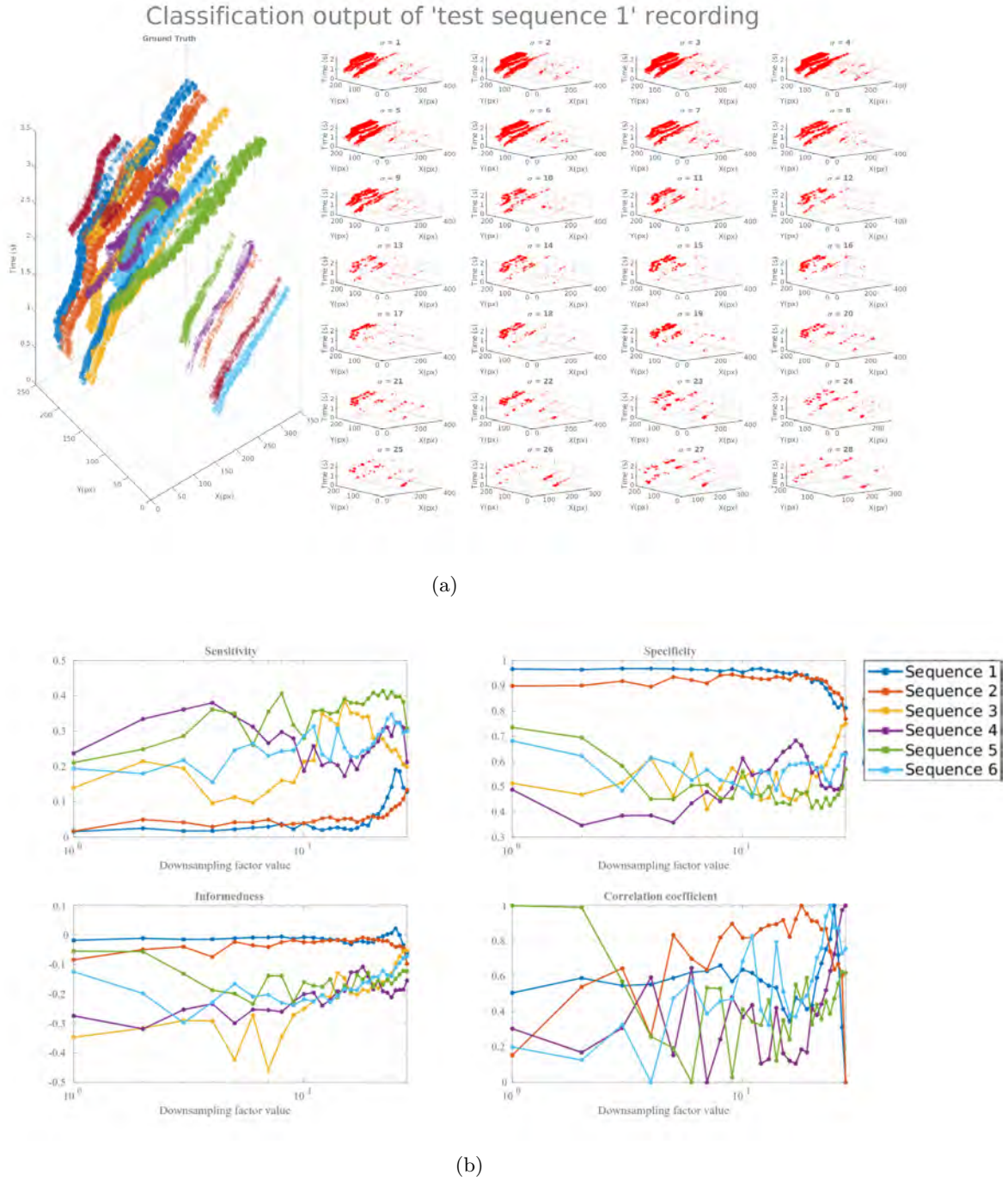


Figure 4.37: Results of the downsampling sweep on the real world sequence 1. (a) The output spatio-temporal patterns generated from the classification output at 28 different spatial resolutions. (b) The results of the evaluation matrices using 28 different downsampling factors for all test sequences.

directly from the number of pixels. These results showed that the per-pixel downsampling technique could help in situations where the size of the objects are uniform in the scenes. It was also evident that it did not lead to a significant information loss, and the performance showed little predictive error even with a high downsampling factor. This was because when the events were pooled using the STP filter and passed through the

input and the hidden layer of the classifier, it masked the originating channels, as the hidden layers only attend an aggregated sum of all input layer activity even with a set of random weights. Therefore, the spatial structure of any input was inherently lost when the data arrived at the hidden layer neurons as the downsampling step maps a region of the input to a single channel, effectively creating a small receptive field for each input to classifier networks. However, the classifier networks still ignored the input's spatial relationships as each input now encodes a little spatial information. Hence, the larger the spatial downsampling, the more spatial information is encoded.

4.5.6 Investigation of Event-based Feature Selection

This section explores the effect of performing feature selection on feature extraction, which was described in detail in Section 4.4.7. In this work, feature selection aims to optimise the number of features in the subsequent network to enhance classification performance and generalisation. The actual raw events usually contain redundant features, which provide irrelevant and superfluous features that provide no useful information to the network due to the scenes' structure and highly skewed number of events in each class. For that reason, these features are best removed. Feature selection is helpful as a part of the data analysis process, which can determine the importance of each feature in classification, revealing the relationship between the selected features and the incoming data.

To generate balanced features during feature extraction, the supervisory signals were fed as input to FEAST and the features training was divided into two phases to manage the number of neurons for each class. The number of events was highly imbalanced in our datasets, where class0 occupied 75%-90% of the original data. Learning all these events will produce bias features. For that reason, two neurons were assigned for class0, and a variable number of neurons between 4 and 28 were allocated for class1 to have more robust representations for the object of interest. Figure 4.38 and Figure 4.39 show the results of feature selection on the lab recorded datasets. It showed that the model became an efficient circle detector by increasing the number of neurons for class1 instead of having fewer neurons for class0. The output events stream shows a clear stream of events for the class of interest in space-time as shown in red and the other class (blue events) with fewer false positives. This allowed for better events segregation during feature extraction and led to a more reliable generaliser.

As shown in Figure 4.38 the classification slightly increased when the number of feature/neurons for class1 were higher. For instance, the classifier sensitivity for test sequence-4 was 0.45 when two neurons were for class0 and four for class1. The sensitivity became 0.82 when the number of neurons for class1 increased to 28. Likewise, the informedness for test sequence-6 increased from 0.14 to 0.71, showing the positive effect of selecting more neurons for class1. The accuracy measure was not displayed as it does not fully reflect the model performance for highly imbalanced data. Based on this, the model performance was judged on other evaluation metrics. Overall, these results vary for each recording depending on the balance ratio for class1/class0 and the uniformity

of the sequence. This method was tested on all laboratory recorded test sequences. The output of the classifier in space and time are presented in Figure 4.39. In contrast to the results on the imbalanced dataset across the laboratory recorded datasets, the model performance showed little performance difference on the balanced dataset for each real-world test sequence as shown in Figure 4.40 and Figure 4.41. For example, for test sequence-1, the sensitivity increased from 0.20 to 0.24, and the informedness increased from 0.05 to 0.20.

Moreover, the increase in the model performance was similar to all other test sequences. This was expected because, for real-world scenes, the fruits' shape was identical to the shape of the leaves. Both contain curves with the same structure, confusing the network during feature extraction and classification. It then led to a slight improvement over the balanced dataset.

The results of the event-based feature selection show that we need more neurons and have more neurons for the class of interest to help the network learn as much representation as possible from the object of interest.

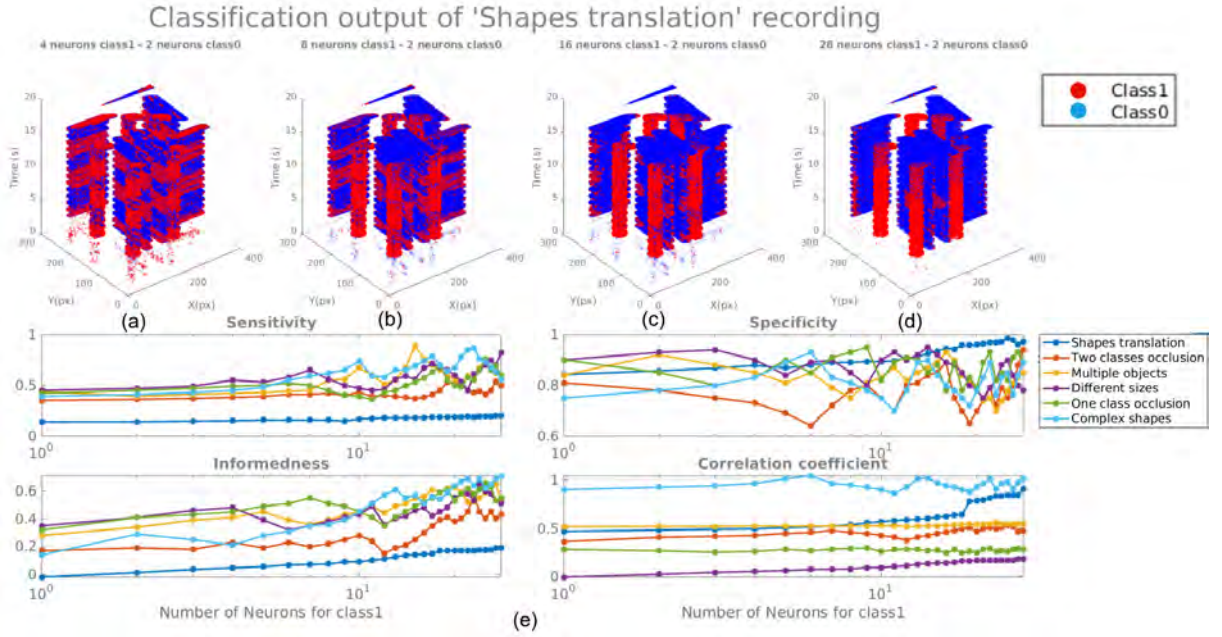


Figure 4.38: Feature selection investigation on the lab recorded datasets. (a), (b), (c) and (d) show the spatio-temporal pattern from the classification output where the number of neurons for class1 is increased from 4 to 28 neurons. (e) provides a summary of the model performance for all other laboratory recorded test sequences.

4.5.7 Comparing Performance with a Noise Filter

As shown in Figure 4.16, a simple background activity filter was used and applied for each incoming event as described in Section 4.4.8. The filter checks whether one of the eight neighbouring pixels has had an event within the last temporal window μ in microseconds. If not, the event is considered as noise and removed. The output of the noise filter consists of a noiseless events stream with less false positive that primarily

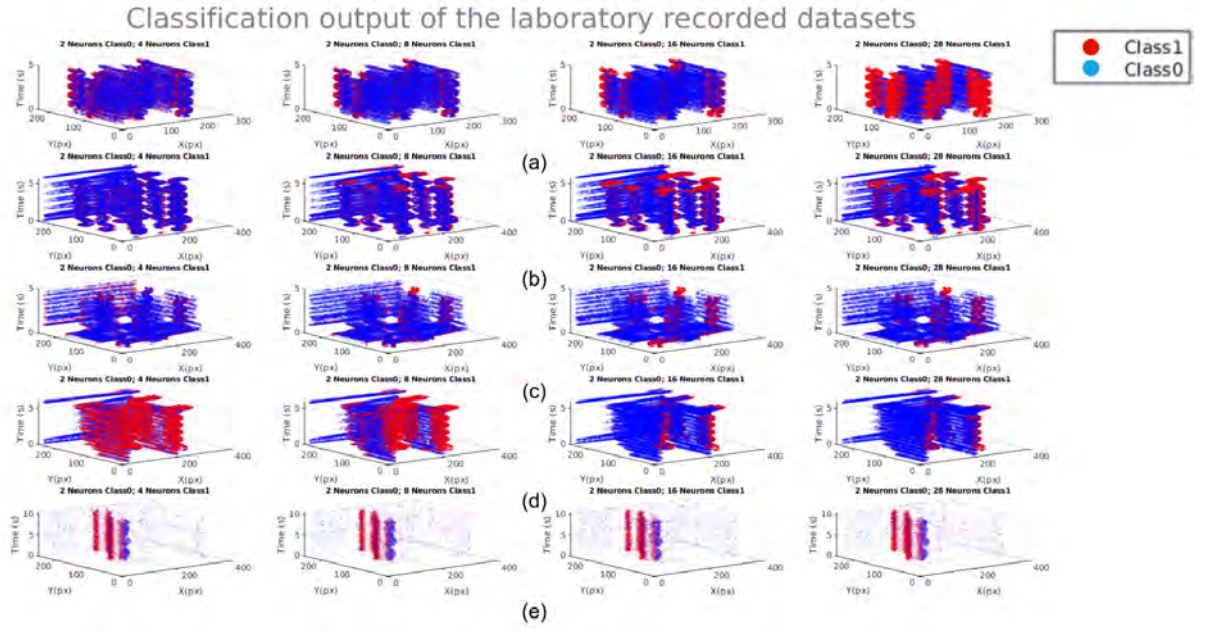


Figure 4.39: Feature selection investigation across the laboratory datasets. (a), (b), (c) and (d) show the spatio-temporal pattern from the classification output for all laboratory recorded test sequences. (a) Two classes occlusion, (b) Multiple objects, (c) Different sizes, (d) One class occlusion, (e) Complex shapes.

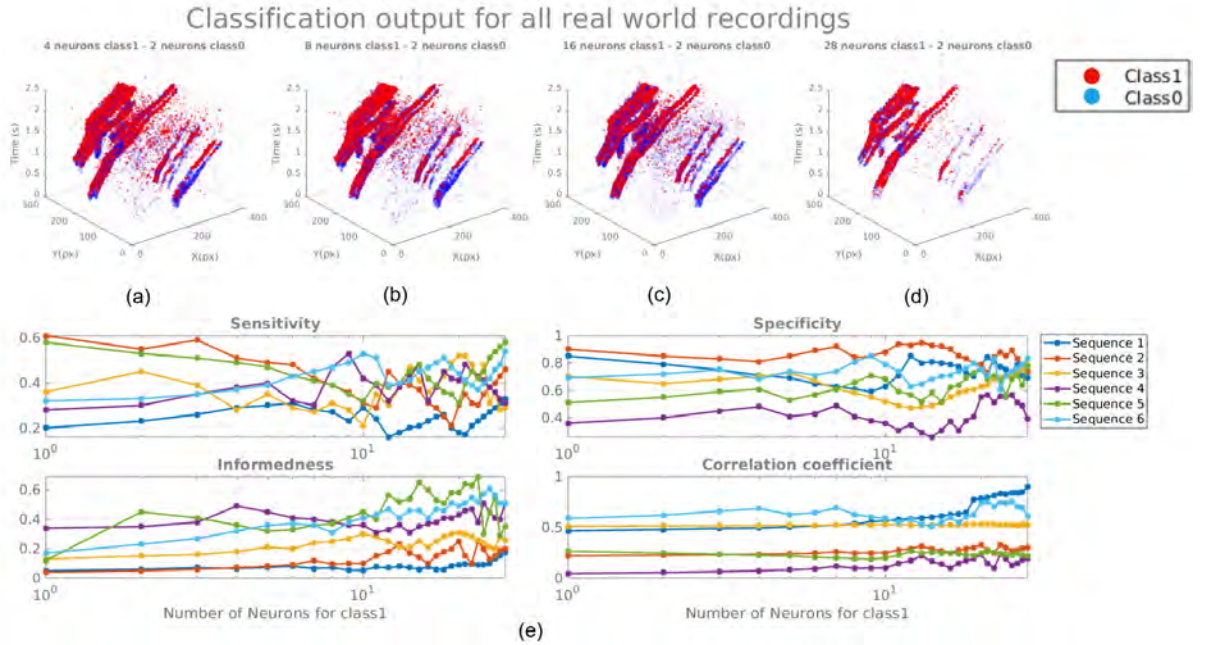


Figure 4.40: Feature selection results on the real world datasets. (a), (b), (c) and (d) show the spatio-temporal pattern from the classification output where the number of neurons for class1 is increased from 4 to 28 neurons. (e) provides a summary of the model performance for all other real world recorded test sequences.

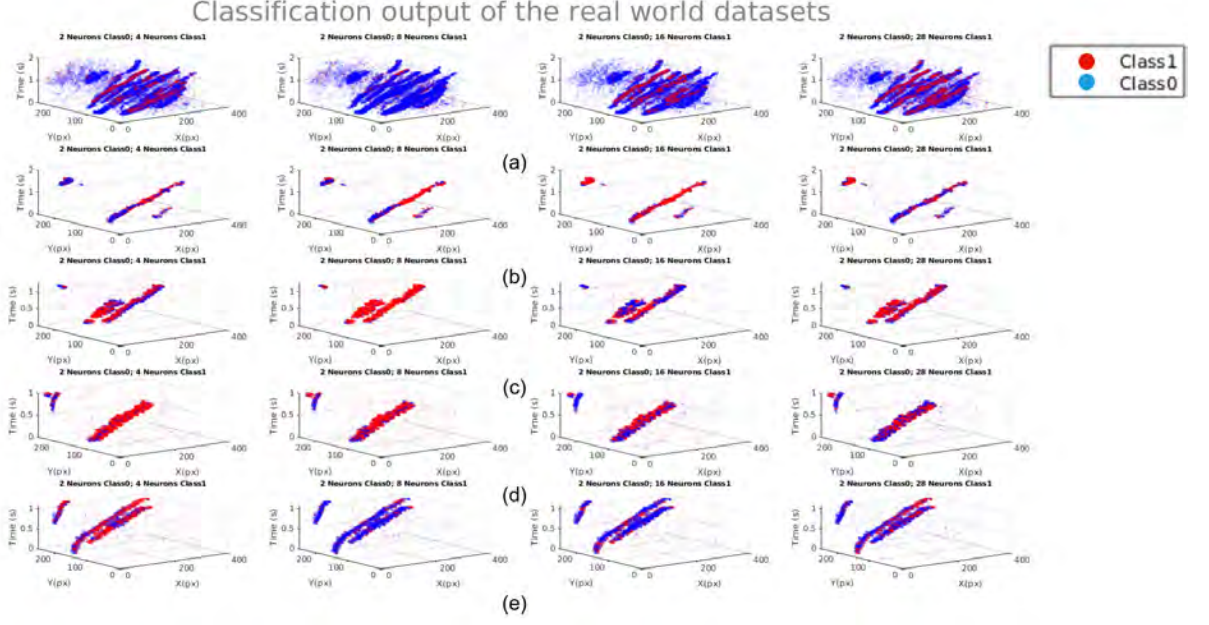
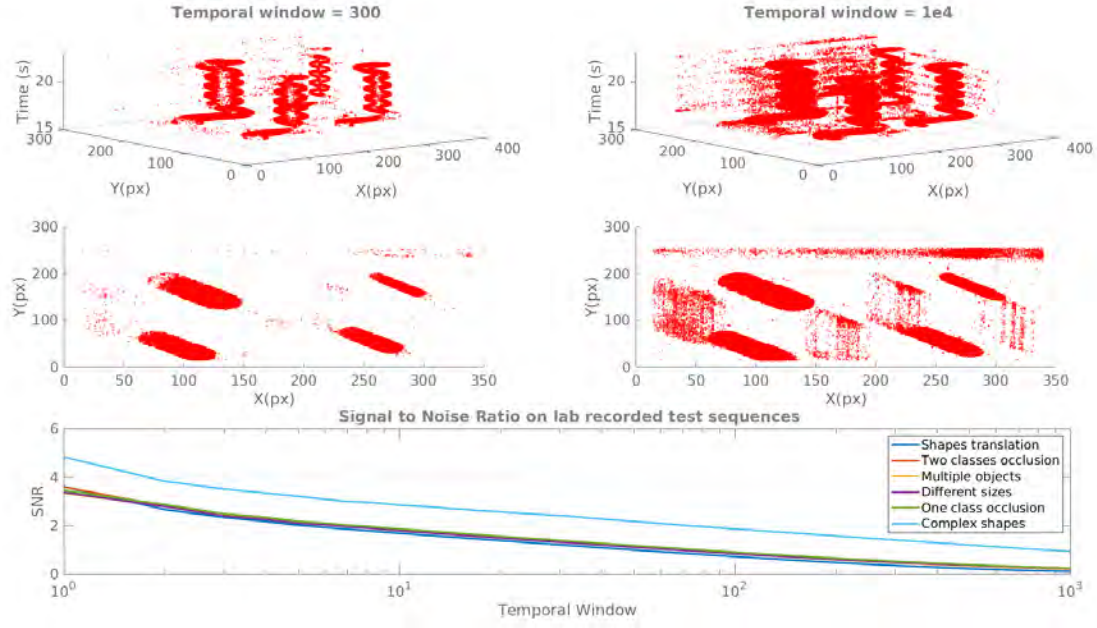
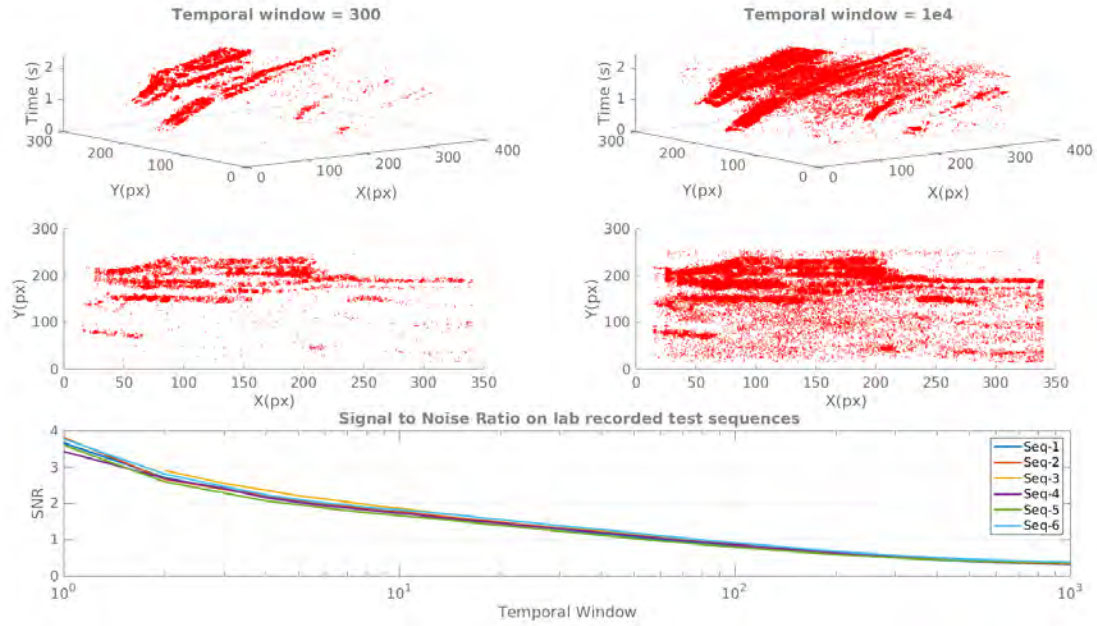


Figure 4.41: Feature selection investigation across the laboratory datasets. (a), (b), (c) and (d) show the spatio-temporal pattern from the classification output for all real world recorded test sequences.

depends on the value of μ . In order to quantify the noise filtering performance using our event-based classification algorithm, we used the SNR as described in Section 3.5.1 as the primary evaluation metrics where the μ is the only variable used. The results of the noise filter are illustrated in Figure 4.42. We noticed that by using a smaller temporal window such as $\mu \leq 300ms$, the noise filter removes most of the surrounding noises resulting in a high SNR signal. For example, in Figure 4.42(a), it was clear that the miss-classified events that belong to the line class were removed using the noise filter, and the events that belong to the circle class were kept, which shows the superiority of applying a noise filter after classification. It was also apparent that the μ and the SNR are inversely proportional to each other. The noise filter shows the same behaviour using real-world scene data as shown in Figure 4.42(b), which is considered noisier with more complex non-uniform patterns. With a $\mu \leq 300ms$ the SNR was higher, removing a large portion of the noise events and resulting in higher signal retention for the object of interest.



(a)



(b)

Figure 4.42: The effect of using different sizes of the temporal window after the classification output removes the false positive. (a) shows the results using the laboratory recorded sequence and (b) shows the results using the real-world recorded sequence. The output events representing class1 are made less noisy by applying a small temporal window. **Top panel:** A dimetric projection of the classification after applying the noise filter on the output events stream with different temporal windows. **Bottom panel:** The SNR results for different temporal windows starting from 1ms to 1s.

4.5.8 Classification Performance with GLS

As described in Section 4.4.9, Ω is the main parameter to optimise to make the data more linearly separable and make the classifier converge. Figure 4.43 shows the classification output using different values of Ω and the linear/rank correlation between the ground truth labels and the predicted labels. We found that for a low Ω , the linear classifier performs poorly on the test set resulting in a correlation coefficient of 0.071, and as the value increases, the correlation also increases. However, the relationship between Ω and the correlation was not linear because at $\Omega \geq 9$, the correlation started to decrease again. Note that the closer the Ω to the ratio between class1 and class0, the better the results. The detailed results for all recordings in the dataset are summarised in Figure 4.44. Each row shows the classification output in a space-time plot for an individual test set. Each data point represents the activated (ON and OFF polarities) pixels belonging to class1. When there are more objects in the field of view, the value of Ω needs to be higher and vice versa.

Based on the results, GLS is considered an essential and crucial pre-processing method for highly imbalanced datasets. It served to help to mitigate the effects of heteroskedasticity and skewness in the data. These results were insightful, as they demonstrated how important to add an imbalanced bias to the non-dominant class. However, one must choose the correct Ω value depending on the ratio between the events, which is required to be optimised and learned.

