



Durham E-Theses

Matching Distorted Video Clips to a Reference Video Database

BOULDERSTONE, RICHARD,ANTHONY

How to cite:

BOULDERSTONE, RICHARD,ANTHONY (2025) *Matching Distorted Video Clips to a Reference Video Database*, Durham theses, Durham University. Available at Durham E-Theses Online:
<http://etheses.dur.ac.uk/16002/>

Use policy

The full-text may be used and/or reproduced, and given to third parties in any format or medium, without prior permission or charge, for personal research or study, educational, or not-for-profit purposes provided that:

- a full bibliographic reference is made to the original source
- a [link](#) is made to the metadata record in Durham E-Theses
- the full-text is not changed in any way

The full-text must not be sold in any format or medium without the formal permission of the copyright holders.

Please consult the [full Durham E-Theses policy](#) for further details.

Matching Distorted Video Clips to a Reference Video Database

Richard A. Boulderstone

A Thesis presented for the degree of
Doctor of Philosophy



Department of Computer Science
Durham University
United Kingdom
November 2024

Abstract

The objective of this study is to develop and validate video matching algorithms that satisfy the quality, performance, scalability, and stability requirements necessary for a commercial video matching service. The study assumes that short, distorted video clips are matched against a large reference video database containing at least 10,000 hours of video.

The literature review revealed few studies that came close to the reference video database size, match accuracy and transaction rate requirements used in this study.

Very few large publicly available datasets for partial video copy detection are available making comparisons with other studies difficult. Synthetic and real-world video datasets were constructed to assess the performance of the various video matching algorithms used.

Both machine learning and non-machine learning algorithms are investigated in this study. The matching algorithms extract and compare frame-level feature vectors from the videos to determine a match score. A variety of techniques are then used to determine whether a particular video clip matches a sequence in the reference video database.

Surprisingly, the non-machine learning algorithms performed better than the machine learning approaches with the overall study objectives achieved by a hybrid algorithm. This hybrid algorithm was found to out-perform the throughput and match accuracy of other partial video copy detection algorithms found in the literature.

Declaration

The work in this thesis is based on research carried out at the Department of Computer Science, Durham University, United Kingdom. No part of this thesis has been submitted elsewhere for any other degree or qualification and it is all my own work unless referenced to the contrary in the text.

Copyright © 2024 by Richard A. Boulderstone.

“The copyright of this thesis rests with the author. No quotations from it should be published without the author’s prior written consent and information derived from it should