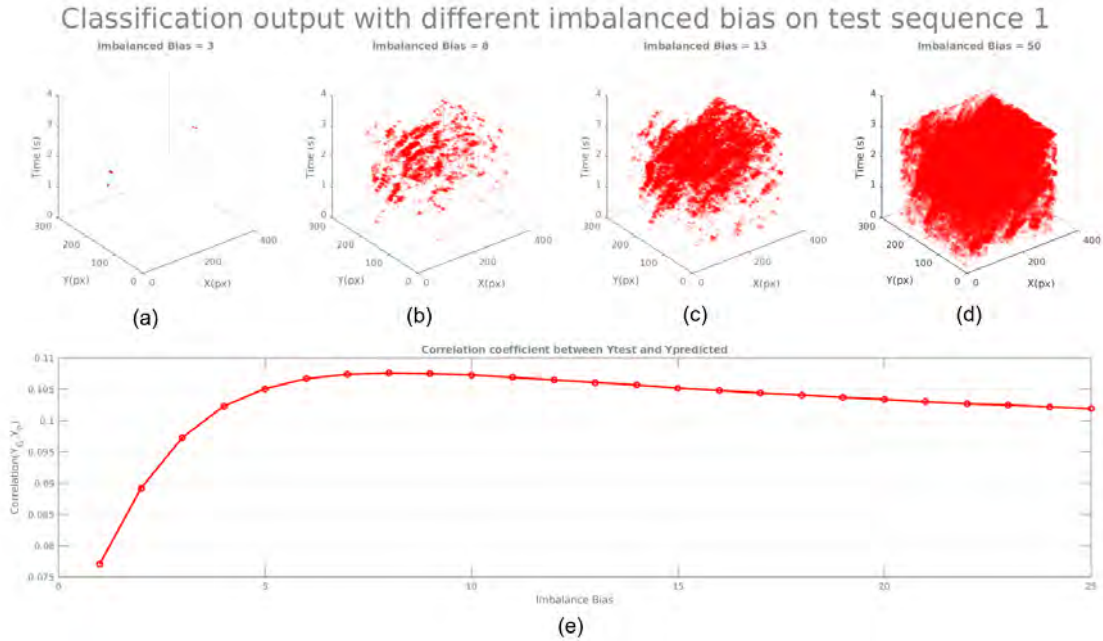


Figure 4.43: Classification output results after applying GLS. (a), (b), (c) and (d) show output spatio-temporal pattern of the classifier with different imbalanced bias value. (e) shows the results of the correlation coefficient with different imbalanced bias.

Classification output with different imbalanced bias on all test sequences

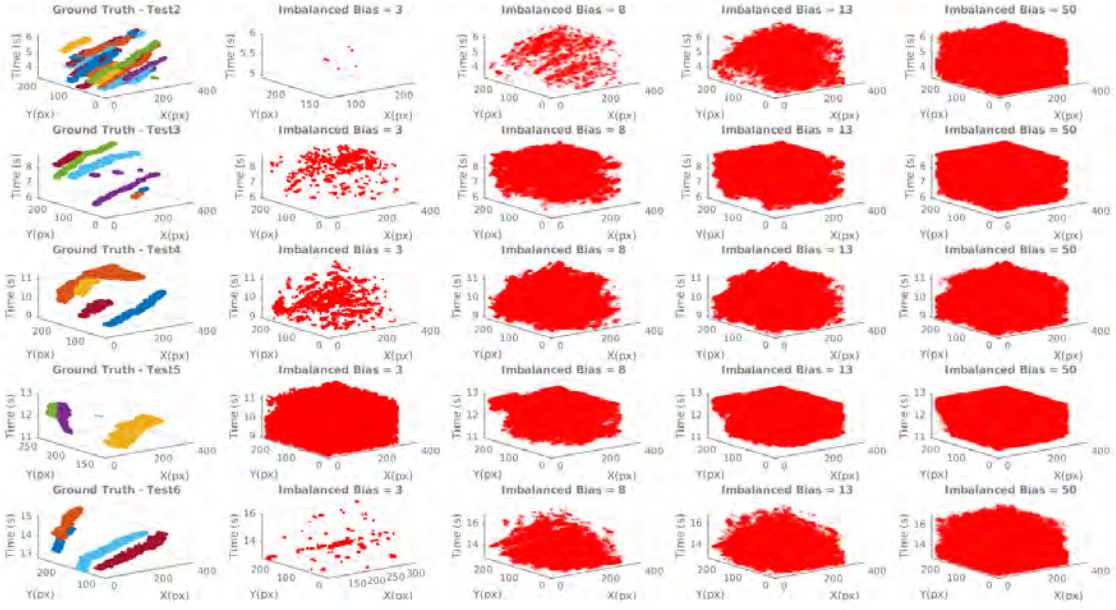


Figure 4.44: Classification output results after applying GLS on real world test sequences.

4.6 Discussion and Future Work

The results presented in this section determine that the primary challenge is not only the processing of complex environments but also the ability to extract relevant features with a low SNR and with high variance in the event rates. An algorithm needs to handle a large amount of unstructured data in a feature-rich scene with significant variations in the internal sensor properties such as different camera biases and circuit types. In this context, all the events are triggered due to the high-frequency movement of the sensor in front of a static scene, making it impossible to eliminate the event using a high pass or low pass filter. In addition, while the sensor is moving, the field of view continuously changes, altering the previously learned features and forgetting the old representations leading to catastrophic forgetting Parisi et al. [2019], which is a common problem in computer vision. While deep learning shows success in this particular domain, it requires an enormous amount of data and computational resources, removing the main vital properties of the sensor as being low-power, high-bandwidth etc. The entire system presented in this section operates in an event-driven paradigm and retains the dense event-based representation generated by the DAVIS sensor. The system is tested and validated on lab recording data and real-world scenes with different conditions. The purpose of the first datasets is to test the system on simpler challenging tasks such as object occlusion, different object size, different orientation etc. to understand the system behaviour against each challenge, while the goal of the real-world datasets was to put the sensor and the system under extreme conditions and evaluate the network behaviour. While the EBC is an efficient sensor to detect saliency, in our case, the entire

scene becomes salient due to the camera motion. The problem is not only to detect salient objects but also to have an efficient adaptive filter that removes the unwanted events and adequately differentiate between the background and the object of interest. This will ensure that such event-based datasets will be noisy and need robust detection and classifier algorithms.

One of the critical aspects of the algorithm is the use of supervisory signals in feature extraction. While FEAST is an efficient feature extractor algorithms, it becomes difficult to construct representation for the object of interest in the very dense and complex scene. For that reason, supervisory enabled feature selection, allocated a specific number of neurons for the class of interest and boosted the network’s performance. However, the feature selection takes the number of neurons for each class as an input parameter depending on the event ratio between the classes. We aim to investigate optimisation methods to learn the number of neurons and weigh them according to any input data in future work.

One important hyperparameter was the downsampling factor that removes the need for applying conventional downsampling methods. This serves in mapping a region of the input to a single channel, effectively creating a small receptive field for each input to the classifier network. Given the event-driven behaviour of the network and the sheer amount of data in each recording, it becomes computationally impossible to pass all the events to the input layer of the classifier locally, creating a computational bottleneck. For that reason, a full sweep of the downsampling factors was tested on the data to find the amount of data that can be removed without harming the network’s performance. Fortunately, the downsampling method has been shown to maintain network performance even with a significant downsampling factor. On the other hand, we found that the classifier performance peaks when the downsampling factor matches the object’s size in the scene. In future work, we aim to investigate the downsampling method when the objects exhibit different sizes by having an ensemble of networks each operate using different downsampling factors that match a specific object size.

In this work, the noise events were examined before and after classification. A portion of them was efficiently learned and eliminated by FEAST, and the rest of the noise events that pass through the classifier network were removed using a noise filter that looks at the active vs non-active pixels within a specified temporal window. This lead to less false positive events. The noise filter is a crucial element in the network pipeline.

Another essential technique is applying GLS before classification. As shown in the result section, by replacing the one-hot encoding method with the GLS, the skewness of the data is reduced, which makes the data more linearly separable for the classifier. The proxy signal Ω was used to estimate the value of class1 labels. This eliminates conventional methods such as undersampling and oversampling input data for each class and solving the problem statistically while keeping all the information in the loop. The implementation of this method can benefit from algorithmic optimisation, but they are presented here in their simplest form as they constitute the basis of the work presented

in the succeeding chapter.

The algorithms were tested on four different tasks: (i) multi-object detections of the same instance (i.e. laboratory dataset), (ii) fruit detection datasets, (iii) digits classification using N-MNIST and (vi) human activity recognition using DVS gestures dataset. However, we did not make a direct comparison based on the model performance across all the datasets because the classification task in the fruit detection data significantly differs from N-MNIST data and DVS gesture. For instance, in the fruit detection data, there are always multiple instances of fruits within the same field of view in front of noisy scenes, and the algorithm has to precisely detect the multiple instances of the same object, whereas, in the N-MNIST, the goal was to classify a single object per recording and the same also applies for the DVS gesture data. For that reason, the model performance was evaluated based on the task individually, and each dataset was considered a single-use case for the algorithm and separated into different sections.

The end goal of this work is to deploy the model on a low power automated robotics platform to detect and count fruits continuously throughout the season. However, it is important to acknowledge that the agricultural application as a use case using the event camera has several drawbacks. For instance, there is little need to high speed reactions or closed loop control, and the application is not specifically power constrained by the sensor and the processor, therefore, this limits the suitability of the EBC in this specific area.

An important future work is to investigate the use of events to frames generation method and use a CNN-based pre-trained model to investigate the benefits of the precise timing carried by the events data. Another important future work direction for this work is to build a large scale dataset from real-world scenes which take advantage of the environment’s geometry and the continuous streams of information. As the results of this work demonstrate the potential of such systems, the practical and real-time implementation of the components also forms a portion of the future directions for this work. The implementation of the feature detector FEAST Afshar et al. [2019c] algorithms open the door to explore many other fundamental research ideas such as (1) continual lifelong learning [Parisi et al., 2019] as well as long-term memory consolidation and retrieval, which gives the system the ability to continually learn over time by accommodating new knowledge while retaining previously learned experiences and potentially overcome the problem of catastrophic forgetting or catastrophic interference. (2) Explore an approach to adaptively learn the sensor bias in environments where depth, light, size of the object continuously change, that way, it can overcome the problem of manual focusing the sensor and potentially reduce the data rate. (3) Explore the possibilities of using a closed-loop system where the human is included in the loop against the system. The system can be evaluated based on its accuracy and performance against humans.

4.7 Conclusion

This work presents an event-based detector and classifier network for agricultural applications to detect and classify fruits. It addresses the problem of detecting objects in cluttered scenes with highly imbalanced classes and shows the challenges and limitations of performing such tasks. The labelled datasets provided a test bench for investigating event-based algorithms for unique and challenging complex scenes. All the recorded datasets were carefully labelled, providing analytically defined ground truth. The labelling procedure provided a highly accurate label set across a wide range of environments.

Statistical measures were used to evaluate the classification performance based on event density activated Spatio-temporal volume slices such as sensitivity, specificity, informedness and correlation coefficient. This facilitates the comparison of the architecture against the raw events stream at each processing stage and provides valuable insights into the dataset's properties.

Several event-based architectures were tested on both types of data where different types of complexity were introduced. The algorithms include an event-based feature detector with and without the supervisory signals that enabled feature selection techniques with an optimised and iterative classification algorithm.

The algorithms were evaluated in terms of their output statistical properties of the data by measuring an optimised proxy measure over the downsampling factor size, feature class distribution and the GLS offset size, which showed to provide superior performance on a very challenging noisy and complex dataset. The same detection-classification architecture was used on the N-MNIST and DVS gesture datasets to evaluate the model performance against different types of events input. In this case, the algorithm proposed in this thesis demonstrated its capabilities in classifying multi-class objects, such as in N-MNIST and efficiently classifying complex human gestures.

Chapter 5

CONCLUSION

5.1 Applicability of Colour Event-based Dataset

A large body of literature concentrates on colour event-based sensor design without providing information about the colour event internal properties and their behaviours. This thesis explores the behaviours of colour event-based vision data and their potential applications. We used the CDAVIS sensor, which features high temporal resolution colour events inspired by precise-timing biological models. This sensor has a Bayer filter that can detect colour variations between two surfaces. Given that the colour DVS pixels only send spikes data, the complete colour information can only be obtained using either the APS readout or events-to-frames reconstruction techniques, which are considered computationally demanding. Based on our research questions in Section 3.2 we have found that it is challenging to estimate the true colour of the object purely based on the output of the event spikes, that is because the spike output output from each colour filter does not correlate with the true colour of the object. That is because the behaviour is influenced by two factors: the colour of the observed object and the background colour. The lack of universal approaches to characterising colour events was the primary motivation for this work, as the current characterisation approaches are made for the monochromatic output of the EBC and cannot be directly applicable to the colour events. The CDAVIS has been used in imaging of neural activities [Moeys et al., 2017] by reconstructing images with HDR from the DVS colour pixels eliminating the need of expensive CMOS sensors. It is also applicable when fast transitions between colours need to be detected in automated industrial processes or systems that need to detect transitions between radiation in the near-infrared band and the visible spectrum [Farian et al., 2015, Lenero-Bardallo et al., 2013]. It can also eliminate spatial redundancy and noise in boisterous scenes and recover the border between two overlapping objects.

Furthermore, the HDR of the DVS pixels tackles the luminance adaptation issue, which can be helpful in applications such as self-driving cars or robot grasping. In this work, we identified several areas of improvement for the sensor that would benefit future algorithm’s implementation. These improvements require hardware development. For example, one of the CDAVIS weaknesses is its lack of sensitivity to short wavelengths

(e.g. blue wavelength). Most silicon-based photodetectors share this limitation, but it is also aggravated by the DAVIS low sensitivity to low light. An increase in sensor sensitivity and a decrease in sensor mismatch will potentially lead to better results. Our characteristic experiment provides more understanding of the behaviour of colour events under different conditions using different camera biases such as threshold and pixel bandwidth. These are crucial for future data collection, algorithms, and processing techniques that take advantage of colour events.

5.2 Viability of Event-Based Object Classification

The work in this thesis deals primarily with the classification of objects from the event-based sensor in complex and dense scenes in the domain of fruits detection. This work explores two primary means to perform such classification. To address the research questions in Section 4.2, in Chapter 4 we present three approaches that operate on the event-streams directly in order to perform classification, maintaining both the spatial and temporal information directly throughout the network pipeline. The network described extracts discriminative features from the event-based data to perform the classification task. Two feature extraction and classification methods were explored in this work, with and without the supervisory signals as input to the network, providing a detailed description and analysis of the nature of these features and methods to assess the quality and usefulness of such features. One of the essential aspects of this network is the ability to remove noise and features that have less contribution to the classification system, and the analysis of the feature detectors includes details on their feature selectivity abilities. This is particularly relevant as learning relevant features in very complex scenes with various cluttered objects is crucial. We found that include supervisory signals during training is beneficial for the network to perform better during inference, this method has significantly outperform unsupervised method.

The results from these approaches demonstrate the viability of an event-based classification using the DAVIS sensor. The event-based nature of most of the systems presented in this work maintains the data-driven paradigm of event-based vision and computation. Although dense real-world scenes heavily impact the system performance, dense scenes remain an incredibly complex task to be solved using an event-based due to the dense temporal structure. This work serves as a framework and investigation into the applicability and efficacy of event-based classification.

5.3 Future Work

Moving forward, we will investigate the many open lines of inquiry indicated in each chapter. The most immediate next step would be creating a large scale agricultural dataset and allow the network to learn all the possible variations in the dataset. Increasing the depth of the network combining multiple feature extraction layers to account for more complex shapes. Integrating the proposed methods into physical robotics systems

is also considered a next step. While the benefit of these methods has been demonstrated on datasets offline, event cameras have yet to be tightly integrated into real-time robotics tasks, such as closed-loop control and model predictive control (MPC). In terms of learning methods, temporal consistency appears to be a crucial factor in robust operation for event cameras in the real world, as the cameras respond only to changes in the scene, the most common way to perceive static parts of the scene is through a memory of the past. For example, when the camera is scanning through a farm to detect and count fruits, it will continuously generate many events. As a result, an algorithm needs to process many events, learn robust features, and continually retain the information that belongs to the fruits during motion. For that reason, an algorithm has to account for different camera motions to improve the detection performance.

Since the fruit detection task is usually performed in a dense and complex environment, as future work, one can transform the dense input stream into a much sparser representations and use SNN algorithms as an alternative paradigm for learning, which show great promise in taking full advantage of the sparse, asynchronous output from event cameras. SNNs are more biologically inspired. The update of the activation at layer is performed asynchronously and in an event-driven fashion, which results in significantly lower bandwidth for sparse representations. By implementing SNN, one can have a fully asynchronous pipeline that goes from sensing to perception, which takes advantage of the low latency, high dynamic range as well as low power advantages of event cameras. However, the big question remains in converting dense data to sparse representation without losing the input stream’s primary information. Developing a stable SNNs is still an open problem.

Furthermore, more work need to be devoted to produces high quality colour event datasets and focus on areas where colour contrast is a more relevant feature than shape and appearance. Secondly, characterising the colour sensor should focus on its colour responses using colour-based measurements. However, the colour event sensors prototype have been unavailable until lately, which has limited access to colour datasets. Moreover, event camera’s resolution has been significantly lower than that of traditional cameras. As more colour prototypes are now available and more companies working on these types of sensors and developing hardware, it is expected that the price will come down and the resolution will go up, resulting in wide adoption of colour events cameras and more application-specific datasets for real-world applications.

References

- VIII. A dynamical theory of the electromagnetic field. *Philosophical Transactions of the Royal Society of London*, 155:459–512, Dec. 1865. ISSN 0261-0523, 2053-9223. doi: 10.1098/rstl.1865.0008. URL <https://royalsocietypublishing.org/doi/10.1098/rstl.1865.0008>.
- Advances in Photoreception: Proceedings of a Symposium on Frontiers of Visual Science*. National Academies Press, Washington, D.C., Jan. 1990. ISBN 978-0-309-04240-6. doi: 10.17226/1570. URL <http://www.nap.edu/catalog/1570>. Pages: 1570.
2021. URL <https://inivation.com/wp-content/uploads/2020/05/White-Paper-May-2020.pdf>.
- S. Afshar. High speed event-based visual processing in the presence of noise. 2020.
- S. Afshar, T. J. Hamilton, J. Tapson, A. van Schaik, and G. Cohen. Investigation of event-based surfaces for high-speed detection, unsupervised feature extraction, and object recognition. *Frontiers in Neuroscience*, 12:1047, 2019a. ISSN 1662-453X. doi: 10.3389/fnins.2018.01047. URL <https://www.frontiersin.org/article/10.3389/fnins.2018.01047>.
- S. Afshar, A. P. Nicholson, A. van Schaik, and G. Cohen. Event-based Object Detection and Tracking for Space Situational Awareness. *arXiv:1911.08730 [cs]*, Nov. 2019b. URL <http://arxiv.org/abs/1911.08730>. arXiv: 1911.08730.
- S. Afshar, Y. Xu, J. Tapson, A. van Schaik, and G. Cohen. Event-based Feature Extraction Using Adaptive Selection Thresholds. pages 1–15, 2019c. URL <http://arxiv.org/abs/1907.07853>.
- T. Ahonen, A. Hadid, and M. Pietikainen. Face Description with Local Binary Patterns: Application to Face Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(12):2037–2041, Dec. 2006. ISSN 0162-8828, 2160-9292. doi: 10.1109/TPAMI.2006.244. URL <http://ieeexplore.ieee.org/document/1717463/>.
- A. C. Aitken. Iv.—on least squares and linear combination of observations. *Proceedings of the Royal Society of Edinburgh*, 55:42–48, 1936. doi: 10.1017/S0370164600014346.
- S. Aksoy and R. M. Haralick. Feature normalization and likelihood-based similarity measures for image retrieval. *Pattern Recogn. Lett.*, 22(5):563–582, Apr. 2001. ISSN

- 0167-8655. doi: 10.1016/S0167-8655(00)00112-4. URL [https://doi.org/10.1016/S0167-8655\(00\)00112-4](https://doi.org/10.1016/S0167-8655(00)00112-4).
- I. Alonso and A. C. Murillo. EV-SegNet: Semantic Segmentation for Event-based Cameras. *arXiv:1811.12039 [cs]*, Nov. 2018. URL <http://arxiv.org/abs/1811.12039>. arXiv: 1811.12039.
- A. Amir, B. Taba, D. Berg, T. Melano, J. McKinstry, C. Di Nolfo, T. Nayak, A. Andreopoulos, G. Garreau, M. Mendoza, J. Kusnitz, M. Debole, S. Esser, T. Delbruck, M. Flickner, and D. Modha. A Low Power, Fully Event-Based Gesture Recognition System. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7388–7397, Honolulu, HI, July 2017. IEEE. ISBN 978-1-5386-0457-1. doi: 10.1109/CVPR.2017.781. URL <https://ieeexplore.ieee.org/document/8100264/>.
- A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning distance functions using equivalence relations. In *Proceedings of the Twentieth International Conference on International Conference on Machine Learning, ICML’03*, page 11–18. AAAI Press, 2003. ISBN 1577351894.
- M. Barbaro, P.-Y. Burgi, A. Mortara, P. Nussbaum, and F. Heitger. A 100×100 pixel silicon retina for gradient extraction with steering filter capabilities and temporal output coding. 37(2):160–172. ISSN 00189200. doi: 10.1109/4.982422. URL <http://ieeexplore.ieee.org/document/982422/>.
- P. Bardow, A. J. Davison, and S. Leutenegger. Simultaneous Optical Flow and Intensity Estimation from an Event Camera. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 884–892, Las Vegas, NV, USA, June 2016. IEEE. ISBN 978-1-4673-8851-1. doi: 10.1109/CVPR.2016.102. URL <http://ieeexplore.ieee.org/document/7780471/>.
- S. Bargoti and J. Underwood. Deep Fruit Detection in Orchards. *arXiv:1610.03677 [cs]*, Sept. 2017. URL <http://arxiv.org/abs/1610.03677>. arXiv: 1610.03677.
- R. Berner and T. Delbruck. Event-based color change pixel in standard CMOS. In *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, pages 349–352, Paris, France, May 2010. IEEE. ISBN 978-1-4244-5308-5. doi: 10.1109/ISCAS.2010.5537787. URL <http://ieeexplore.ieee.org/document/5537787/>.
- R. Berner and T. Delbruck. Event-Based Pixel Sensitive to Changes of Color and Brightness. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 58(7): 1581–1590, July 2011. ISSN 1549-8328, 1558-0806. doi: 10.1109/TCSI.2011.2157770. URL <http://ieeexplore.ieee.org/document/5892910/>.
- R. Berner, P. Lichtsteiner, and T. Delbruck. Self-timed vertacolor dichromatic vision sensor for low power pattern detection. In *2008 IEEE International Symposium on Circuits and Systems*, pages 1032–1035, Seattle, WA, USA, May 2008. IEEE. ISBN

- 978-1-4244-1683-7. doi: 10.1109/ISCAS.2008.4541597. URL <http://ieeexplore.ieee.org/document/4541597/>.
- A. L. Blum and P. Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97(1-2):245–271, Dec. 1997. ISSN 00043702. doi: 10.1016/S0004-3702(97)00063-5. URL <https://linkinghub.elsevier.com/retrieve/pii/S0004370297000635>.
- A. Botchkarev. Performance Metrics (Error Measures) in Machine Learning Regression, Forecasting and Prognostics: Properties and Typology. *Interdisciplinary Journal of Information, Knowledge, and Management*, 14:045–076, 2019. ISSN 1555-1229, 1555-1237. doi: 10.28945/4184. URL <http://arxiv.org/abs/1809.03006>. arXiv: 1809.03006.
- C. Brandli, R. Berner, Minhao Yang, Shih-Chii Liu, and T. Delbruck. A 240×180 130 dB 3 μ s Latency Global Shutter Spatiotemporal Vision Sensor. *IEEE J. Solid-State Circuits*, 49(10):2333–2341, Oct. 2014. ISSN 0018-9200, 1558-173X. doi: 10.1109/JSSC.2014.2342715. URL <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6889103>.
- C. Breuil, B. J. Jennings, S. Barthelmé, N. Guyader, and F. A. A. Kingdom. Color improves edge classification in human vision. *PLOS Computational Biology*, 15(10): e1007398, Oct. 2019. ISSN 1553-7358. doi: 10.1371/journal.pcbi.1007398. URL <https://dx.plos.org/10.1371/journal.pcbi.1007398>.
- V. Buhrmester, D. Münch, D. Bulatov, and M. Arens. Evaluating the Impact of Color Information in Deep Neural Networks. In A. Morales, J. Fierrez, J. S. Sánchez, and B. Ribeiro, editors, *Pattern Recognition and Image Analysis*, volume 11867, pages 302–316. Springer International Publishing, Cham, 2019. ISBN 978-3-030-31331-9 978-3-030-31332-6. doi: 10.1007/978-3-030-31332-6_27. URL http://link.springer.com/10.1007/978-3-030-31332-6_27. Series Title: Lecture Notes in Computer Science.
- L. Camunas-Mesa, C. Zamarreno-Ramos, A. Linares-Barranco, A. J. Acosta-Jimenez, T. Serrano-Gotarredona, and B. Linares-Barranco. An Event-Driven Multi-Kernel Convolution Processor Module for Event-Driven Vision Sensors. *IEEE Journal of Solid-State Circuits*, 47(2):504–517, Feb. 2012. ISSN 0018-9200, 1558-173X. doi: 10.1109/JSSC.2011.2167409. URL <http://ieeexplore.ieee.org/document/6054033/>.
- M. Cannici, M. Ciccone, A. Romanoni, and M. Matteucci. Asynchronous Convolutional Networks for Object Detection in Neuromorphic Cameras. *arXiv:1805.07931 [cs]*, June 2019. URL <http://arxiv.org/abs/1805.07931>. arXiv: 1805.07931.
- T.-J. Chin, S. Bagchi, A. Eriksson, and A. van Schaik. Star Tracking using an Event Camera. *arXiv:1812.02895 [cs]*, Apr. 2019. URL <http://arxiv.org/abs/1812.02895>. arXiv: 1812.02895.

- G. Cohen, S. Afshar, G. Orchard, J. Tapson, R. Benosman, and A. van Schaik. Spatial and Temporal Downsampling in Event-Based Visual Classification. *IEEE Transactions on Neural Networks and Learning Systems*, 29(10):5030–5044, Oct. 2018. ISSN 2162-237X, 2162-2388. doi: 10.1109/TNNLS.2017.2785272. URL <https://ieeexplore.ieee.org/document/8260859/>.
- 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, June 2019. ISSN 0021-9142, 2195-0571. doi: 10.1007/s40295-018-00140-5. URL <http://link.springer.com/10.1007/s40295-018-00140-5>.
- 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, San Jose, CA, USA, July 2011. IEEE. ISBN 978-1-4244-9635-8. doi: 10.1109/IJCNN.2011.6033299. URL <http://ieeexplore.ieee.org/document/6033299/>.
- D. Czech and G. Orchard. Evaluating noise filtering for event-based asynchronous change detection image sensors. In *2016 6th IEEE International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, pages 19–24, Singapore, Singapore, June 2016. IEEE. ISBN 978-1-5090-3287-7. doi: 10.1109/BIOROB.2016.7523452. URL <http://ieeexplore.ieee.org/document/7523452/>.
- N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*, volume 1, pages 886–893, San Diego, CA, USA, 2005. IEEE. ISBN 978-0-7695-2372-9. doi: 10.1109/CVPR.2005.177. URL <http://ieeexplore.ieee.org/document/1467360/>.
- 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, 2013. ISSN 1662-453X. doi: 10.3389/fnins.2013.00223. URL <http://journal.frontiersin.org/article/10.3389/fnins.2013.00223/abstract>.
- 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, New Orleans, LA, USA, May 2007. IEEE. ISBN 978-1-4244-0920-4 978-1-4244-0921-1. doi: 10.1109/ISCAS.2007.378038. URL <http://ieeexplore.ieee.org/document/4252767/>.
- T. Delbruck, B. Linares-Barranco, E. Culurciello, and C. Posch. Activity-driven, event-based vision sensors. In *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, pages 2426–2429, Paris, France, May 2010. IEEE. ISBN 978-1-4244-5308-5. doi: 10.1109/ISCAS.2010.5537149. URL <http://ieeexplore.ieee.org/document/5537149/>.

- P. A. Dias, A. Tabb, and H. Medeiros. Apple flower detection using deep convolutional networks. *Computers in Industry*, 99:17–28, Aug. 2018. ISSN 01663615. doi: 10.1016/j.compind.2018.03.010. URL <https://linkinghub.elsevier.com/retrieve/pii/S016636151730502X>.
- D. Drazen, P. Lichtsteiner, P. Häfliger, T. Delbrück, and A. Jensen. Toward real-time particle tracking using an event-based dynamic vision sensor. *Experiments in Fluids*, 51(5):1465–1469, Nov. 2011. ISSN 0723-4864, 1432-1114. doi: 10.1007/s00348-011-1207-y. URL <http://link.springer.com/10.1007/s00348-011-1207-y>.
- M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, Jan. 2015a.
- M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The Pascal Visual Object Classes Challenge: A Retrospective. *International Journal of Computer Vision*, 111(1):98–136, Jan. 2015b. ISSN 0920-5691, 1573-1405. doi: 10.1007/s11263-014-0733-5. URL <http://link.springer.com/10.1007/s11263-014-0733-5>.
- J. Fan and R. Li. Variable Selection via Nonconcave Penalized Likelihood and its Oracle Properties. *Journal of the American Statistical Association*, 96(456):1348–1360, Dec. 2001. ISSN 0162-1459, 1537-274X. doi: 10.1198/016214501753382273. URL <http://www.tandfonline.com/doi/abs/10.1198/016214501753382273>.
- L. Farian, J. A. Lenero-Bardallo, and P. Häfliger. A Bio-Inspired AER Temporal Tri-Color Differentiator Pixel Array. *IEEE Transactions on Biomedical Circuits and Systems*, 9(5):686–698, Oct. 2015. ISSN 1932-4545, 1940-9990. doi: 10.1109/TBCAS.2015.2492460. URL <http://ieeexplore.ieee.org/document/7313036/>.
- P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multi-scale, deformable part model. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, Anchorage, AK, USA, June 2008. IEEE. ISBN 978-1-4244-2242-5. doi: 10.1109/CVPR.2008.4587597. URL <http://ieeexplore.ieee.org/document/4587597/>.
- R. A. Fisher. THE STATISTICAL UTILIZATION OF MULTIPLE MEASUREMENTS. *Annals of Eugenics*, 8(4):376–386, Aug. 1938. ISSN 20501420. doi: 10.1111/j.1469-1809.1938.tb02189.x. URL <http://doi.wiley.com/10.1111/j.1469-1809.1938.tb02189.x>.
- C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg. DSSD : Deconvolutional Single Shot Detector. *arXiv:1701.06659 [cs]*, Jan. 2017. URL <http://arxiv.org/abs/1701.06659>. arXiv: 1701.06659.

- Z. Fu and A. H. Titus. CMOS Neuromorphic Optical Sensor Chip With Color Change-Intensity Change Disambiguation (CCICD). *IEEE Sensors Journal*, 9(6): 689–696, June 2009. ISSN 1530-437X. doi: 10.1109/JSEN.2009.2020676. URL <http://ieeexplore.ieee.org/document/4909423/>.
- K. Fukushima, Y. Yamaguchi, M. Yasuda, and S. Nagata. An electronic model of the retina. *Proceedings of the IEEE*, 58(12):1950–1951, 1970. ISSN 0018-9219. doi: 10.1109/PROC.1970.8066. URL <http://ieeexplore.ieee.org/document/1449996/>.
- K. Fukushima, S. Miyake, and T. Ito. Neocognitron: A neural network model for a mechanism of visual pattern recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-13(5):826–834, Sept. 1983. ISSN 0018-9472, 2168-2909. doi: 10.1109/TSMC.1983.6313076. URL <http://ieeexplore.ieee.org/document/6313076/>.
- B. Funt and L. Zhu. Does Colour Really Matter? Evaluation via Object Classification. *Color and Imaging Conference*, 2018(1):268–271, Nov. 2018. ISSN 2166-9635. doi: 10.2352/ISSN.2169-2629.2018.26.268. URL <https://www.ingentaconnect.com/content/10.2352/ISSN.2169-2629.2018.26.268>.
- 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, Oct. 2018. ISSN 0162-8828, 2160-9292, 1939-3539. doi: 10.1109/TPAMI.2017.2769655. URL <https://ieeexplore.ieee.org/document/8094962/>.
- G. Gallego, T. Delbruck, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. Davison, J. Conradt, K. Daniilidis, and D. Scaramuzza. Event-based Vision: A Survey. *arXiv:1904.08405 [cs]*, Feb. 2020. URL <http://arxiv.org/abs/1904.08405>. arXiv: 1904.08405.
- R. Girshick. Fast R-CNN. *arXiv:1504.08083 [cs]*, Sept. 2015. URL <http://arxiv.org/abs/1504.08083>. arXiv: 1504.08083.
- R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *arXiv:1311.2524 [cs]*, Oct. 2014. URL <http://arxiv.org/abs/1311.2524>. arXiv: 1311.2524.
- 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, Daejeon, South Korea, Oct. 2016. IEEE. ISBN 978-1-5090-3762-9. doi: 10.1109/IROS.2016.7759345. URL <http://ieeexplore.ieee.org/document/7759345/>.
- A. Glover and C. Bartolozzi. Robust visual tracking with a freely-moving event camera. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems*

- (*IROS*), pages 3769–3776, Vancouver, BC, Sept. 2017. IEEE. ISBN 978-1-5386-2682-5. doi: 10.1109/IROS.2017.8206226. URL <http://ieeexplore.ieee.org/document/8206226/>.
- A. Gongal, S. Amatyia, M. Karkee, Q. Zhang, and K. Lewis. Sensors and systems for fruit detection and localization: A review. *Computers and Electronics in Agriculture*, 116:8–19, Aug. 2015. ISSN 01681699. doi: 10.1016/j.compag.2015.05.021. URL <https://linkinghub.elsevier.com/retrieve/pii/S0168169915001581>.
- P. Gouras. Color vision by peter gouras. URL <https://webvision.med.utah.edu/book/part-vii-color-vision/color-vision/>.
- I. Guyon, S. Gunn, M. Nikraves, and L. A. Zadeh. *Feature Extraction: Foundations and Applications (Studies in Fuzziness and Soft Computing)*. Springer-Verlag, Berlin, Heidelberg, 2006. ISBN 3540354875.
- J. G. Ha, H. Moon, J. T. Kwak, S. I. Hassan, M. Dang, O. N. Lee, and H. Y. Park. Deep convolutional neural network for classifying Fusarium wilt of radish from unmanned aerial vehicles. *Journal of Applied Remote Sensing*, 11(04):1, Dec. 2017. ISSN 1931-3195. doi: 10.1117/1.JRS.11.042621. URL <https://www.spiedigitallibrary.org/journals/journal-of-applied-remote-sensing/volume-11/issue-04/042621/Deep-convolutional-neural-network-for-classifying-Fusarium-wilt-of-radish/10.1117/1.JRS.11.042621.full>.
- T. Hansen and K. R. Gegenfurtner. Color contributes to object-contour perception in natural scenes. *Journal of Vision*, 17(3):14, Mar. 2017. ISSN 1534-7362. doi: 10.1167/17.3.14. URL <http://jov.arvojournals.org/article.aspx?doi=10.1167/17.3.14>.
- A. B. Hassanat, M. A. Abbadi, G. A. Altarawneh, and A. A. Alhasanat. Solving the Problem of the K Parameter in the KNN Classifier Using an Ensemble Learning Approach. *arXiv:1409.0919 [cs]*, Sept. 2014. URL <http://arxiv.org/abs/1409.0919>. arXiv: 1409.0919.
- C. Hegde, M. Wakin, and R. Baraniuk. Random Projections for Manifold Learning. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 641–648. Curran Associates, Inc., 2008. URL <http://papers.nips.cc/paper/3191-random-projections-for-manifold-learning.pdf>.
- G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew. Extreme learning machine: Theory and applications. 70(1):489–501. ISSN 09252312. doi: 10.1016/j.neucom.2005.12.126. URL <https://linkinghub.elsevier.com/retrieve/pii/S0925231206000385>.
- D. H. Hubel and T. N. Wiesel. Receptive fields of single neurones in the cat’s striate cortex. *The Journal of Physiology*, 148(3):574–591, Oct. 1959. ISSN 00223751. doi:

- 10.1113/jphysiol.1959.sp006308. URL <http://doi.wiley.com/10.1113/jphysiol.1959.sp006308>.
- S.-H. Ieng, C. Posch, and R. Benosman. Asynchronous Neuromorphic Event-Driven Image Filtering. *Proceedings of the IEEE*, 102(10):1485–1499, Oct. 2014. ISSN 0018-9219, 1558-2256. doi: 10.1109/JPROC.2014.2347355. URL <http://ieeexplore.ieee.org/document/6895246/>.
- S. Ioffe and C. Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *arXiv:1502.03167 [cs]*, Mar. 2015. URL <http://arxiv.org/abs/1502.03167>. arXiv: 1502.03167.
- K. Jameson and R. G. D’Andrade. It’s not really red, green, yellow, blue: an inquiry into perceptual color space. In C. L. Hardin and L. Maffi, editors, *Color Categories in Thought and Language*, pages 295–319. Cambridge University Press, 1 edition, Aug. 1997. ISBN 978-0-521-49693-3 978-0-521-49800-5 978-0-511-51981-9. doi: 10.1017/CBO9780511519819.014. URL https://www.cambridge.org/core/product/identifier/CBO9780511519819A024/type/book_part.
- L. Jiao, F. Zhang, F. Liu, S. Yang, L. Li, Z. Feng, and R. Qu. A Survey of Deep Learning-based Object Detection. *IEEE Access*, 7:128837–128868, 2019. ISSN 2169-3536. doi: 10.1109/ACCESS.2019.2939201. URL <http://arxiv.org/abs/1907.09408>. arXiv: 1907.09408.
- J. Kaiser, H. Mostafa, and E. Neftci. Synaptic Plasticity Dynamics for Deep Continuous Local Learning (DECOLLE). *Frontiers in Neuroscience*, 14:424, May 2020. ISSN 1662-453X. doi: 10.3389/fnins.2020.00424. URL <https://www.frontiersin.org/article/10.3389/fnins.2020.00424/full>.
- N. Kasabov, K. Dhoble, N. Nuntalid, and G. Indiveri. Dynamic evolving spiking neural networks for on-line spatio- and spectro-temporal pattern recognition. *Neural Networks*, 41:188–201, May 2013. ISSN 08936080. doi: 10.1016/j.neunet.2012.11.014. URL <https://linkinghub.elsevier.com/retrieve/pii/S0893608012003139>.
- H. Kim, A. Handa, R. Benosman, S.-H. Ieng, and A. Davison. Simultaneous mosaicing and tracking with an event camera. In *Proceedings of the British Machine Vision Conference*. BMVA Press, 2014a. doi: <http://dx.doi.org/10.5244/C.28.26>.
- H. Kim, A. Handa, R. Benosman, S.-H. Ieng, and A. Davison. Simultaneous Mosaicing and Tracking with an Event Camera. In *Proceedings of the British Machine Vision Conference 2014*, pages 26.1–26.12, Nottingham, 2014b. British Machine Vision Association. ISBN 978-1-901725-52-0. doi: 10.5244/C.28.26. URL <http://www.bmva.org/bmvc/2014/papers/paper066/index.html>.
- H. Kim, S. Leutenegger, and A. J. Davison. Real-Time 3D Reconstruction and 6-DoF Tracking with an Event Camera. In B. Leibe, J. Matas, N. Sebe, and M. Welling,

- editors, *Computer Vision – ECCV 2016*, volume 9910, pages 349–364. Springer International Publishing, Cham, 2016. ISBN 978-3-319-46465-7 978-3-319-46466-4. doi: 10.1007/978-3-319-46466-4_21. URL http://link.springer.com/10.1007/978-3-319-46466-4_21. Series Title: Lecture Notes in Computer Science.
- J. Kramer. An on/off transient imager with event-driven, asynchronous read-out. In *2002 IEEE International Symposium on Circuits and Systems. Proceedings (Cat. No.02CH37353)*, pages II–165–II–168, Phoenix-Scottsdale, AZ, USA, 2002. IEEE. ISBN 978-0-7803-7448-5. doi: 10.1109/ISCAS.2002.1010950. URL <http://ieeexplore.ieee.org/document/1010950/>.
- X. Lagorce, C. Meyer, S.-H. Ieng, D. Filliat, and R. Benosman. Asynchronous Event-Based Multikernel Algorithm for High-Speed Visual Features Tracking. *IEEE Transactions on Neural Networks and Learning Systems*, 26(8):1710–1720, Aug. 2015. ISSN 2162-237X, 2162-2388. doi: 10.1109/TNNLS.2014.2352401. URL <https://ieeexplore.ieee.org/document/6899691>.
- X. Lagorce, G. Orchard, F. Galluppi, B. E. Shi, and R. B. Benosman. HOTS: A Hierarchy of Event-Based Time-Surfaces for Pattern Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(7):1346–1359, July 2017. ISSN 0162-8828, 2160-9292. doi: 10.1109/TPAMI.2016.2574707. URL <http://ieeexplore.ieee.org/document/7508476/>.
- E. H. Land. Color vision and the natural image. part i. *Proceedings of the National Academy of Sciences of the United States of America*, 45(1):115, 1959.
- Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, Nov. 1998. ISSN 00189219. doi: 10.1109/5.726791. URL <http://ieeexplore.ieee.org/document/726791/>.
- B. B. Lee. The evolution of concepts of color vision. *Neurociencias*, 4(4):209, 2008.
- J. A. Lee and M. Verleysen. *Nonlinear Dimensionality Reduction*. Springer Publishing Company, Incorporated, 1st edition, 2007. ISBN 0387393501.
- J. H. Lee, T. Delbruck, M. Pfeiffer, P. K. J. 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, Dec. 2014. ISSN 2162-237X, 2162-2388. doi: 10.1109/TNNLS.2014.2308551. URL <http://ieeexplore.ieee.org/document/6774446/>.
- J. H. Lee, T. Delbruck, and M. Pfeiffer. Training Deep Spiking Neural Networks Using Backpropagation. *Frontiers in Neuroscience*, 10, Nov. 2016. ISSN 1662-453X. doi: 10.3389/fnins.2016.00508. URL <http://journal.frontiersin.org/article/10.3389/fnins.2016.00508/full>.

- J. A. Lenero-Bardallo, T. Serrano-Gotarredona, and B. Linares-Barranco. A mismatch calibrated bipolar spatial contrast AER retina with adjustable contrast threshold. In *2009 IEEE International Symposium on Circuits and Systems*, pages 1493–1496, Taipei, Taiwan, May 2009. IEEE. ISBN 978-1-4244-3827-3. doi: 10.1109/ISCAS.2009.5118050. URL <http://ieeexplore.ieee.org/document/5118050/>.
- J. A. Lenero-Bardallo, D. H. Bryn, and P. Hafliger. Flame monitoring with an AER color vision sensor. In *2013 IEEE International Symposium on Circuits and Systems (ISCAS2013)*, pages 2404–2407, Beijing, May 2013. IEEE. ISBN 978-1-4673-5762-3 978-1-4673-5760-9 978-1-4673-5761-6. doi: 10.1109/ISCAS.2013.6572363. URL <http://ieeexplore.ieee.org/document/6572363/>.
- J. A. Lenero-Bardallo, D. H. Bryn, and P. Hafliger. Bio-Inspired Asynchronous Pixel Event Tricolor Vision Sensor. *IEEE Transactions on Biomedical Circuits and Systems*, 8(3):345–357, June 2014. ISSN 1932-4545, 1940-9990. doi: 10.1109/TBCAS.2013.2271382. URL <http://ieeexplore.ieee.org/document/6575193/>.
- C. Li, C. Brandli, R. Berner, H. Liu, M. Yang, S.-C. Liu, and T. Delbruck. Design of an RGBW color VGA rolling and global shutter dynamic and active-pixel vision sensor. In *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 718–721, Lisbon, Portugal, May 2015. IEEE. ISBN 978-1-4799-8391-9. doi: 10.1109/ISCAS.2015.7168734. URL <http://ieeexplore.ieee.org/document/7168734/>.
- J. Li, F. Shi, W.-H. Liu, D. Zou, Q. Wang, P. K. Park, and H. Ryu. Adaptive temporal pooling for object detection using dynamic vision sensor. In *BMVC*, 2017.
- P. Lichtsteiner, C. Posch, and T. Delbruck. A 128×128 120 dB 15 μ s latency asynchronous temporal contrast vision sensor. 43(2):566–576. ISSN 0018-9200. doi: 10.1109/JSSC.2007.914337. URL <http://ieeexplore.ieee.org/document/4444573/>.
- P. Lichtsteiner, C. Posch, and T. Delbruck. A 128×128 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, San Francisco, CA, 2006. IEEE. ISBN 978-1-4244-0079-9. doi: 10.1109/ISSCC.2006.1696265. URL <http://ieeexplore.ieee.org/document/1696265/>.
- T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár. Microsoft COCO: Common Objects in Context. *arXiv:1405.0312 [cs]*, Feb. 2015. URL <http://arxiv.org/abs/1405.0312>. arXiv: 1405.0312.
- T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal Loss for Dense Object Detection. *arXiv:1708.02002 [cs]*, Feb. 2018. URL <http://arxiv.org/abs/1708.02002>. arXiv: 1708.02002.

- W. Lin, K. Hasenstab, G. M. Cunha, and A. Schwartzman. Comparison of handcrafted features and convolutional neural networks for liver mr image adequacy assessment. *Scientific Reports*, 10(1):1–11, 2020.
- A. Linares-Barranco, F. Gomez-Rodriguez, V. Villanueva, L. Longinotti, and T. Delbruck. A USB3.0 FPGA event-based filtering and tracking framework for dynamic vision sensors. In *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 2417–2420, Lisbon, Portugal, May 2015. IEEE. ISBN 978-1-4799-8391-9. doi: 10.1109/ISCAS.2015.7169172. URL <http://ieeexplore.ieee.org/document/7169172/>.
- M. Litzenberger, B. Kohn, A. 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, Toronto, ON, Canada, 2006a. IEEE. ISBN 978-1-4244-0093-5. doi: 10.1109/ITSC.2006.1706816. URL <http://ieeexplore.ieee.org/document/1706816/>.
- M. Litzenberger, C. Posch, D. Bauer, A. Belbachir, P. Schon, B. Kohn, and H. Garn. Embedded Vision System for Real-Time Object Tracking using an Asynchronous Transient Vision Sensor. In *2006 IEEE 12th Digital Signal Processing Workshop & 4th IEEE Signal Processing Education Workshop*, pages 173–178, Teton National Park, WY, USA, Sept. 2006b. IEEE. ISBN 978-1-4244-0535-0. doi: 10.1109/DSPWS.2006.265448. URL <http://ieeexplore.ieee.org/document/4041053/>.
- B. Liu, Y. Zhang, D. He, and Y. Li. Identification of Apple Leaf Diseases Based on Deep Convolutional Neural Networks. *Symmetry*, 10(1):11, Dec. 2017. ISSN 2073-8994. doi: 10.3390/sym10010011. URL <http://www.mdpi.com/2073-8994/10/1/11>.
- H. Liu and H. Motoda. *Feature Extraction, Construction and Selection: A Data Mining Perspective*. Kluwer Academic Publishers, USA, 1998. ISBN 0792381963.
- H. Liu, C. Brandli, C. Li, S.-C. Liu, and T. Delbruck. Design of a spatiotemporal correlation filter for event-based sensors. In *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 722–725, Lisbon, Portugal, May 2015. IEEE. ISBN 978-1-4799-8391-9. doi: 10.1109/ISCAS.2015.7168735. URL <http://ieeexplore.ieee.org/document/7168735/>.
- M. Liu and T. Delbruck. Adaptive Time-Slice Block-Matching Optical Flow Algorithm for Dynamic Vision Sensors. 2018. doi: 10.5167/UZH-168589. URL <https://www.zora.uzh.ch/id/eprint/168589>. Publisher: Proceedings of BMVC 2018.
- W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. SSD: Single Shot MultiBox Detector. *arXiv:1512.02325 [cs]*, 9905:21–37, 2016. doi: 10.1007/978-3-319-46448-0_2. URL <http://arxiv.org/abs/1512.02325>. arXiv: 1512.02325.

- Y. Lu, S. Yi, N. Zeng, Y. Liu, and Y. Zhang. Identification of rice diseases using deep convolutional neural networks. *Neurocomputing*, 267:378–384, Dec. 2017. ISSN 09252312. doi: 10.1016/j.neucom.2017.06.023. URL <https://linkinghub.elsevier.com/retrieve/pii/S0925231217311384>.
- L. Luo, Y. Tang, X. Zou, C. Wang, P. Zhang, and W. Feng. Robust Grape Cluster Detection in a Vineyard by Combining the AdaBoost Framework and Multiple Color Components. *Sensors*, 16(12):2098, Dec. 2016. ISSN 1424-8220. doi: 10.3390/s16122098. URL <http://www.mdpi.com/1424-8220/16/12/2098>.
- J. Ma, K. Du, F. Zheng, L. Zhang, Z. Gong, and Z. Sun. A recognition method for cucumber diseases using leaf symptom images based on deep convolutional neural network. *Computers and Electronics in Agriculture*, 154:18–24, Nov. 2018. ISSN 01681699. doi: 10.1016/j.compag.2018.08.048. URL <https://linkinghub.elsevier.com/retrieve/pii/S0168169918309360>.
- S. Ma, X. Zhang, C. Jia, Z. Zhao, S. Wang, and S. Wang. Image and Video Compression with Neural Networks: A Review. *IEEE Transactions on Circuits and Systems for Video Technology*, 30(6):1683–1698, June 2020. ISSN 1051-8215, 1558-2205. doi: 10.1109/TCSVT.2019.2910119. URL <http://arxiv.org/abs/1904.03567>. arXiv: 1904.03567.
- M. A. Mahowald. *VLSI Analogs of Neuronal Visual Processing: A Synthesis of Form and Function*. PhD thesis, USA, 1992.
- U. Mallik, M. Clapp, Edward Choi, G. Cauwenberghs, and R. Etienne-Cummings. Temporal change threshold detection imager. In *ISSCC. 2005 IEEE International Digest of Technical Papers. Solid-State Circuits Conference, 2005.*, pages 362–364, San Francisco, CA, USA, 2005. IEEE. ISBN 978-0-7803-8904-5. doi: 10.1109/ISSCC.2005.1494019. URL <http://ieeexplore.ieee.org/document/1494019/>.
- A. I. Maqueda, A. Loquercio, G. Gallego, N. Garcia, and D. Scaramuzza. Event-Based Vision Meets Deep Learning on Steering Prediction for Self-Driving Cars. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5419–5427, Salt Lake City, UT, June 2018. IEEE. ISBN 978-1-5386-6420-9. doi: 10.1109/CVPR.2018.00568. URL <https://ieeexplore.ieee.org/document/8578666/>.
- A. Marcireau, S.-H. Ieng, and R. Benosman. Sepia, tarsier, and chameleon: A modular c++ framework for event-based computer vision. 13:1338. ISSN 1662-453X. doi: 10.3389/fnins.2019.01338. URL <https://www.frontiersin.org/article/10.3389/fnins.2019.01338/full>.
- A. Marcireau, S.-H. Ieng, C. Simon-Chane, and R. B. Benosman. Event-Based Color Segmentation With a High Dynamic Range Sensor. *Frontiers in Neuroscience*, 12: 135, Apr. 2018. ISSN 1662-453X. doi: 10.3389/fnins.2018.00135. URL <http://journal.frontiersin.org/article/10.3389/fnins.2018.00135/full>.

- N. Matsuda, O. Cossairt, and M. Gupta. MC3D: Motion Contrast 3D Scanning. In *2015 IEEE International Conference on Computational Photography (ICCP)*, pages 1–10, Houston, TX, USA, Apr. 2015. IEEE. ISBN 978-1-4799-8667-5. doi: 10.1109/ICCPHOT.2015.7168370. URL <http://ieeexplore.ieee.org/document/7168370/>.
- M. Mehdipour Ghazi, B. Yanikoglu, and E. Aptoula. Plant identification using deep neural networks via optimization of transfer learning parameters. *Neurocomputing*, 235:228–235, Apr. 2017. ISSN 09252312. doi: 10.1016/j.neucom.2017.01.018. URL <https://linkinghub.elsevier.com/retrieve/pii/S0925231217300498>.
- D. P. Moeys, C. Li, J. N. Martel, S. Bamford, L. Longinotti, V. Motsnyi, D. San Segundo Bello, and T. Delbruck. Color temporal contrast sensitivity in dynamic vision sensors. In *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–4, Baltimore, MD, May 2017. IEEE. ISBN 978-1-4673-6853-7. doi: 10.1109/ISCAS.2017.8050412. URL <https://ieeexplore.ieee.org/document/8050412/>.
- D. P. Moeys, F. Corradi, C. Li, S. A. Bamford, L. Longinotti, F. F. Voigt, S. Berry, G. Taverni, F. Helmchen, and T. Delbruck. A Sensitive Dynamic and Active Pixel Vision Sensor for Color or Neural Imaging Applications. *IEEE Transactions on Biomedical Circuits and Systems*, 12(1):123–136, Feb. 2018. ISSN 1932-4545, 1940-9990. doi: 10.1109/TBCAS.2017.2759783. URL <https://ieeexplore.ieee.org/document/8094907/>.
- H. Mostafa, V. Ramesh, and G. Cauwenberghs. Deep Supervised Learning Using Local Errors. *Frontiers in Neuroscience*, 12:608, Aug. 2018. ISSN 1662-453X. doi: 10.3389/fnins.2018.00608. URL <https://www.frontiersin.org/article/10.3389/fnins.2018.00608/full>.
- L. Nanni, S. Ghidoni, and S. Brahnam. Handcrafted vs. non-handcrafted features for computer vision classification. *Pattern Recognition*, 71:158–172, Nov. 2017. ISSN 00313203. doi: 10.1016/j.patcog.2017.05.025. URL <https://linkinghub.elsevier.com/retrieve/pii/S0031320317302224>.
- P. M. Narendra and K. Fukunaga. A branch and bound algorithm for feature subset selection. *IEEE Trans. Comput.*, 26(9):917–922, Sept. 1977. ISSN 0018-9340. doi: 10.1109/TC.1977.1674939. URL <https://doi.org/10.1109/TC.1977.1674939>.
- E. O. Neftci, H. Mostafa, and F. Zenke. Surrogate Gradient Learning in Spiking Neural Networks: Bringing the Power of Gradient-Based Optimization to Spiking Neural Networks. *IEEE Signal Processing Magazine*, 36(6):51–63, Nov. 2019. ISSN 1053-5888, 1558-0792. doi: 10.1109/MSP.2019.2931595. URL <https://ieeexplore.ieee.org/document/8891809/>.
- Z. Ni, S.-H. Ieng, C. Posch, S. Régner, and R. Benosman. Visual Tracking Using Neuromorphic Asynchronous Event-Based Cameras. *Neural Computation*, 27(4):

- 925–953, Apr. 2015. ISSN 0899-7667, 1530-888X. doi: 10.1162/NECO_a_00720. URL <https://direct.mit.edu/neco/article/27/4/925-953/8061>.
- P. O’Connor, D. Neil, S.-C. Liu, T. Delbruck, and M. Pfeiffer. Real-time classification and sensor fusion with a spiking deep belief network. *Frontiers in Neuroscience*, 7, 2013. ISSN 1662-453X. doi: 10.3389/fnins.2013.00178. URL <http://journal.frontiersin.org/article/10.3389/fnins.2013.00178/abstract>.
- J. Olsson and P. Hafliger. Two color asynchronous event photo pixel. In *2008 IEEE International Symposium on Circuits and Systems*, pages 2146–2149, Seattle, WA, USA, May 2008. IEEE. ISBN 978-1-4244-1683-7. doi: 10.1109/ISCAS.2008.4541875. URL <http://ieeexplore.ieee.org/document/4541875/>.
- G. Orchard, A. Jayawant, G. K. Cohen, and N. Thakor. Converting Static Image Datasets to Spiking Neuromorphic Datasets Using Saccades. *Front. Neurosci.*, 9, Nov. 2015. ISSN 1662-453X. doi: 10.3389/fnins.2015.00437. URL <http://journal.frontiersin.org/Article/10.3389/fnins.2015.00437/abstract>.
- V. Padala, A. Basu, and G. Orchard. A Noise Filtering Algorithm for Event-Based Asynchronous Change Detection Image Sensors on TrueNorth and Its Implementation on TrueNorth. *Frontiers in Neuroscience*, 12:118, Mar. 2018. ISSN 1662-453X. doi: 10.3389/fnins.2018.00118. URL <http://journal.frontiersin.org/article/10.3389/fnins.2018.00118/full>.
- 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. *arXiv:1811.10180 [cs]*, Nov. 2018. URL <http://arxiv.org/abs/1811.10180>. arXiv: 1811.10180.
- G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter. Continual Lifelong Learning with Neural Networks: A Review. *Neural Networks*, 113:54–71, May 2019. ISSN 08936080. doi: 10.1016/j.neunet.2019.01.012. URL <http://arxiv.org/abs/1802.07569>. arXiv: 1802.07569.
- G. Pass, R. Zabih, and J. Miller. Comparing images using color coherence vectors. In *Proceedings of the fourth ACM international conference on Multimedia - MULTIMEDIA ’96*, pages 65–73, Boston, Massachusetts, United States, 1996. ACM Press. ISBN 978-0-89791-871-8. doi: 10.1145/244130.244148. URL <http://portal.acm.org/citation.cfm?doid=244130.244148>.
- K. Pearson. LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, Nov. 1901. ISSN 1941-5982, 1941-5990. doi: 10.1080/14786440109462720. URL <https://www.tandfonline.com/doi/full/10.1080/14786440109462720>.
- C. Posch. Bioinspired Vision Sensing. In G. Cristóbal, L. Perrinet, and M. S. Keil, editors, *Biologically Inspired Computer Vision*, pages 11–28. Wiley-VCH Verlag GmbH

- & Co. KGaA, Weinheim, Germany, Aug. 2015. ISBN 978-3-527-68086-3 978-3-527-41264-8. doi: 10.1002/9783527680863.ch2. URL <http://doi.wiley.com/10.1002/9783527680863.ch2>.
- 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, Jan. 2011. ISSN 0018-9200, 1558-173X. doi: 10.1109/JSSC.2010.2085952. URL <http://ieeexplore.ieee.org/document/5648367/>.
- M. Rahnemoonfar and C. Sheppard. Deep Count: Fruit Counting Based on Deep Simulated Learning. *Sensors*, 17(4):905, Apr. 2017. ISSN 1424-8220. doi: 10.3390/s17040905. URL <https://www.mdpi.com/1424-8220/17/4/905>.
- A. Rakotomamonjy. Variable selection using svm based criteria. *J. Mach. Learn. Res.*, 3(null):1357–1370, Mar. 2003. ISSN 1532-4435.
- H. Rebecq, T. Horstschaefer, 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, Apr. 2017. ISSN 2377-3766, 2377-3774. doi: 10.1109/LRA.2016.2645143. URL <http://ieeexplore.ieee.org/document/7797445/>.
- 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, Dec. 2018. ISSN 0920-5691, 1573-1405. doi: 10.1007/s11263-017-1050-6. URL <http://link.springer.com/10.1007/s11263-017-1050-6>.
- H. Rebecq, R. Ranftl, V. Koltun, and D. Scaramuzza. High Speed and High Dynamic Range Video with an Event Camera. *arXiv:1906.07165 [cs]*, June 2019. URL <http://arxiv.org/abs/1906.07165>. arXiv: 1906.07165.
- J. Redmon and A. Farhadi. YOLO9000: Better, Faster, Stronger. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6517–6525, Honolulu, HI, July 2017. IEEE. ISBN 978-1-5386-0457-1. doi: 10.1109/CVPR.2017.690. URL <http://ieeexplore.ieee.org/document/8100173/>.
- J. Redmon and A. Farhadi. YOLOv3: An Incremental Improvement. *arXiv:1804.02767 [cs]*, Apr. 2018. URL <http://arxiv.org/abs/1804.02767>. arXiv: 1804.02767.
- J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You Only Look Once: Unified, Real-Time Object Detection. *arXiv:1506.02640 [cs]*, May 2016. URL <http://arxiv.org/abs/1506.02640>. arXiv: 1506.02640.
- S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *arXiv:1506.01497 [cs]*, Jan. 2016. URL <http://arxiv.org/abs/1506.01497>. arXiv: 1506.01497.

- P. Rogister, R. Benosman, Sio-Hoi Ieng, P. Lichtsteiner, and T. Delbruck. Asynchronous Event-Based Binocular Stereo Matching. *IEEE Transactions on Neural Networks and Learning Systems*, 23(2):347–353, Feb. 2012a. ISSN 2162-237X, 2162-2388. doi: 10.1109/TNNLS.2011.2180025. URL <http://ieeexplore.ieee.org/document/6112233/>.
- P. Rogister, R. Benosman, Sio-Hoi Ieng, P. Lichtsteiner, and T. Delbruck. Asynchronous Event-Based Binocular Stereo Matching. *IEEE Transactions on Neural Networks and Learning Systems*, 23(2):347–353, Feb. 2012b. ISSN 2162-237X, 2162-2388. doi: 10.1109/TNNLS.2011.2180025. URL <http://ieeexplore.ieee.org/document/6112233/>.
- O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. *arXiv:1505.04597 [cs]*, May 2015. URL <http://arxiv.org/abs/1505.04597>. arXiv: 1505.04597.
- P. Roy and V. Isler. Vision-Based Apple Counting and Yield Estimation. In D. Kulić, Y. Nakamura, O. Khatib, and G. Venture, editors, *2016 International Symposium on Experimental Robotics*, volume 1, pages 478–487. Springer International Publishing, Cham, 2017. ISBN 978-3-319-50114-7 978-3-319-50115-4. doi: 10.1007/978-3-319-50115-4_42. URL http://link.springer.com/10.1007/978-3-319-50115-4_42. Series Title: Springer Proceedings in Advanced Robotics.
- O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *arXiv:1409.0575 [cs]*, Jan. 2015. URL <http://arxiv.org/abs/1409.0575>. arXiv: 1409.0575.
- I. Sa, Z. Ge, F. Dayoub, B. Upcroft, T. Perez, and C. McCool. DeepFruits: A Fruit Detection System Using Deep Neural Networks. *Sensors*, 16(8):1222, Aug. 2016. ISSN 1424-8220. doi: 10.3390/s16081222. URL <http://www.mdpi.com/1424-8220/16/8/1222>.
- C. Scheerlinck, N. Barnes, and R. Mahony. Continuous-time Intensity Estimation Using Event Cameras. *arXiv:1811.00386 [cs]*, Nov. 2018. URL <http://arxiv.org/abs/1811.00386>. arXiv: 1811.00386.
- C. Scheerlinck, H. Rebecq, T. Stoffregen, N. Barnes, R. Mahony, and D. Scaramuzza. CED: Color Event Camera Dataset. *arXiv:1904.10772 [cs]*, Apr. 2019. URL <http://arxiv.org/abs/1904.10772>. arXiv: 1904.10772.
- D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, and M. Young. Machine learning: The high interest credit card of technical debt. In *SE4ML: Software Engineering for Machine Learning (NIPS 2014 Workshop)*, 2014.
- T. J. Sejnowski. The unreasonable effectiveness of deep learning in artificial intelligence. *Proceedings of the National Academy of Sciences*, 117(48):30033–30038, Dec. 2020.

- ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.1907373117. URL <http://www.pnas.org/lookup/doi/10.1073/pnas.1907373117>.
- Y. Sekikawa, K. Hara, and H. Saito. EventNet: Asynchronous Recursive Event Processing. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3882–3891, Long Beach, CA, USA, June 2019. IEEE. ISBN 978-1-72813-293-8. doi: 10.1109/CVPR.2019.00401. URL <https://ieeexplore.ieee.org/document/8954313/>.
- K. J. Seymour, M. A. Williams, and A. N. Rich. The Representation of Color across the Human Visual Cortex: Distinguishing Chromatic Signals Contributing to Object Form Versus Surface Color. *Cerebral Cortex*, 26(5):1997–2005, May 2016. ISSN 1047-3211, 1460-2199. doi: 10.1093/cercor/bhv021. URL <https://academic.oup.com/cercor/article-lookup/doi/10.1093/cercor/bhv021>.
- K. Shimonomura. Color Silicon Retina System with Color Constancy. *The Journal of The Institute of Image Information and Television Engineers*, 65(3):361–363, 2011. ISSN 1881-6908, 1342-6907. doi: 10.3169/itej.65.361. URL <http://joi.jlc.jst.go.jp/JST.JSTAGE/itej/65.361?from=CrossRef>.
- A. Sironi, M. Brambilla, N. Bourdis, X. Lagorce, and R. Benosman. HATS: Histograms of Averaged Time Surfaces for Robust Event-based Object Classification. *arXiv:1803.07913 [cs]*, Mar. 2018. URL <http://arxiv.org/abs/1803.07913>. arXiv: 1803.07913.
- Y. Song, C. Glasbey, G. Horgan, G. Polder, J. Dieleman, and G. van der Heijden. Automatic fruit recognition and counting from multiple images. *Biosystems Engineering*, 118:203–215, Feb. 2014. ISSN 15375110. doi: 10.1016/j.biosystemseng.2013.12.008. URL <https://linkinghub.elsevier.com/retrieve/pii/S1537511013002109>.
- N. Sánchez-Marroño, A. Alonso-Betanzos, and M. Tombilla-Sanromán. Filter Methods for Feature Selection – A Comparative Study. In H. Yin, P. Tino, E. Corchado, W. Byrne, and X. Yao, editors, *Intelligent Data Engineering and Automated Learning - IDEAL 2007*, volume 4881, pages 178–187. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007. ISBN 978-3-540-77225-5. doi: 10.1007/978-3-540-77226-2_19. URL http://link.springer.com/10.1007/978-3-540-77226-2_19. Series Title: Lecture Notes in Computer Science.
- G. Taverni, D. Paul Moeys, C. Li, C. Cavaco, V. Motsnyi, D. San Segundo Bello, and T. Delbruck. Front and Back Illuminated Dynamic and Active Pixel Vision Sensors Comparison. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 65(5): 677–681, May 2018. ISSN 1549-7747, 1558-3791. doi: 10.1109/TCSII.2018.2824899. URL <https://ieeexplore.ieee.org/document/8334288/>.
- A. Trémeau, S. Tominaga, and K. N. Plataniotis. Color in Image and Video Processing: Most Recent Trends and Future Research Directions. *EURASIP Journal on Image*

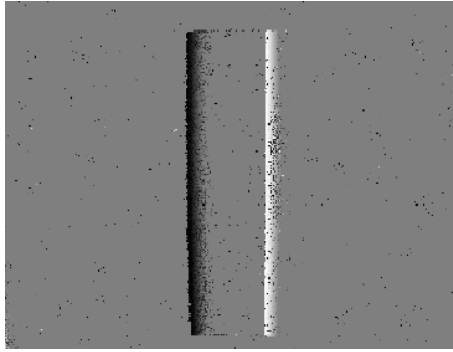
- and Video Processing*, 2008:1–26, 2008. ISSN 1687-5176, 1687-5281. doi: 10.1155/2008/581371. URL <http://jivp.eurasipjournals.com/content/2008/1/581371>.
- L. van der Maaten, E. O. Postma, and J. van den Herik. Dimensionality reduction: A comparative review. 2009.
- A. van Schaik and J. Tapson. Online and adaptive pseudoinverse solutions for ELM weights. *Neurocomputing*, 149:233–238, Feb. 2015. ISSN 09252312. doi: 10.1016/j.neucom.2014.01.071. URL <https://linkinghub.elsevier.com/retrieve/pii/S0925231214011485>.
- 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, Apr. 2018. ISSN 2377-3766, 2377-3774. doi: 10.1109/LRA.2018.2793357. URL <http://arxiv.org/abs/1709.06310>. arXiv: 1709.06310.
- P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–511–I–518, Kauai, HI, USA, 2001. IEEE Comput. Soc. ISBN 978-0-7695-1272-3. doi: 10.1109/CVPR.2001.990517. URL <http://ieeexplore.ieee.org/document/990517/>.
- P. Viola and M. J. Jones. Robust Real-Time Face Detection. *International Journal of Computer Vision*, 57(2):137–154, May 2004. ISSN 0920-5691. doi: 10.1023/B:VISI.0000013087.49260.fb. URL <http://link.springer.com/10.1023/B:VISI.0000013087.49260.fb>.
- V. Viqueira Pérez, D. De Fez Saiz, and F. Martinez Verdú. Colour vision: theories and principles. In *Colour Measurement*, pages 3–e2. Elsevier, 2010. ISBN 978-1-84569-559-0. doi: 10.1533/9780857090195.1.3. URL <https://linkinghub.elsevier.com/retrieve/pii/B9781845695590500005>.
- C. Wang, W. S. Lee, X. Zou, D. Choi, H. Gan, and J. Diamond. Correction to: Detection and counting of immature green citrus fruit based on the Local Binary Patterns (LBP) feature using illumination-normalized images. *Precision Agriculture*, 19(6): 1084–1084, Dec. 2018. ISSN 1385-2256, 1573-1618. doi: 10.1007/s11119-018-9580-7. URL <http://link.springer.com/10.1007/s11119-018-9580-7>.
- L. Wang, I. M. Mostafavi, Y.-S. Ho, and K.-J. Yoon. Event-Based High Dynamic Range Image and Very High Frame Rate Video Generation Using Conditional Generative Adversarial Networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10073–10082, Long Beach, CA, USA, June 2019. IEEE. ISBN 978-1-72813-293-8. doi: 10.1109/CVPR.2019.01032. URL <https://ieeexplore.ieee.org/document/8954323/>.

- K. Q. Weinberger and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. *J. Mach. Learn. Res.*, 10:207–244, June 2009. ISSN 1532-4435.
- D. Wettschereck, D. W. Aha, and T. Mohri. [No title found]. *Artificial Intelligence Review*, 11(1/5):273–314, 1997. ISSN 02692821. doi: 10.1023/A:1006593614256. URL <http://link.springer.com/10.1023/A:1006593614256>.
- L. Yang, R. Jin, and R. Sukthankar. Bayesian Active Distance Metric Learning. *arXiv:1206.5283 [cs, stat]*, June 2012. URL <http://arxiv.org/abs/1206.5283>. arXiv: 1206.5283.
- Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning, ICML '97*, page 412–420, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc. ISBN 1558604863.
- Y. Yu, K. Zhang, L. Yang, and D. Zhang. Fruit detection for strawberry harvesting robot in non-structural environment based on Mask-RCNN. *Computers and Electronics in Agriculture*, 163:104846, Aug. 2019. ISSN 01681699. doi: 10.1016/j.compag.2019.06.001. URL <https://linkinghub.elsevier.com/retrieve/pii/S0168169919301103>.
- K. Zaghloul and K. Boahen. Optic Nerve Signals in a Neuromorphic Chip I: Outer and Inner Retina Models. *IEEE Transactions on Biomedical Engineering*, 51(4):657–666, Apr. 2004a. ISSN 0018-9294. doi: 10.1109/TBME.2003.821039. URL <http://ieeexplore.ieee.org/document/1275581/>.
- K. Zaghloul and K. Boahen. Optic Nerve Signals in a Neuromorphic Chip II: Testing and Results. *IEEE Transactions on Biomedical Engineering*, 51(4):667–675, Apr. 2004b. ISSN 0018-9294. doi: 10.1109/TBME.2003.821040. URL <http://ieeexplore.ieee.org/document/1275582/>.
- F. H. Zaidi, J. T. Hull, S. Peirson, K. Wulff, D. Aeschbach, J. J. Gooley, G. Brainard, K. Gregory-Evans, J. Rizzo, C. A. Czeisler, R. Foster, M. J. Moseley, and S. W. Lockley. Short-Wavelength Light Sensitivity of Circadian, Pupillary, and Visual Awareness in Humans Lacking an Outer Retina. *Current Biology*, 17(24):2122–2128, Dec. 2007. ISSN 09609822. doi: 10.1016/j.cub.2007.11.034. URL <https://linkinghub.elsevier.com/retrieve/pii/S0960982207022737>.
- Zhenjiang Ni, A. Bolopion, J. Agnus, R. Benosman, and S. Regnier. Asynchronous Event-Based Visual Shape Tracking for Stable Haptic Feedback in Microrobotics. *IEEE Transactions on Robotics*, 28(5):1081–1089, Oct. 2012. ISSN 1552-3098, 1941-0468. doi: 10.1109/TRO.2012.2198930. URL <http://ieeexplore.ieee.org/document/6204348/>.

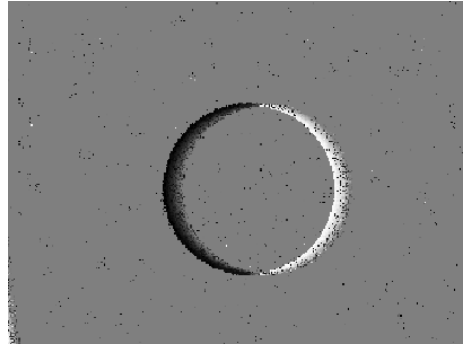
- L. Zhong, L. Hu, and H. Zhou. Deep learning based multi-temporal crop classification. *Remote Sensing of Environment*, 221:430–443, Feb. 2019. ISSN 00344257. doi: 10.1016/j.rse.2018.11.032. URL <https://linkinghub.elsevier.com/retrieve/pii/S0034425718305418>.
- R. Zhou, L. Damerow, Y. Sun, and M. M. Blanke. Using colour features of cv. ‘Gala’ apple fruits in an orchard in image processing to predict yield. *Precision Agriculture*, 13(5):568–580, Oct. 2012. ISSN 1385-2256, 1573-1618. doi: 10.1007/s11119-012-9269-2. URL <http://link.springer.com/10.1007/s11119-012-9269-2>.
- A. Z. Zhu, L. Yuan, K. Chaney, and K. Daniilidis. EV-FlowNet: Self-Supervised Optical Flow Estimation for Event-based Cameras. *Robotics: Science and Systems XIV*, June 2018a. doi: 10.15607/RSS.2018.XIV.062. URL <http://arxiv.org/abs/1802.06898>. arXiv: 1802.06898.
- A. Z. Zhu, L. Yuan, K. Chaney, and K. Daniilidis. Unsupervised Event-based Learning of Optical Flow, Depth, and Egomotion. *arXiv:1812.08156 [cs]*, Dec. 2018b. URL <http://arxiv.org/abs/1812.08156>. arXiv: 1812.08156.
- Z. Zou, Z. Shi, Y. Guo, and J. Ye. Object Detection in 20 Years: A Survey. *arXiv:1905.05055 [cs]*, May 2019. URL <http://arxiv.org/abs/1905.05055>. arXiv: 1905.05055.

Appendix A

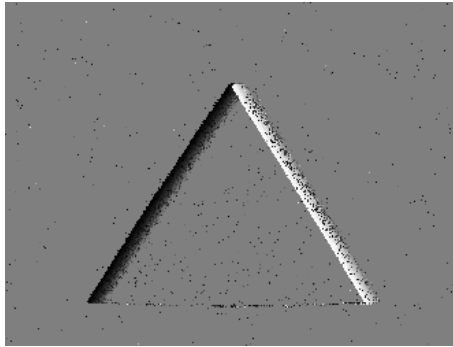
Object Classification using Statistical Properties with K-Nearest Neighbors



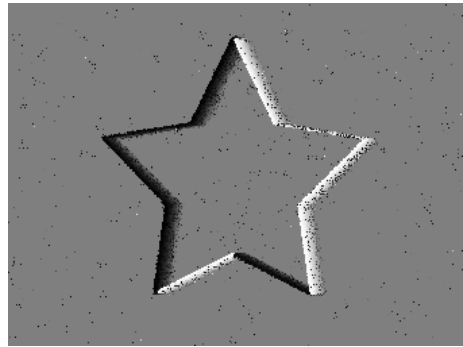
(a)



(b)



(c)



(d)

Figure A.1: The classes selected for classification.

As described in Section 3.5.6, we concluded that colour events are best to be used in cases where shapes and appearances are not a relevant source of information. It was shown that it is possible to classify circles with the same shape and appearance purely based on colour events internal properties. In this section, the aim was to show that classifying objects based on their geometrical appearance led to a similar outcome using the standard DAVIS sensor. In this experiment, different objects with different

colour are used as an input stimuli (See Figure A.1) to two types of network: (1) KNN classifier and (2) FEAST with an ELM.

Table A.1: Statistical summary of the 4 classes dataset. Results are shown separately for the training set of 50 recordings per class, and the testing set comprising of another 50 recordings

Statistics	Training Set		Test Set	
	Mean	Std	Mean	Std
Duration of Recordings (s)	2.04	1.36	2.04	1.36
Number of Events (*1e5)	3.60	0.31	3.60	0.35
Number of ON Events (*1e3)	2.34	0.52	2.32	0.53
Number of OFF Events (*1e5)	3.36	0.27	3.36	0.27
X Address (px)	186.74	2.94	186.74	2.84
Y Address (px)	133.87	0.97	133.87	0.97

As the nature of each training and testing item is no longer a static image, but rather an event sequence from a physical device, the specific characteristics relating to events of each sequence varies slightly, which was similar to what was observed in Section 3.5.6 due to the nature of the patterns and the event-based nature of the DAVIS device. Table A.1 shows the characteristics of the sequences in the training and testing dataset, respectively. It can be seen that the structure of both the testing and training set are similar. That is because the pattern lengths are consistent between the testing and training sets, and based on this data, a fixed pattern length of 2 seconds was chosen to contain all the patterns sequences fully.

The first classifier was based on KNN algorithm. K was set to be \sqrt{x} where x is the total number of the training set. Cross-validation of 50/50 between the training and testing set was applied to the data. Figure A.2 shows the results of the KNN classifier when applied to all classes. The classifiers based on a total number of events and OFF events yielded statistically insignificant classification results. As the original objects are positioned at the centre of the camera field of view and using the same motion movement with the same recording duration, it holds that the datasets exhibit this same property. It is to be expected that the number of events and OFF Events should hold no statistical values. Other statistical results demonstrated that the object x and y addresses contain significant classification power, which was expected due to the differences in the original shape.

The second classifier was based on FEAST algorithm, which was followed by a backend classifier. Two methods were implemented to extract robust and discriminative features from the original datasets: (1) extract features individually from each class as shown in Figure A.3(a) (2) extract unsupervised features from all classes from the training set as shown in Figure A.3(b). The latter extracts unsupervised features from

each object regardless of their class, taking advantage of the adaptive threshold and the weight updated rules which exist in FEAST—however, the former use supervisory signals during feature extraction. A set of feature neurons were assigned and dedicated for each class, forcing FEAST only to learn relevant features from each object. For the first method, 20 neurons were selected for each class, making the total number of neurons 80 for all the classes combined, and 80 neurons were assigned for the second method to make a valid performance comparison between both implementations. The same linear and ELM classifier with the same number of hidden neurons were used in both methods. Table A.2 and Table A.3 presents the results on both methods using the linear classifier. It was evident that having dedicated features for each class led to better classification results. For instance, the network achieved 0.99 informedness on the circle while the second approach achieved 0.55, which is a significant drop. The model accuracy was 98.99% for network pipeline one and 71.33% for pipeline 2. These results indicated the significant improvement in performance by using the supervisory labels as input to FEAST as opposed to learning all the features from all the classes combined. Figure A.4 shows the classification results using ELM with different numbers of neurons in the hidden layer by performing a sweep from 1000 neurons to 8000 neurons. As expected, the classifier based on the first network architecture (i.e. using dedicated features per class) achieved superior performance, evident in overall evaluation metrics. The average accuracy across all hidden neurons was between 60% and 80% for network architecture one and between 40% and 60% for the network architecture 2. Increasing the number of neurons in the hidden layers produce no significant improvement in the classification performance.

Based on these observations and the results in Section 3.5.6, we concluded the following: (1) the classification task is one of the least challenging classification tasks for the DAVIS and CDAVIS where there is a single object per recording and each object has a different shape, (2) having dedicated features for each class will always guarantee superior results and (3) colour events are suitable to be used where colour features are more visible and relevant than the shape of the object.

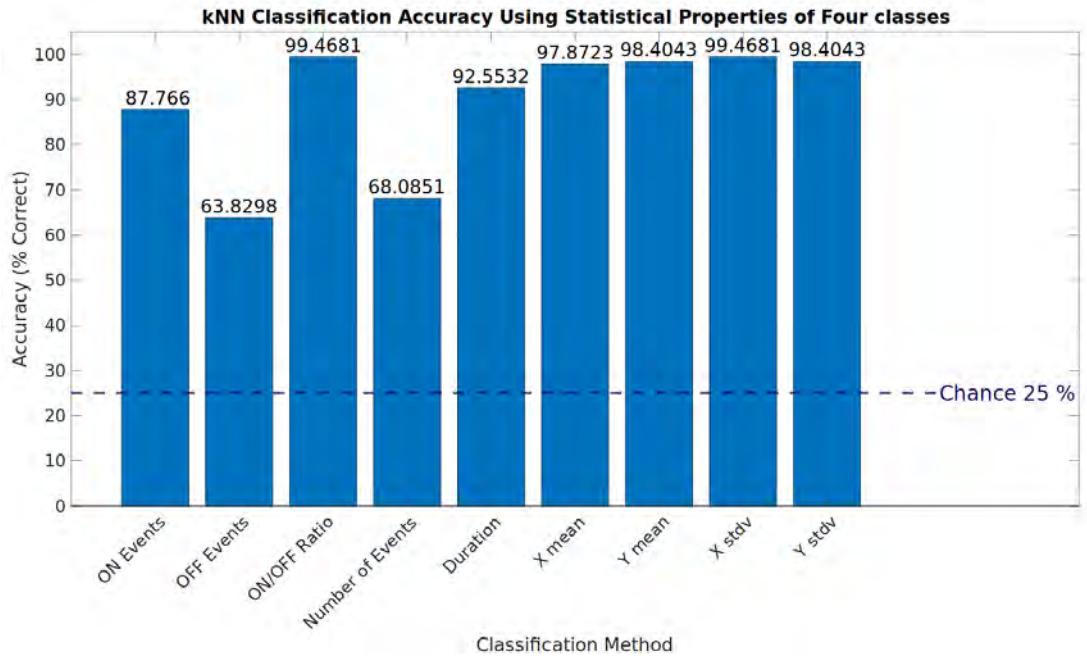


Figure A.2: Classification results for the KNN classifier based on the statistical properties of the events.

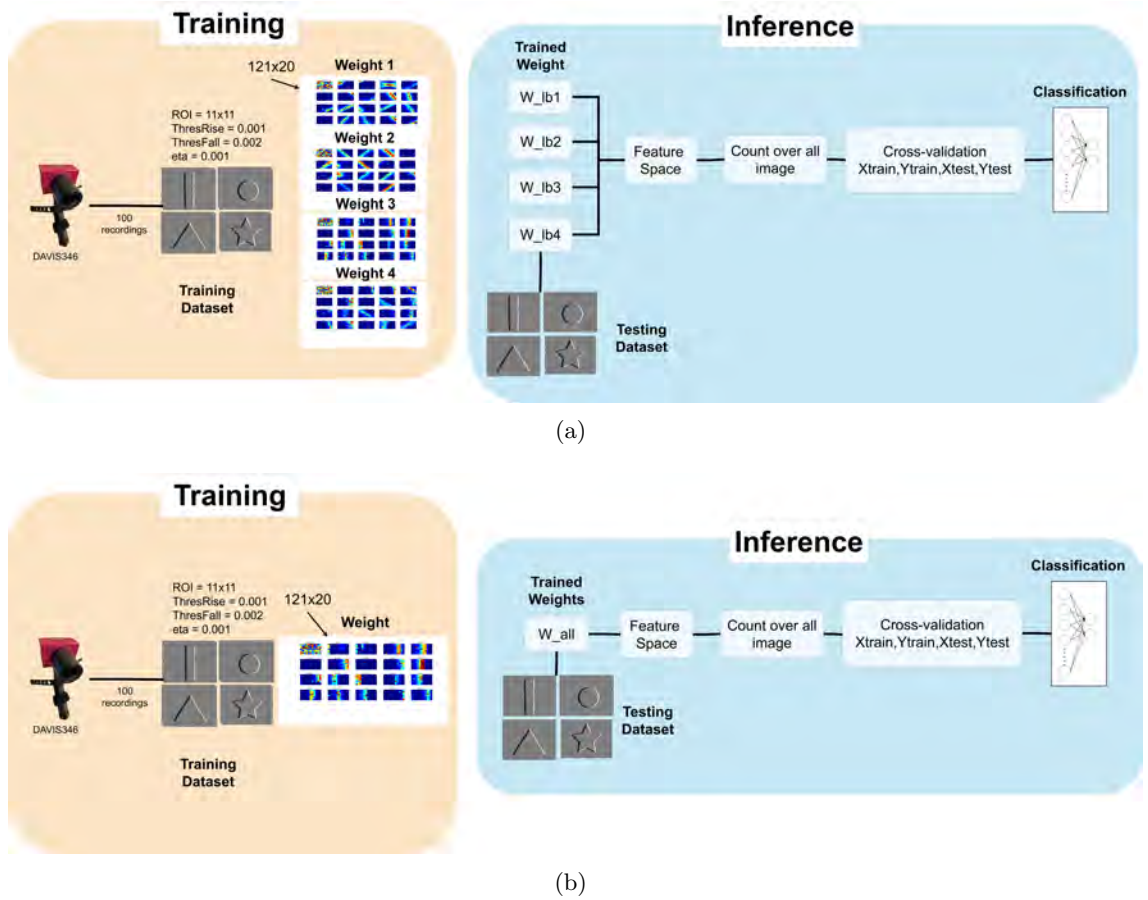


Figure A.3: Two types of classification architectures using FEAST algorithm. (a) A feature detector for each class followed by a classifier network. This is called "Architecture 1". (b) A feature detector for all the classes followed by a classifier network. This is called "Architecture 2".

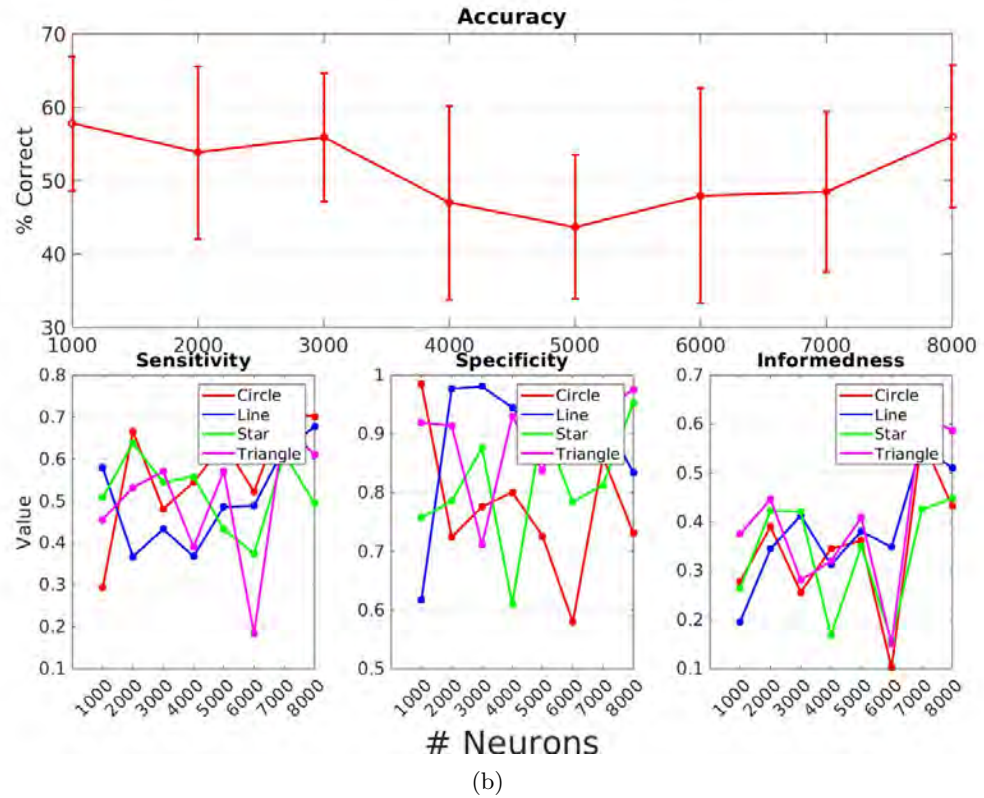
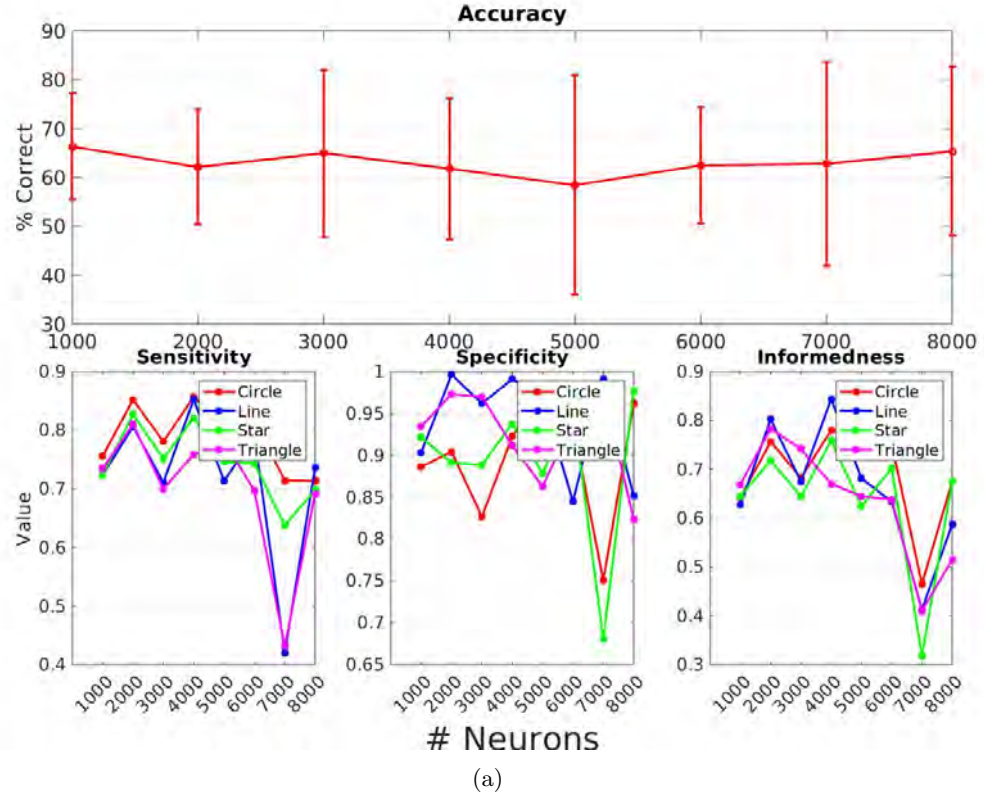


Figure A.4: A summary of the performance of the classification architecture using ELM classifier. (a) shows results on network Architecture 1 using dedicated features for each class. (b) shows results on network Architecture 2 using features from all the classes.

Table A.2: Linear classifier performance for network architecture 1

	Circle	Line	Star	Triangle
Accuracy	98.99			
Sensitivity	0.99	0.97	0.98	0.99
Specificity	0.99	0.99	0.99	0.99
Informedness	0.99	0.97	0.97	0.99

Table A.3: Linear classifier performance for network architecture 2

	Circle	Line	Star	Triangle
Accuracy	71.33			
Sensitivity	0.73	0.83	0.47	0.80
Specificity	0.82	0.91	0.91	0.96
Informedness	0.55	0.75	0.39	0.77

Appendix B

Dichromatic Sensor Assembly: Camera Synchronisation

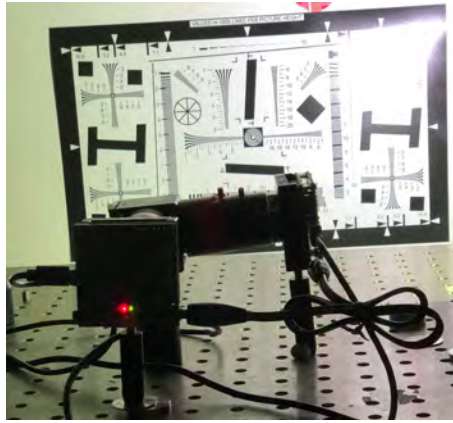
We built a dichromatic event-based colour sensor as an association of two DAVIS cameras acquiring red and green light exposures. The sensor captures light through a hot mirror reflecting infra-red light. A dichroic beam splitter directs photons with wavelengths larger than 610 nm towards the red sensor. The other photons are reflected towards the green sensor, allowing photons with a wavelength between 335 and 610 nm. Before hitting the red and green sensors, photons cross-band filters mimic the filtering functions of the conventional colour filter array or Bayer matrices pixels. Each sensor uses a C-mount objective. Consequently, each sensor is calibrated individually. Figure B.1(a), Figure B.1(b) and Figure B.1(c) illustrates the camera assembly with all the optical components. To account for the mechanical imperfections of the prototype, a spatial calibration step is required to ensure the colour sensor camera shares the same field of view. A calibration board was used with the sensor before each recording. The spatial calibration is valid only when the object within the camera field of view share the same size. The camera associated with each colour component generates an independent stream of events. To ensure both of the streams are triggered simultaneously, we have synchronised both of the camera timestamps. To achieve this, we utilised the synchronisation protocol associated with the camera. In this protocol, a device must be told whether it is a master, which produces synchronisation pulses, or a slave, which receives synchronisation pulses. A 10 kHz clock is used to advance the timestamps. In slaves, timestamps are allowed to advance on the falling edge of the clock. If the falling edge is delayed by a short period, then any events in that waiting period continue to take the same timestamp until the falling edge is detected. After a short period, a falling pulse is assumed to represent a reset pulse.

The synchronisation of multiple DAVIS cameras was performed by using 3.5 mm audio jacks connected at both sides of the camera labelled "IN" and "OUT". To minimise the synchronisation delay, we used the event-based software processor jAER¹. We modified the process of writing events to the .aodat file format by removing the unrec-

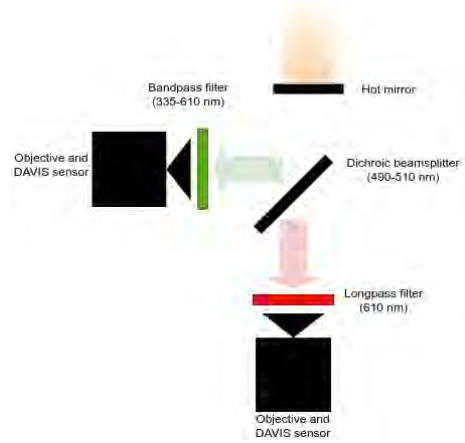
¹<https://github.com/SensorsINI/jaer>

essary headers and information in the binary file, including the camera specifications. Therefore, it takes less time to synchronise without writing this extra information. The synchronisation delay was reduced from between 200-300ms to as low as 0.5ms.

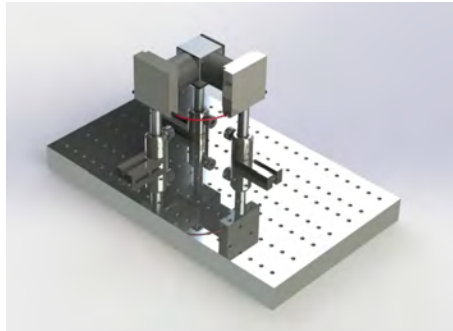
This was considered as an initial dichromatic colour EBC prototype. However, the cost of assembling two cameras and a beam splitter is much higher than colour filter arrays, and the error rate is much higher due to the physical position of the colour filter, beam splitter and the lens-camera flange distance. Therefore, the attention was shifted toward characterising the CDAVIS as described in Chapter 3.



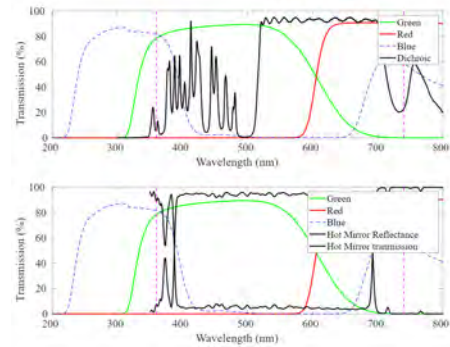
(a)



(b)



(c)



(d)

Figure B.1: Dichromatic sensor assembly. (a) The two chips EBC is an assembly of two monochromatic DAVIS cameras. (b) The two chips EBC with the beamsplitter to split the incoming light. (c) A 3D render of the entire setup in full assembly on the optic table. (d) Colour filters and hot mirror wavelength range and transmission percentage.

Appendix C

Additional Tables and Figures for the Event-based Classifier

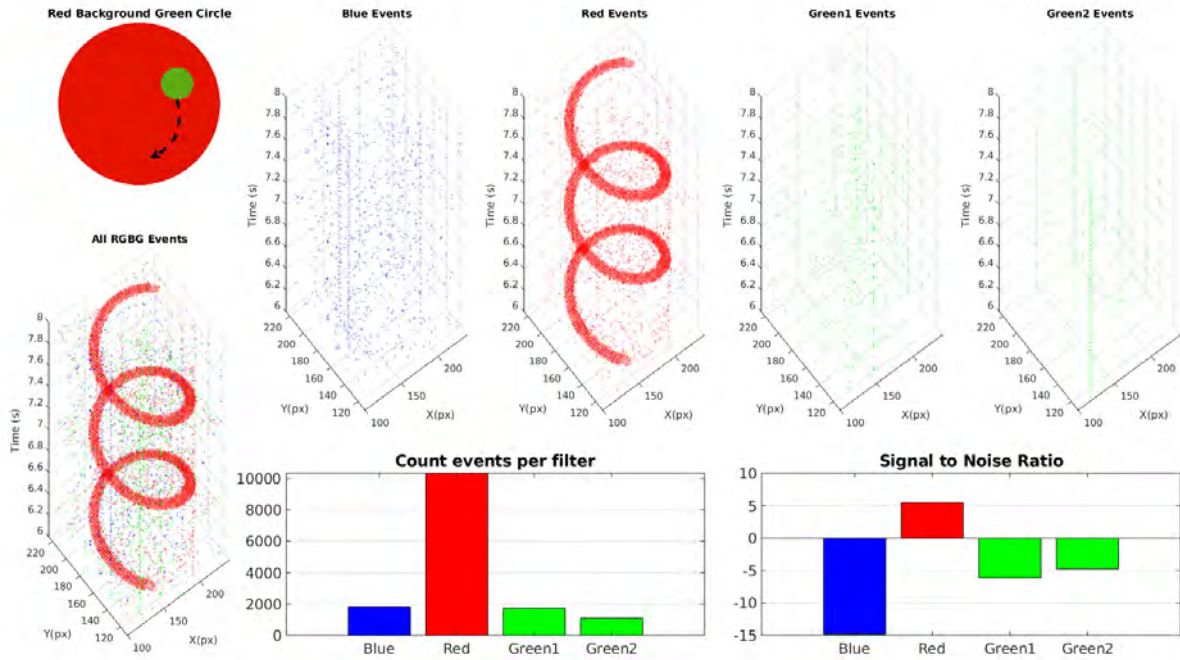


Figure C.1: A green circle rotated over a red background. In this case only the rotating circles can be seen through the red filter.

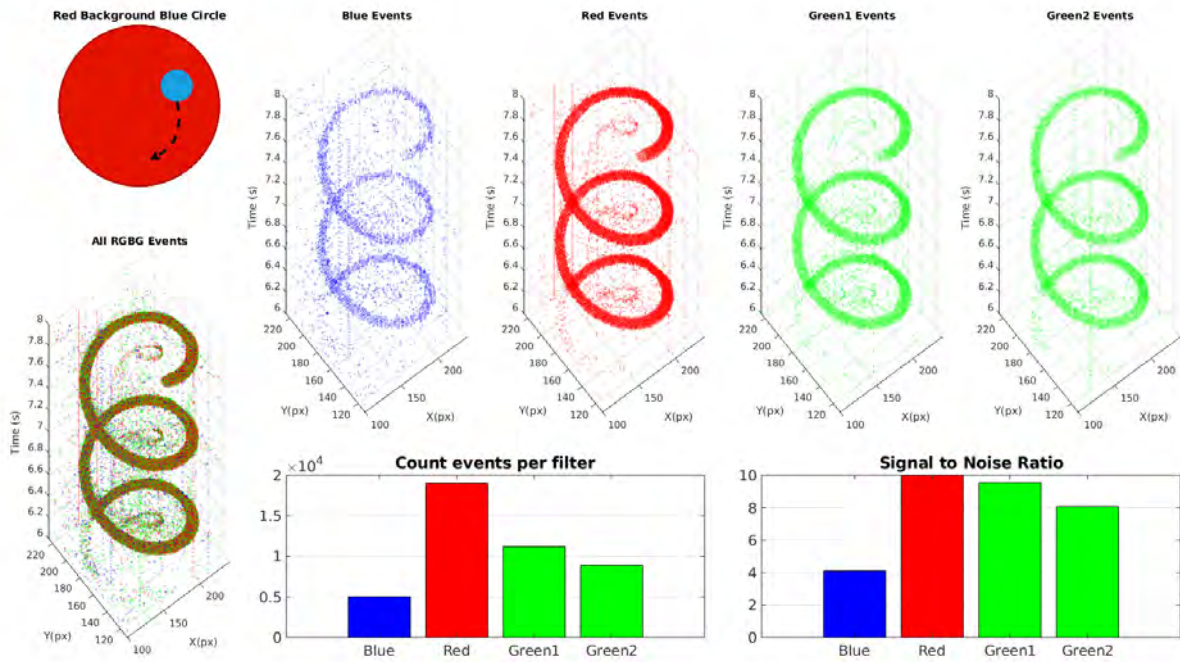


Figure C.2: A blue circle rotated over a red background. Due to the contrast between the red and the blue the circle can be seen through all colour filters.

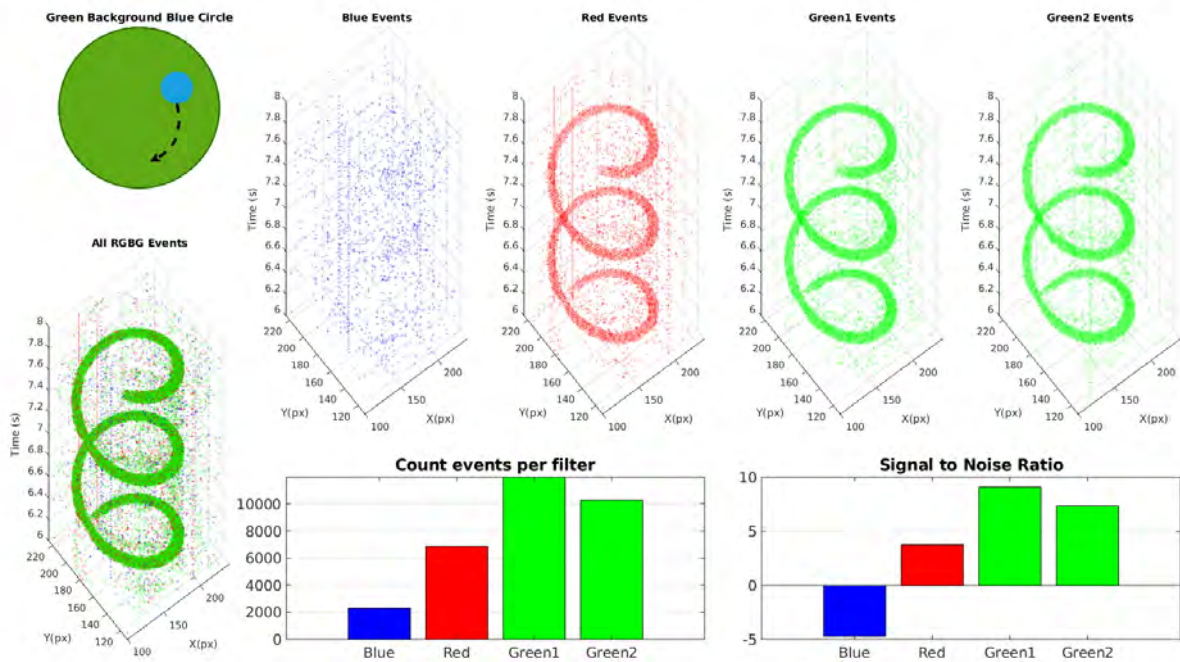


Figure C.3: A blue circle rotated over a green background. In this case the circle is visible through the red and green filter but not the blue due to the contrast difference between the blue and the green.

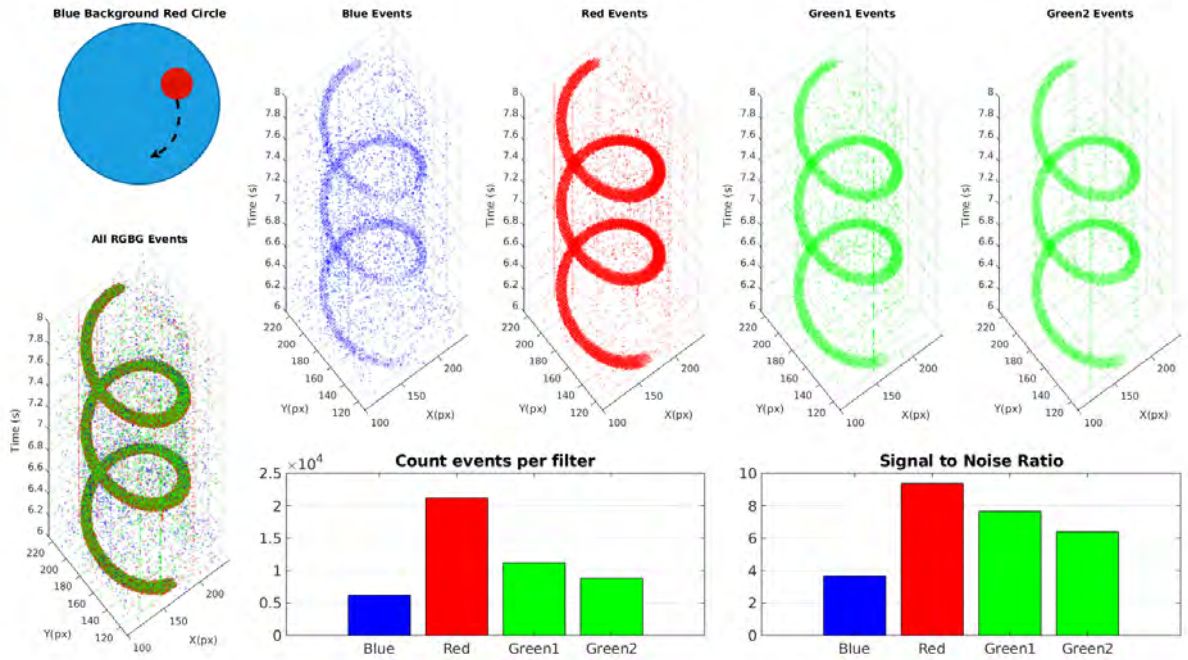


Figure C.4: A red circle rotated over a blue background.

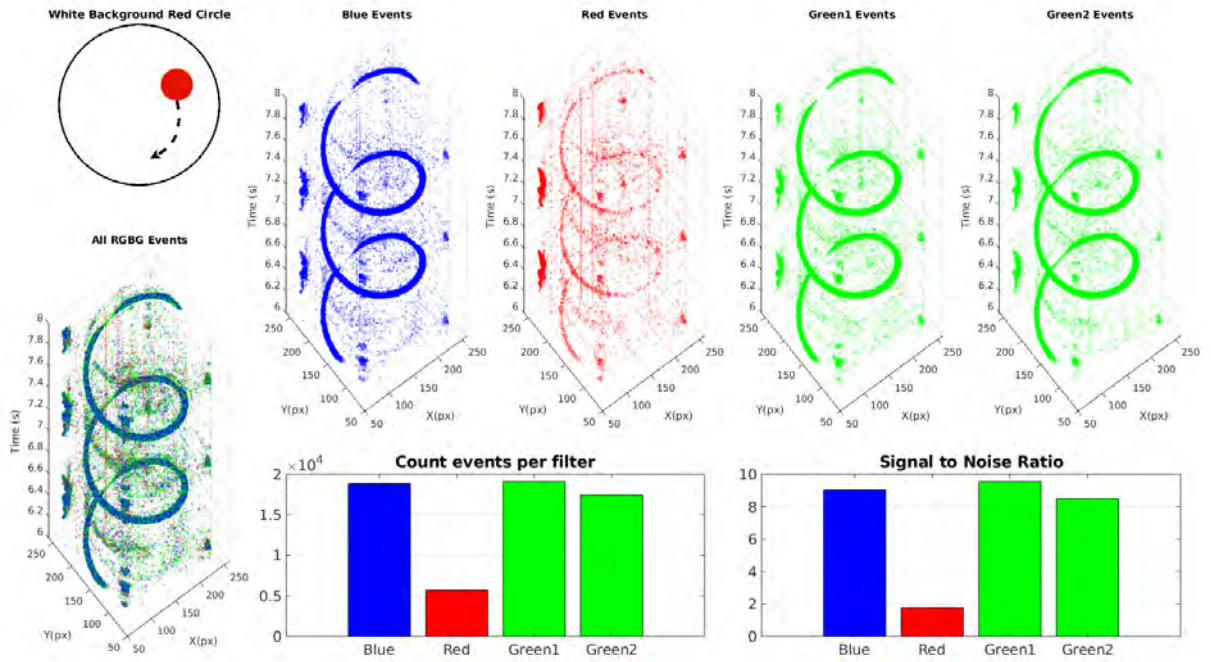


Figure C.5: A red circle rotated over a white background.

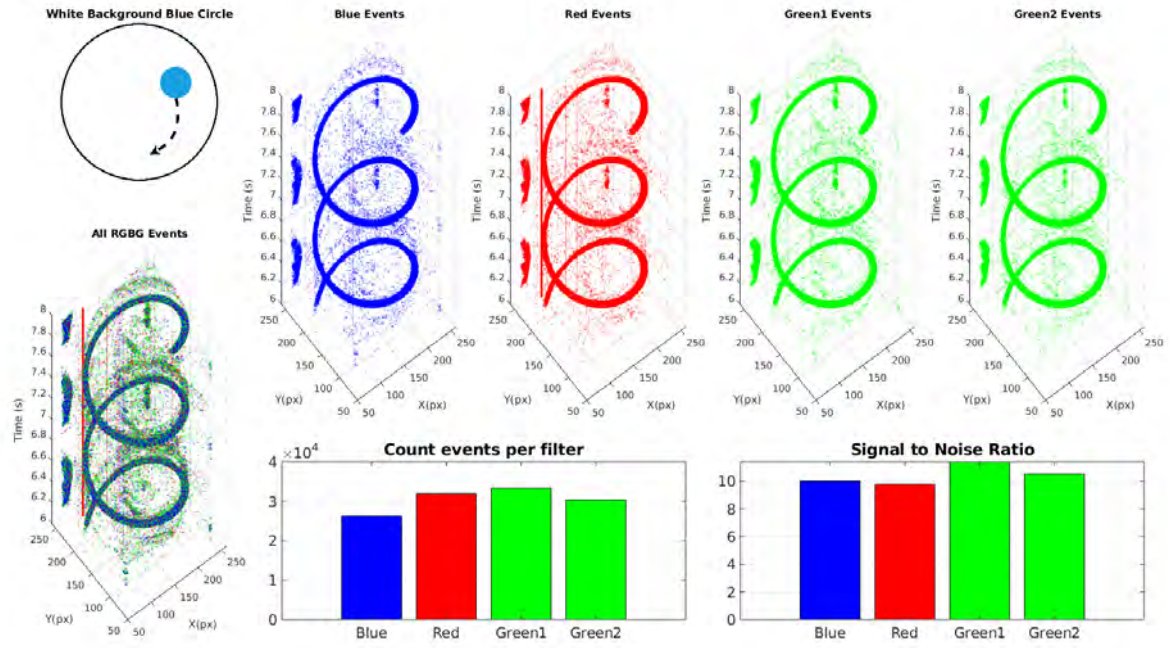


Figure C.6: A blue circle rotated over a white background.

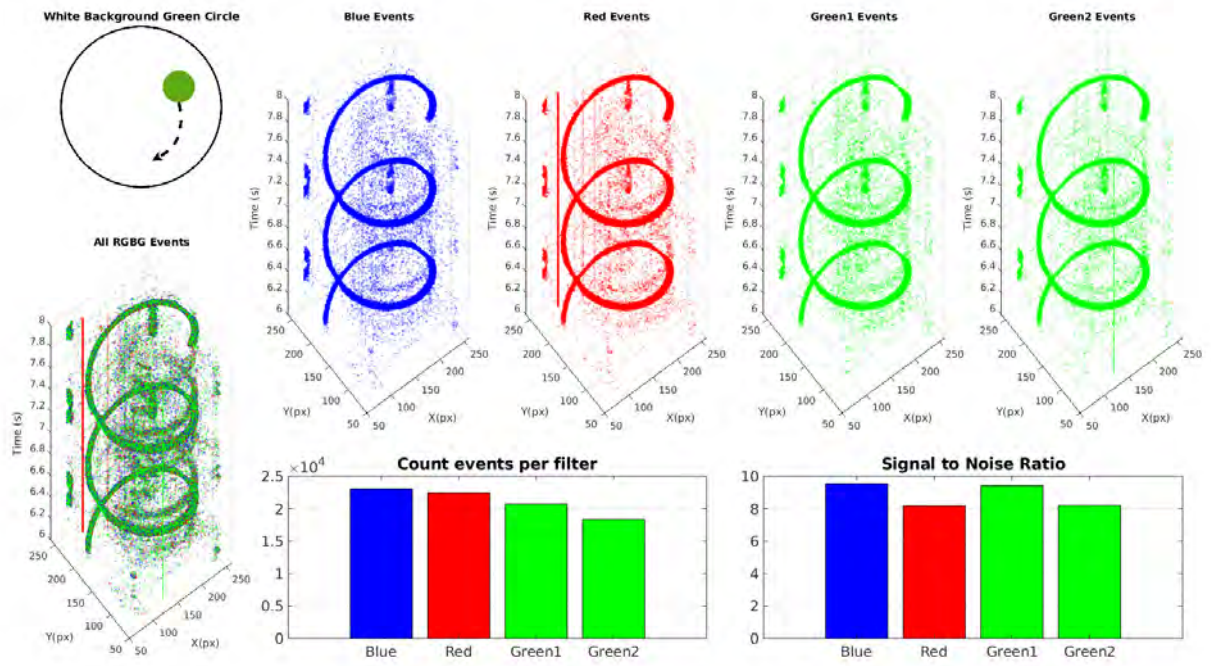


Figure C.7: A green circle rotated over a white background.

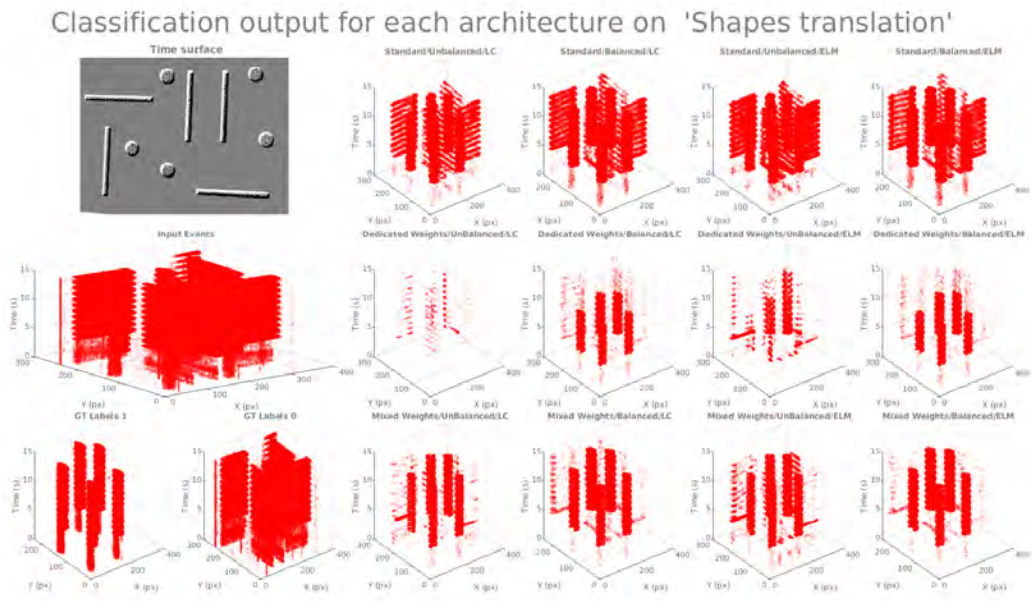


Figure C.8: Lab recorded data "Shapes translation". Each plot shows the classification output for each architecture using both classification methods on the balanced and imbalanced dataset.

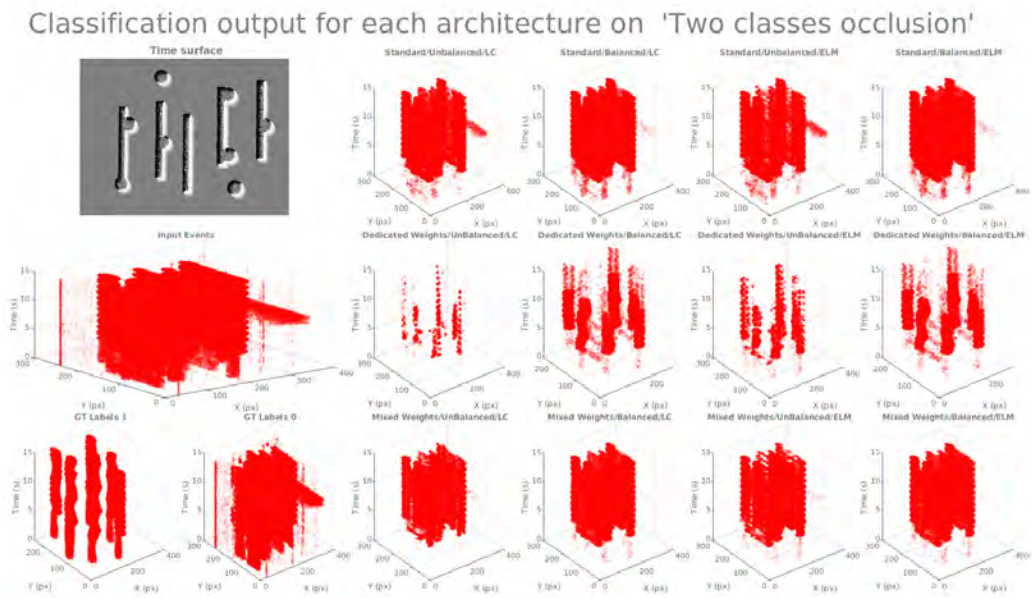


Figure C.9: Lab recorded "Two classes occlusion". Each plot shows the classification output for each architecture using both classification methods on the balanced and imbalanced dataset.

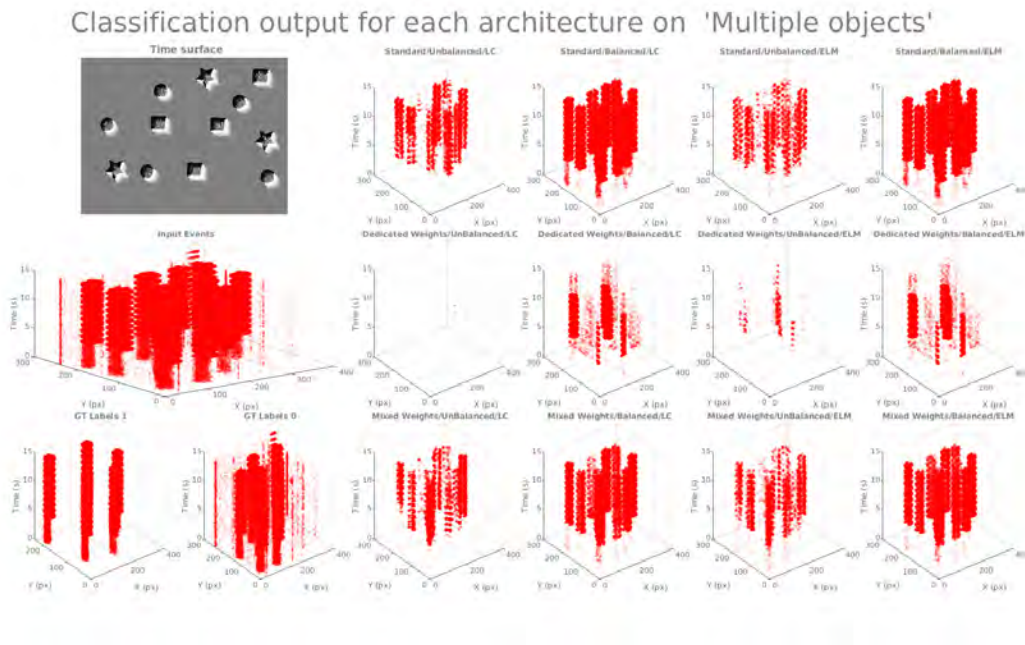


Figure C.10: Lab recorded data "Multiple objects". Each plot shows the classification output for each architecture using both classification methods on the balanced and imbalanced dataset.

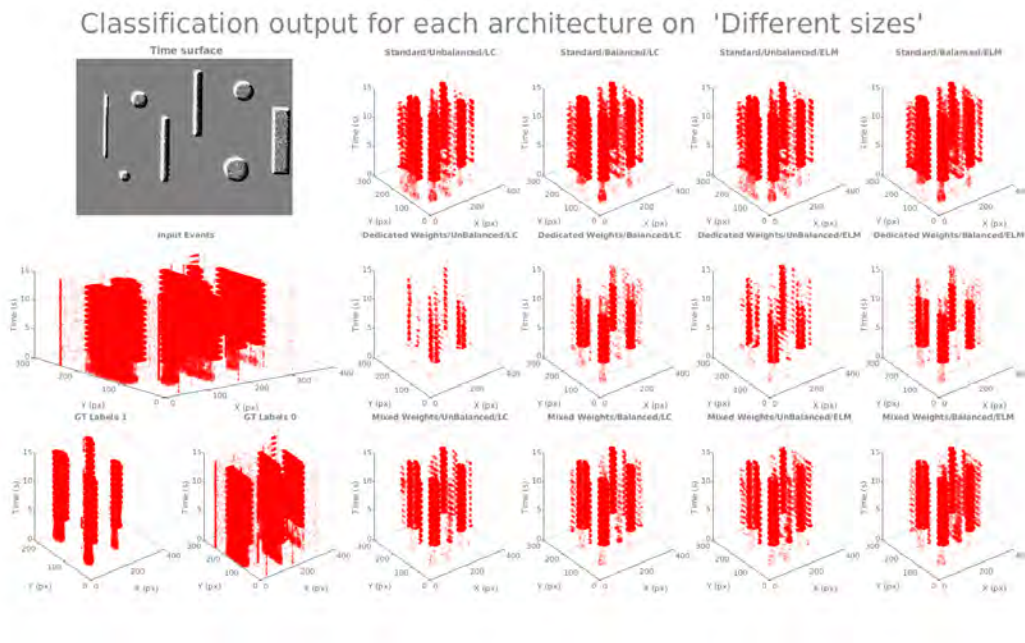


Figure C.11: Lab recorded data "Different sizes". Each plot shows the classification output for each architecture using both classification methods on the balanced and imbalanced dataset.

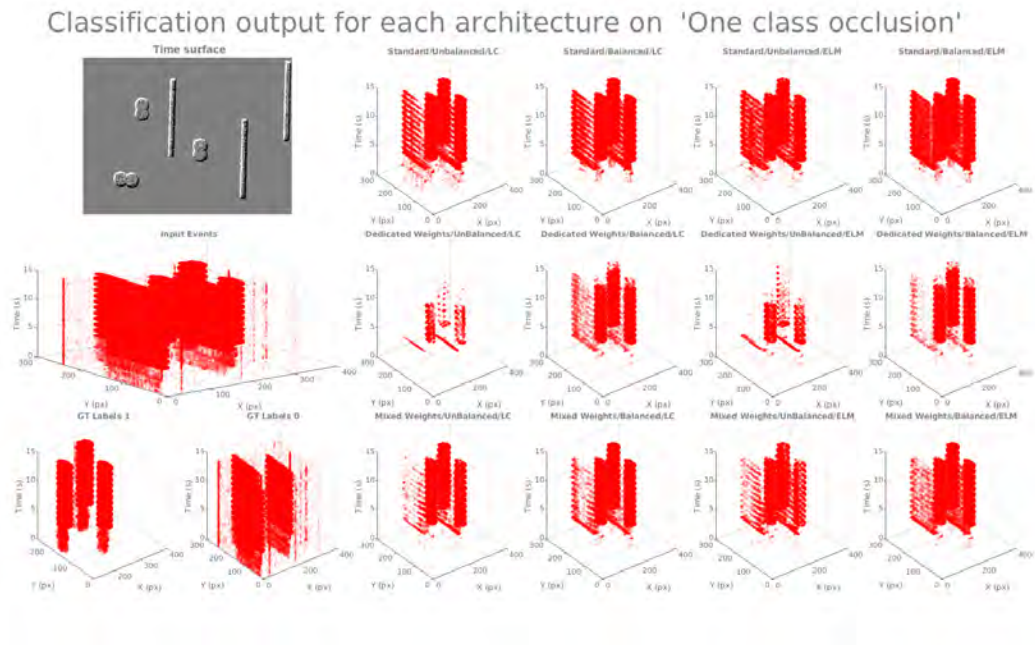


Figure C.12: Lab recorded data "One class occlusion". Each plot shows the classification output for each architecture using both classification methods on the balanced and imbalanced dataset.

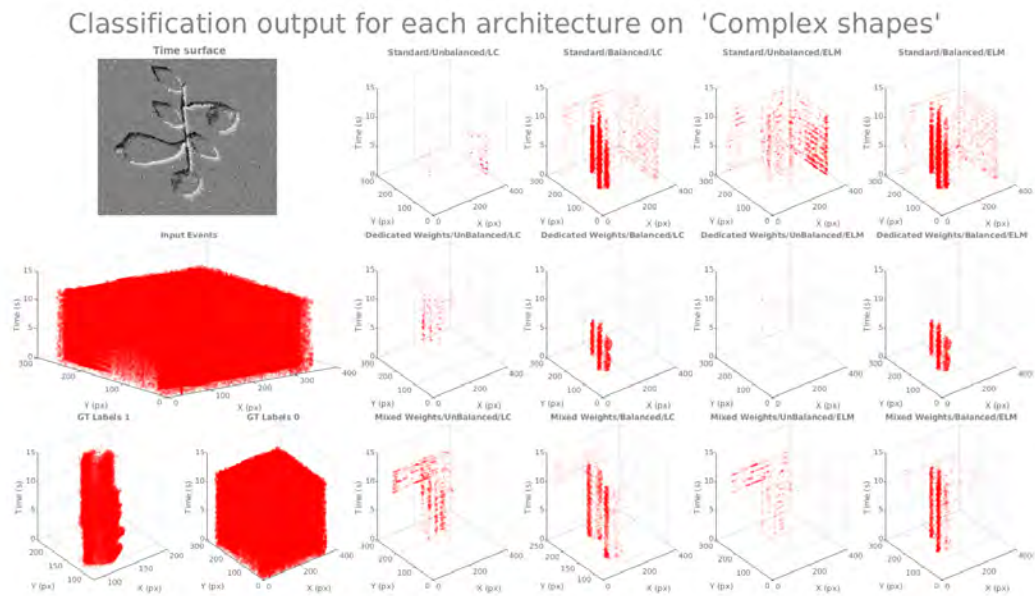


Figure C.13: Lab recorded data "Complex shapes". Each plot shows the classification output for each architecture using both classification methods on the balanced and imbalanced dataset.

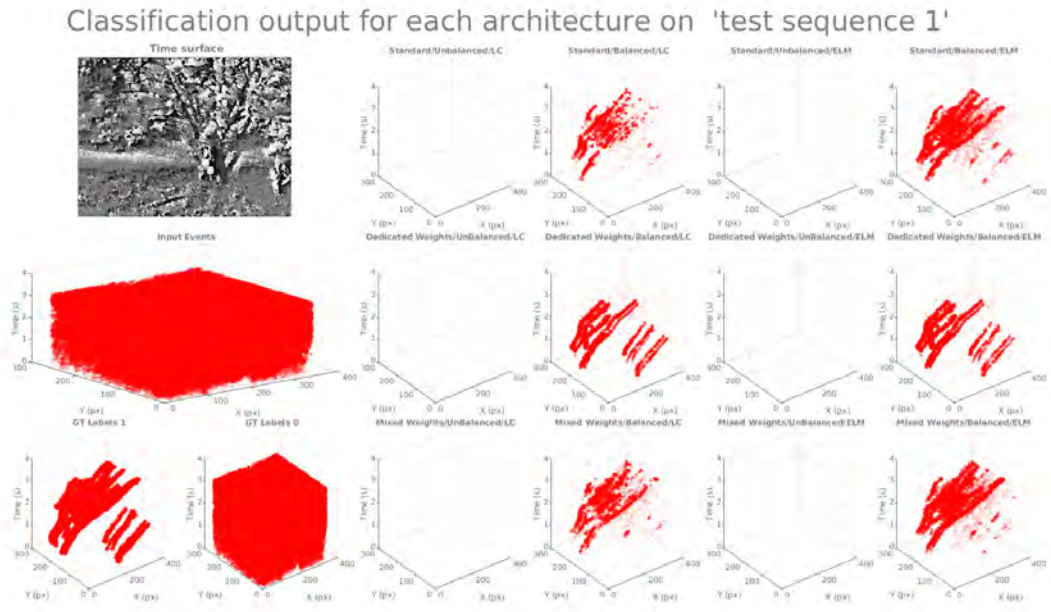


Figure C.14: Real world data "test sequence 1". Each plot shows the classification output for each architecture using both classification methods on the balanced and imbalanced dataset.

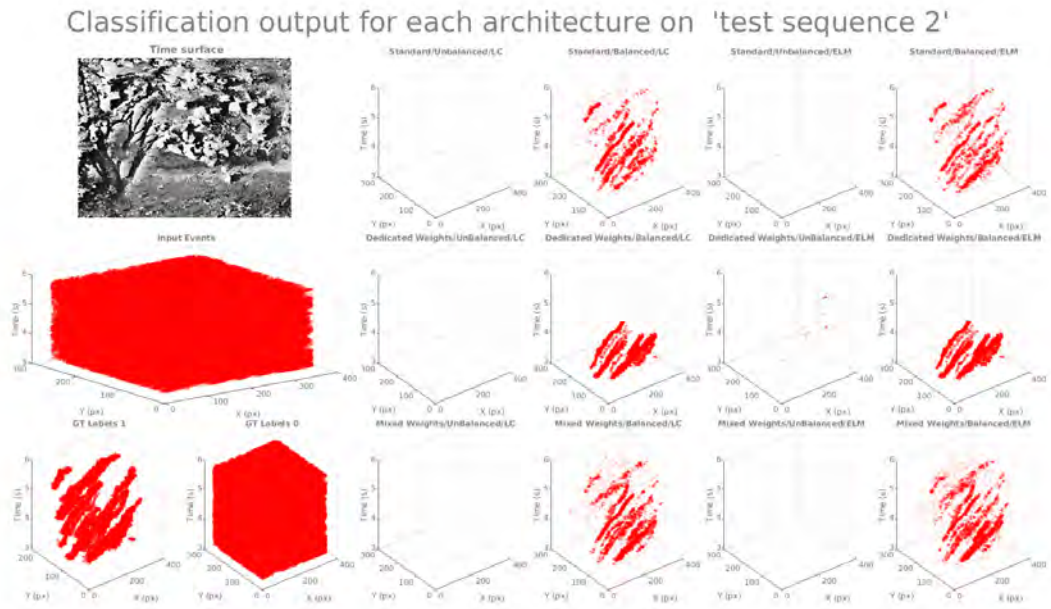


Figure C.15: Real world data "test sequence 2". Each plot shows the classification output for each architecture using both classification methods on the balanced and imbalanced dataset.

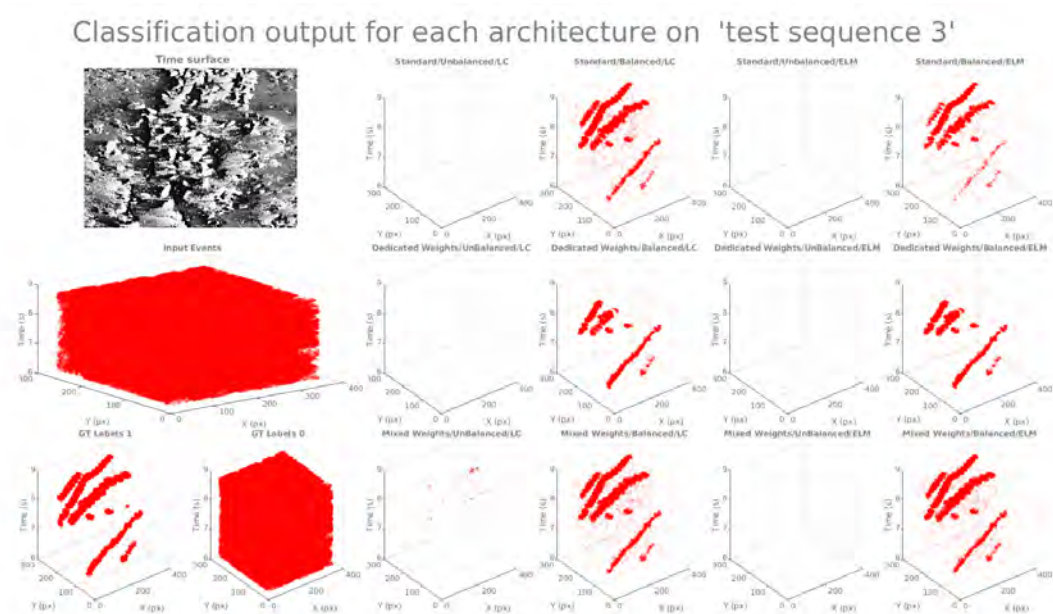


Figure C.16: Real world data "test sequence 3". Each plot shows the classification output for each architecture using both classification methods on the balanced and imbalanced dataset.

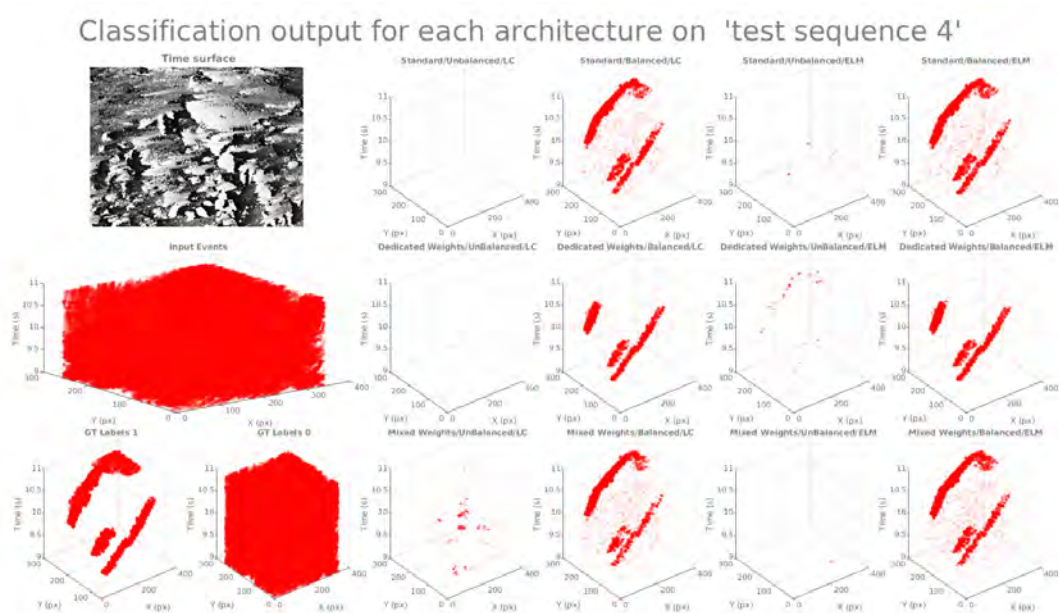


Figure C.17: Real world data "test sequence 4". Each plot shows the classification output for each architecture using both classification methods on the balanced and imbalanced dataset.

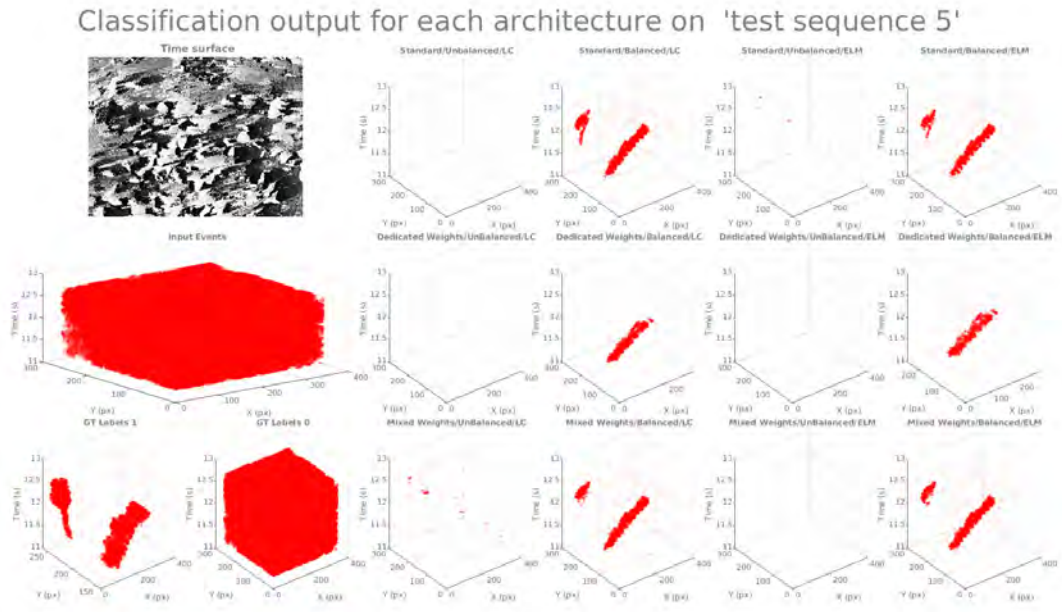


Figure C.18: Real world data "test sequence 5". Each plot shows the classification output for each architecture using both classification methods on the balanced and imbalanced dataset.

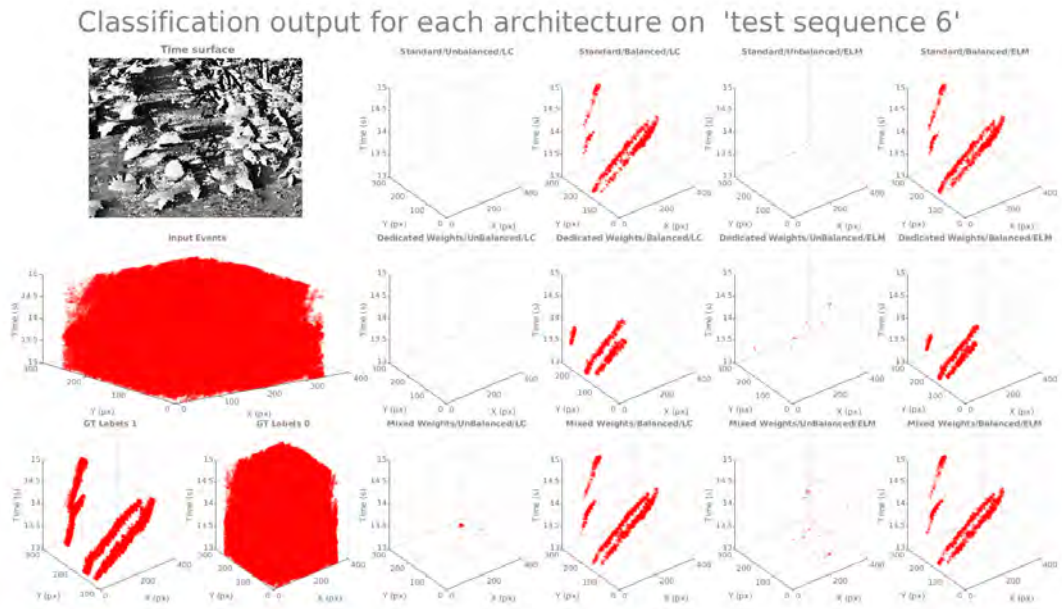


Figure C.19: Real world data "test sequence 6". Each plot shows the classification output for each architecture using both classification methods on the balanced and imbalanced dataset.

Linear Classifier Network

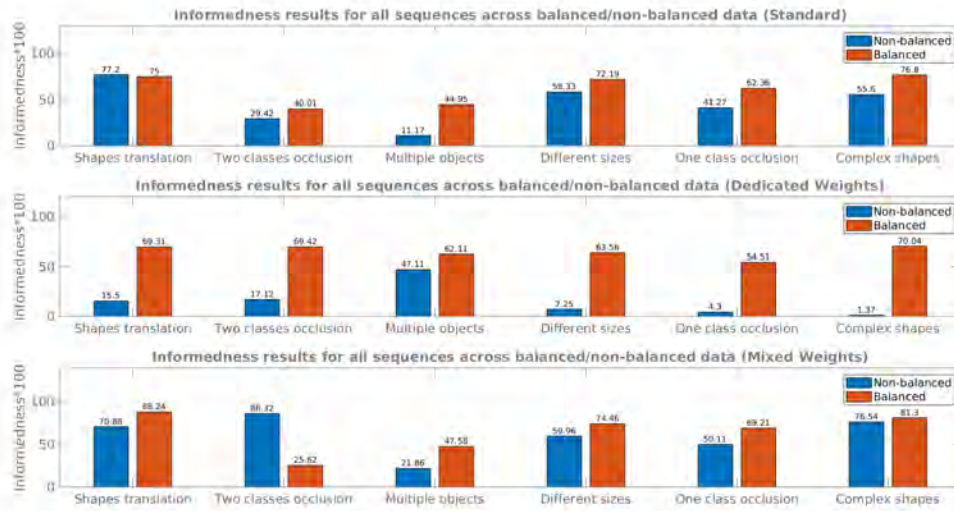


Figure C.20: Comparison of the linear classifier results for balanced and non-balanced classes for each testing sequence using the lab recorded datasets.

ELM Network

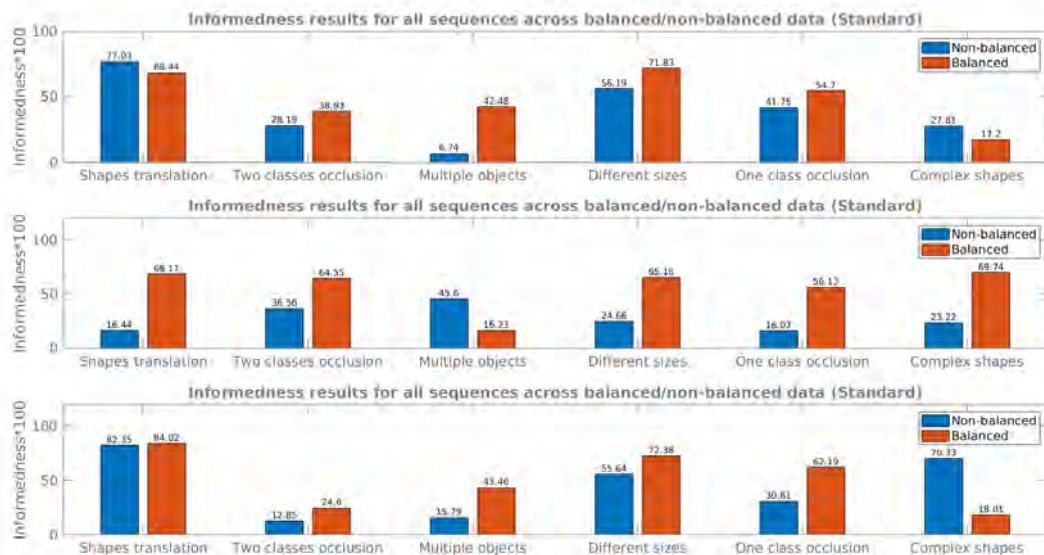


Figure C.21: Comparison of the ELM classifier results for balanced and non-balanced classes for each testing sequence using the lab recorded datasets.

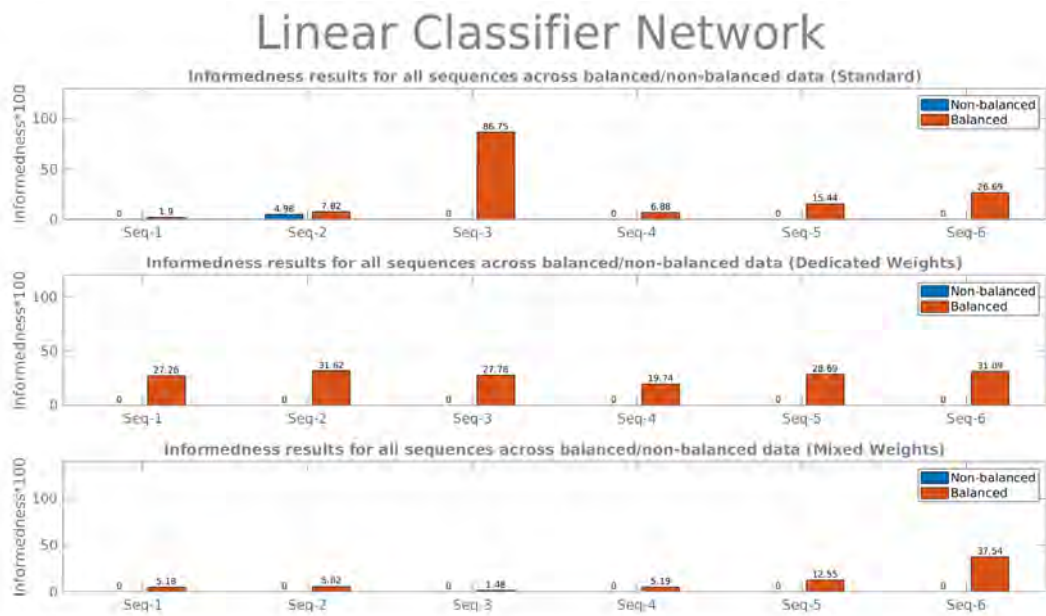


Figure C.22: Comparison of the linear classifier results for balanced and non-balanced classes for each testing sequence using the real-world datasets.

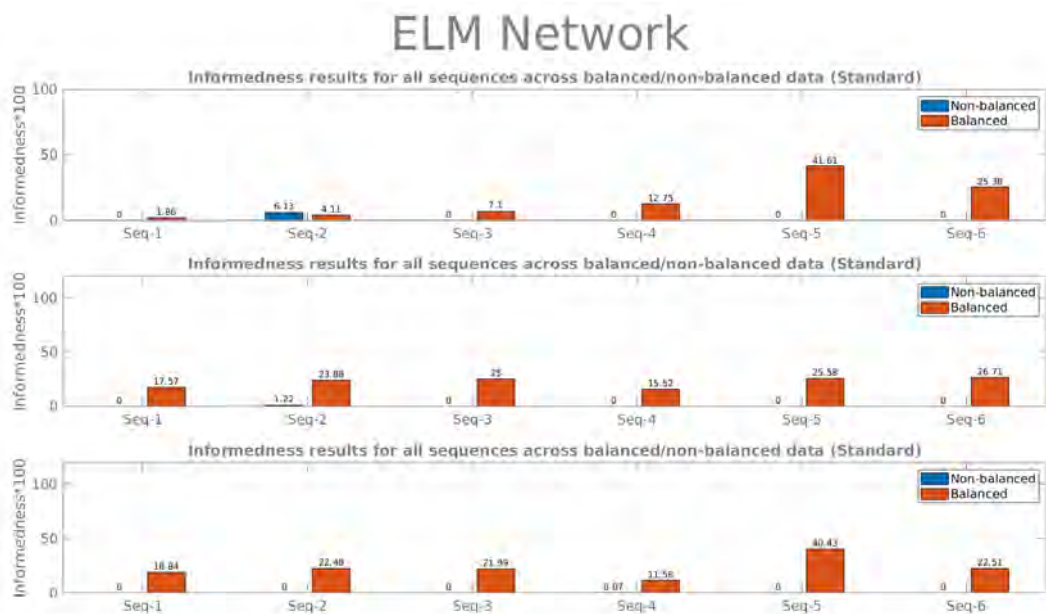


Figure C.23: Comparison of the ELM classifier results for balanced and non-balanced classes for each testing sequence using the real-world datasets.

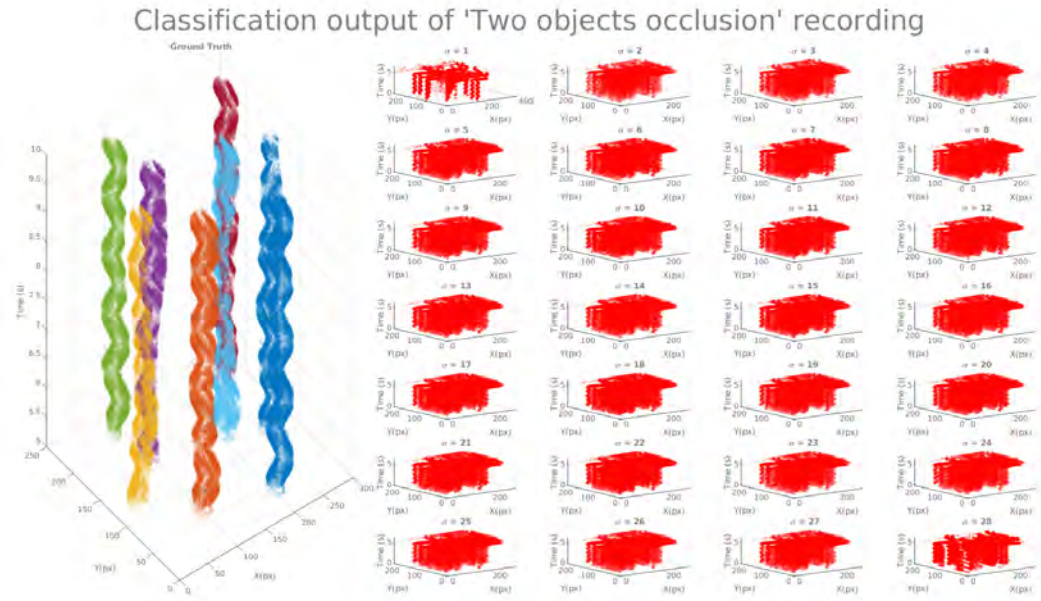


Figure C.24: Space-time plot generated at 28 different spatial resolutions on the "Two object occlusion" test sequence.

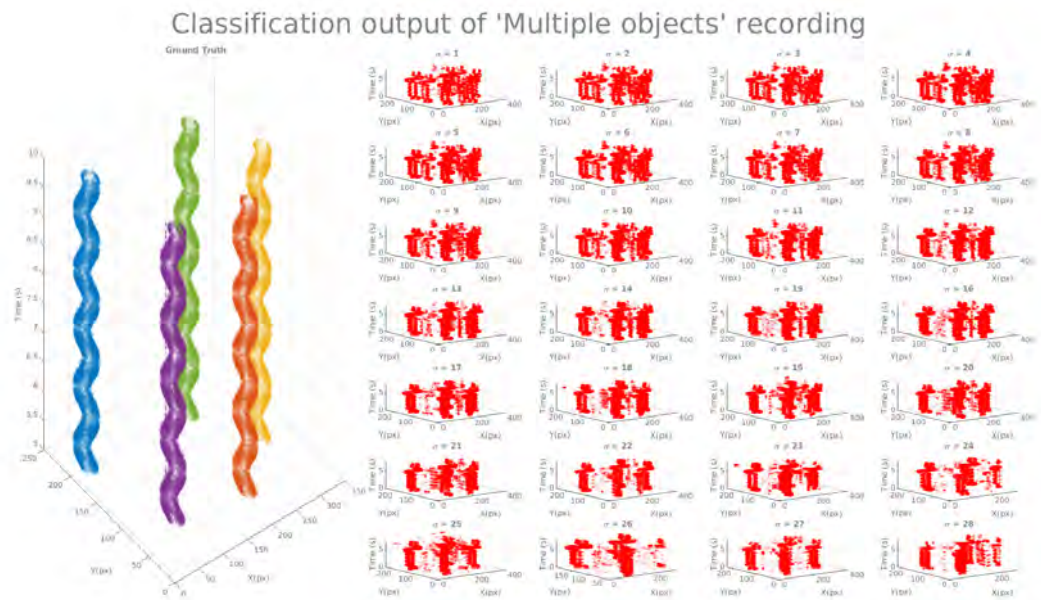


Figure C.25: Space-time plot generated at 28 different spatial resolutions on the "Multiple objects" test sequence.

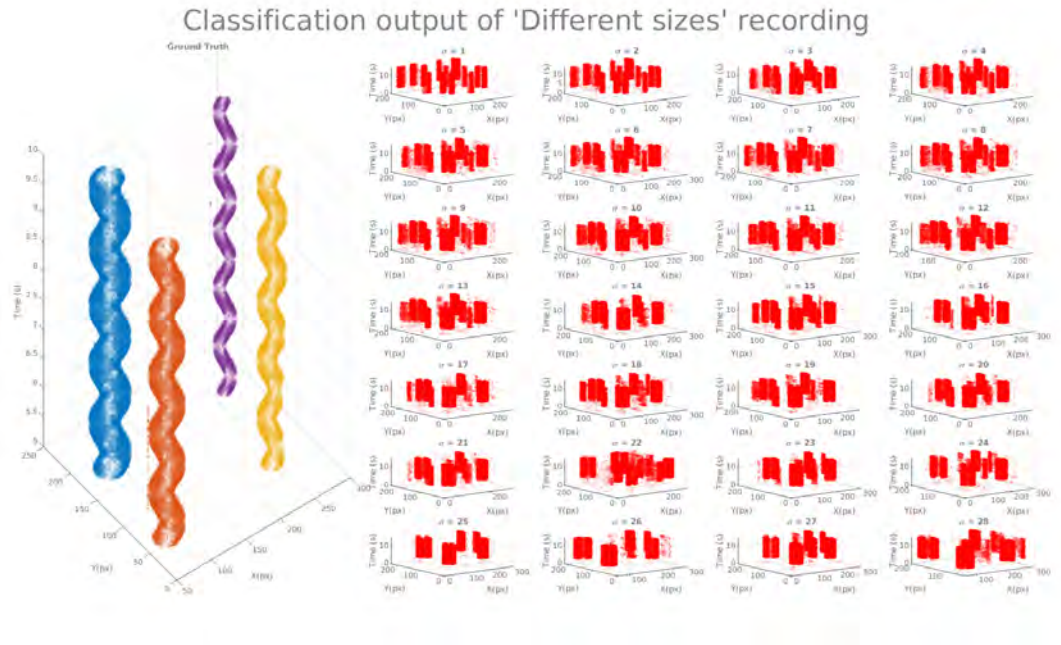


Figure C.26: Space-time plot generated at 28 different spatial resolutions on the "Different sizes" test sequence.

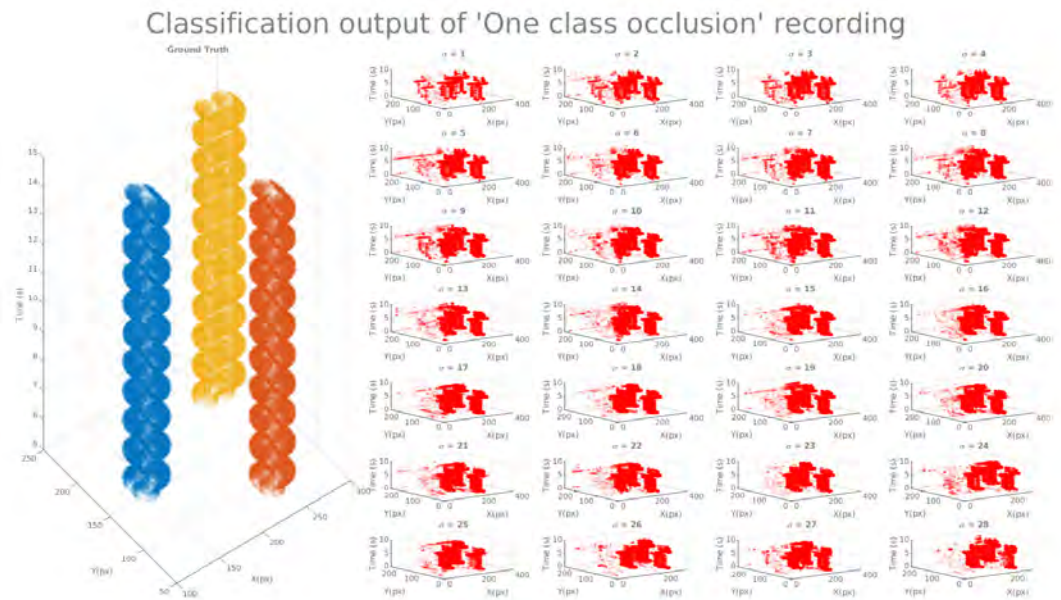


Figure C.27: Space-time plot generated at 28 different spatial resolutions on the "One class occlusion" test sequence.

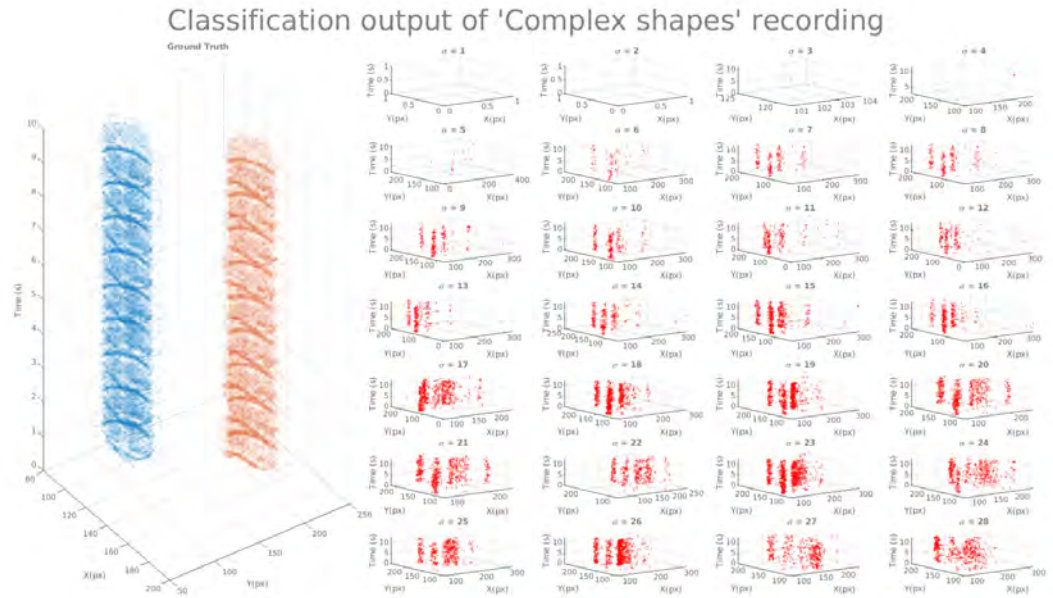


Figure C.28: Space-time plot generated at 28 different spatial resolutions on the "Complex shapes" test sequence.

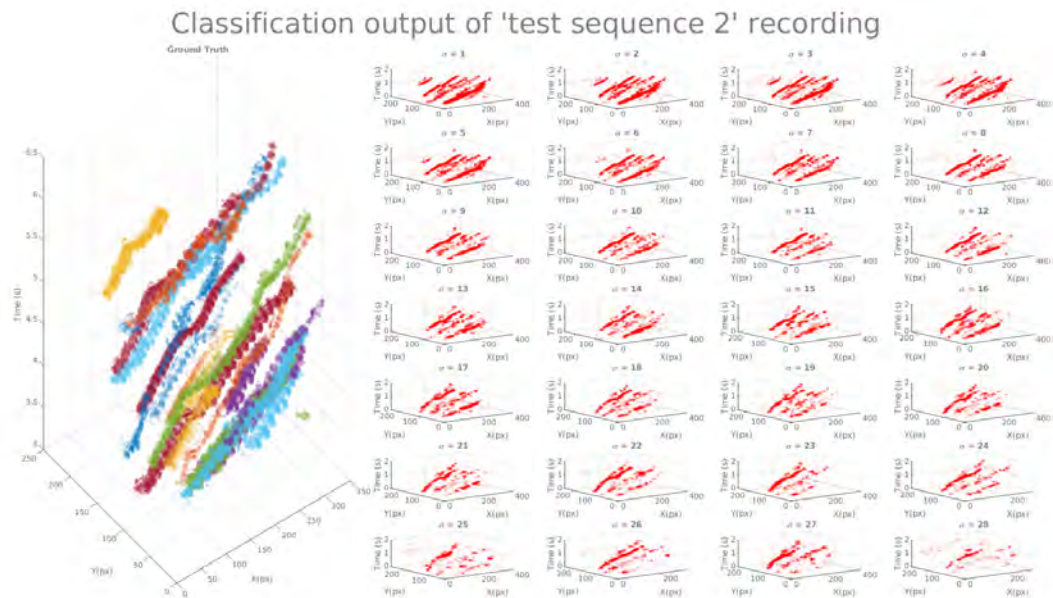


Figure C.29: Real world data "Test sequence 2". Space-time plot generated at 28 different spatial resolutions on the lab recorded data. It shows the classification output for each downsampling value on the test set in 3D space-time.

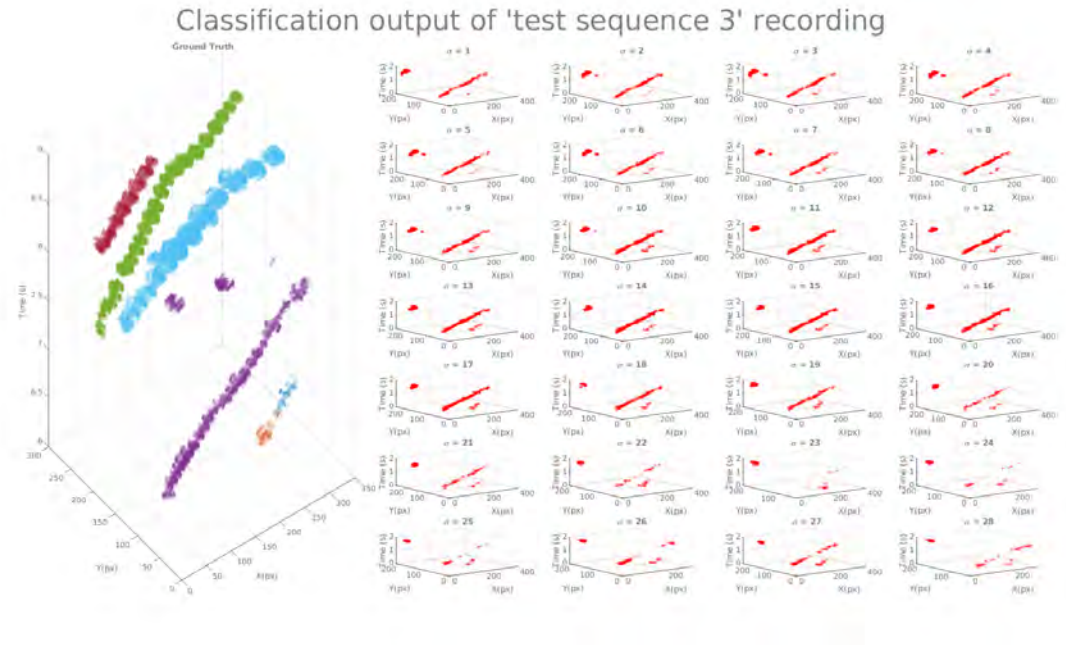


Figure C.30: Real world data "Test sequence 3". Space-time plot generated at 28 different spatial resolutions on the lab recorded data. It shows the classification output for each downsampling value on the test set in 3D space-time.

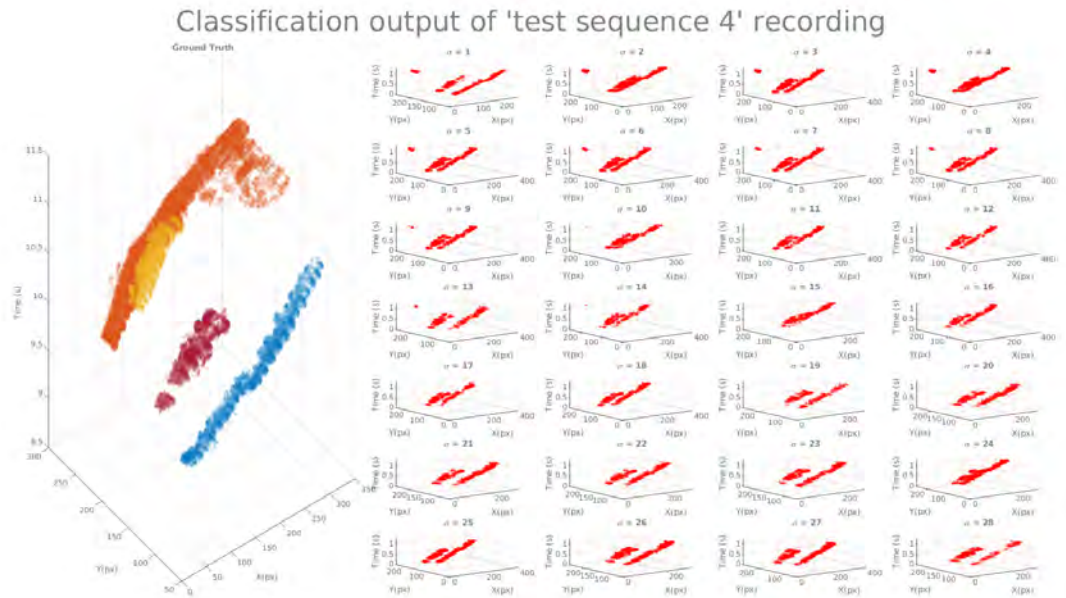


Figure C.31: Real world data "Test sequence 4". Space-time plot generated at 28 different spatial resolutions on the lab recorded data. It shows the classification output for each downsampling value on the test set in 3D space-time.

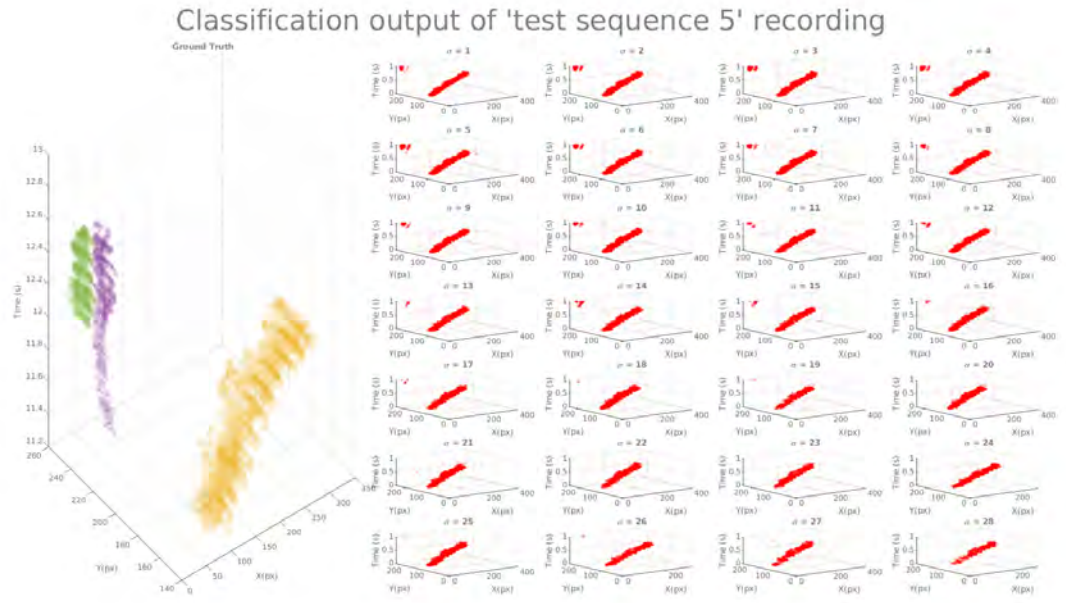


Figure C.32: Real world data "Test sequence 5". Space-time plot generated at 28 different spatial resolutions on the lab recorded data. It shows the classification output for each downsampling value on the test set in 3D space-time.

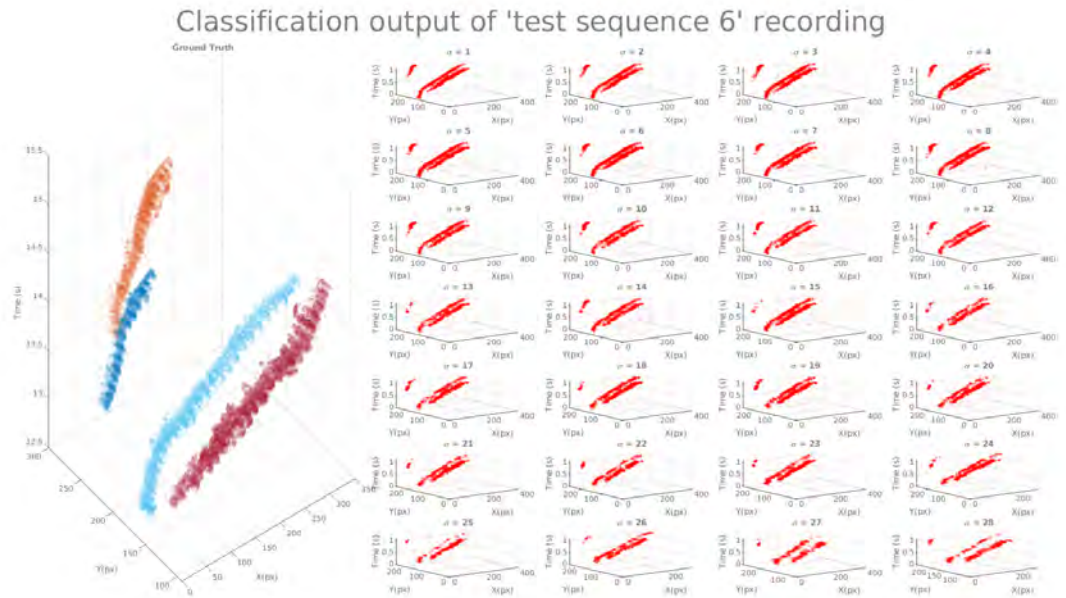


Figure C.33: Real world data "Test sequence 6". Space-time plot generated at 28 different spatial resolutions on the lab recorded data. It shows the classification output for each downsampling value on the test set in 3D space-time.

Appendix D

Representations of Colour Events in Various Scenes



Figure D.1: Time collapsed image in the time dimension with pixel representing the colour events index. Recording shows different geometrical shapes with different colours with horizontal translation in the field of view.

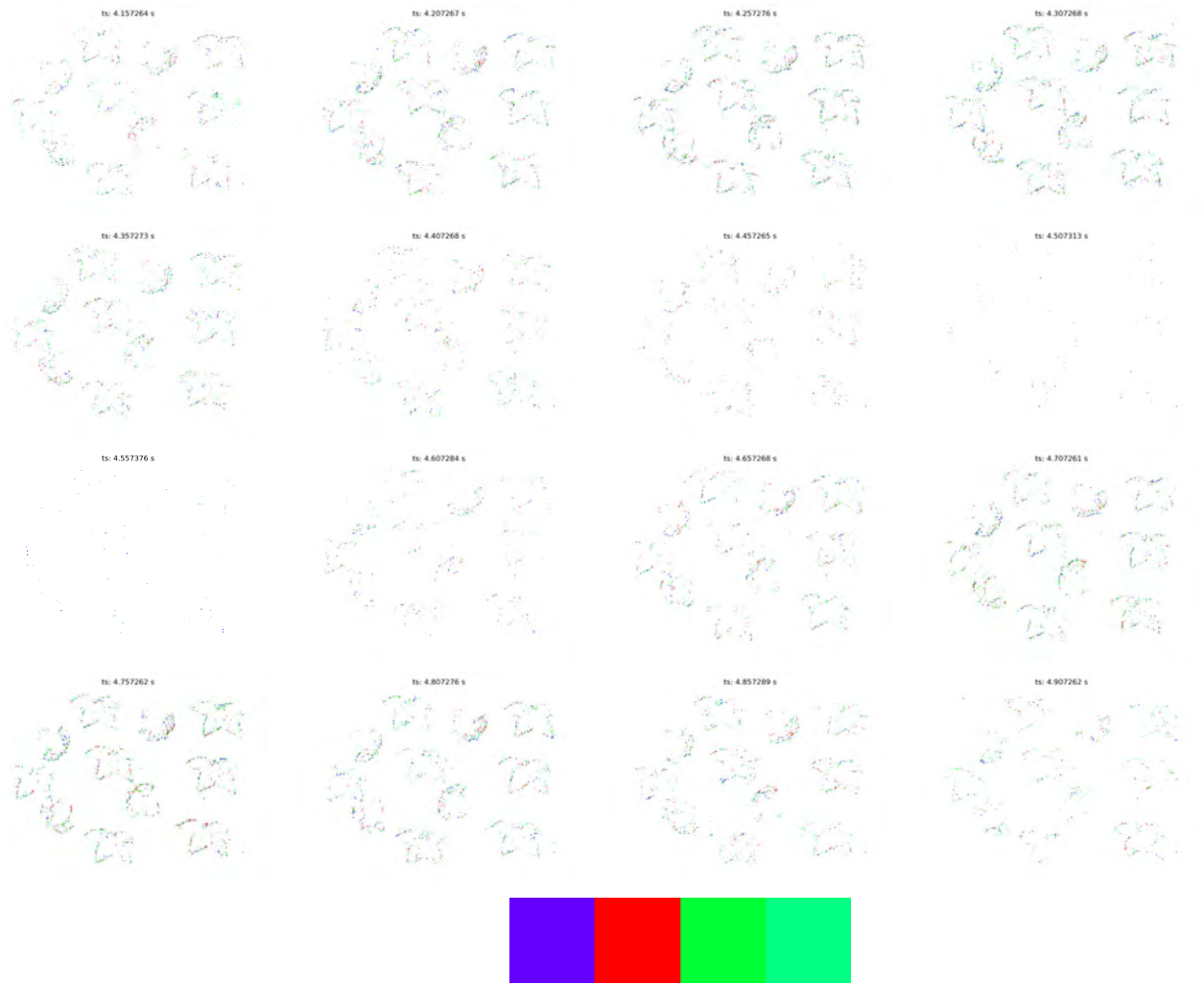


Figure D.2: Time collapsed image in the time dimension with pixel representing the colour events index. Recording shows 2D complex shapes of fruits and leaves with horizontal translation in the field of view.

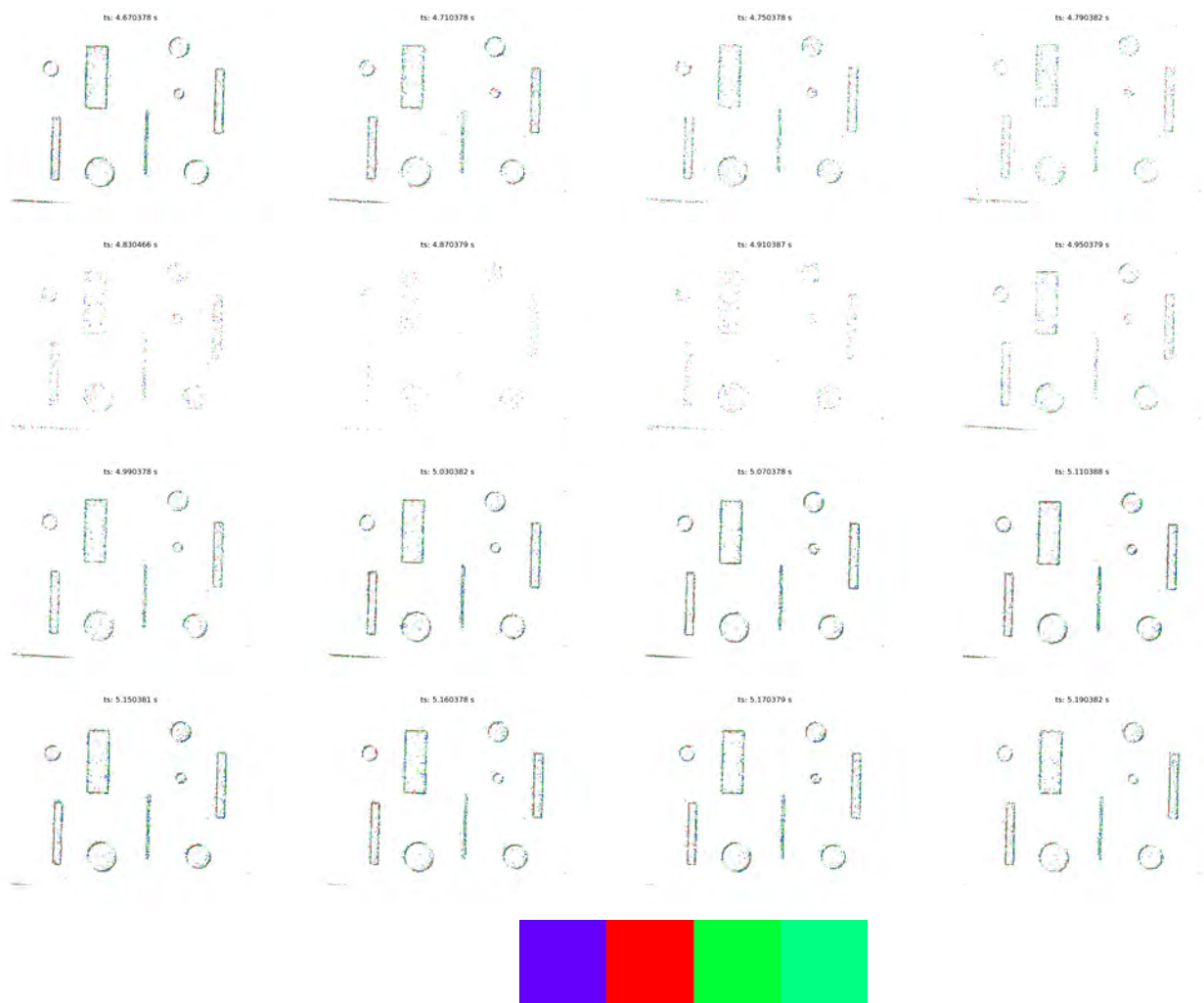


Figure D.3: Time collapsed image in the time dimension with pixel representing the colour events index. Recording shows two classes objects with different size with horizontal translation in the field of view.

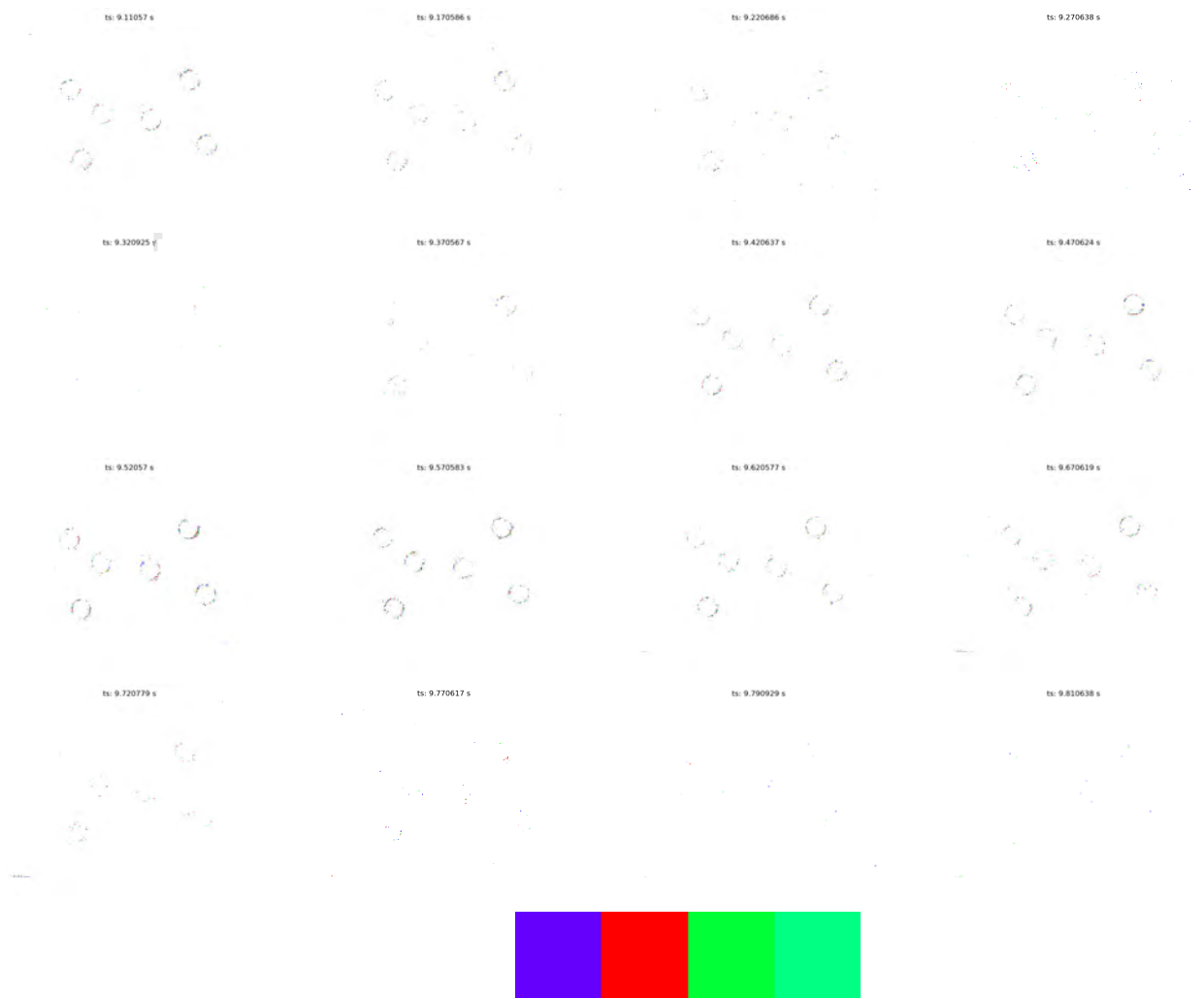


Figure D.4: Time collapsed image in the time dimension with pixel representing the colour events index. Recording shows two classes objects with different size with horizontal translation in the field of view.

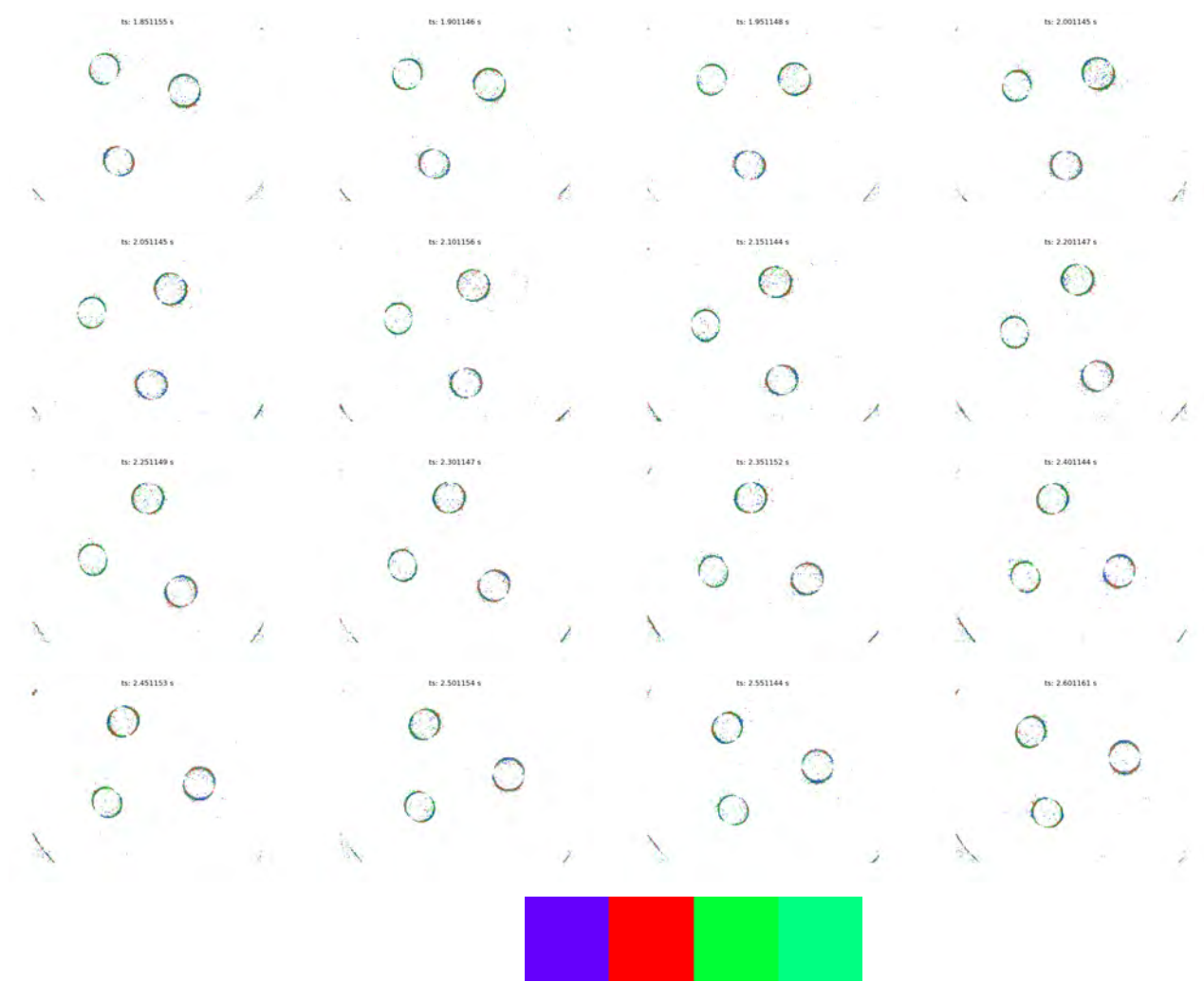


Figure D.5: Time collapsed image in the time dimension with pixel representing the colour events index. Recording shows three circles with different colour in rotation mode.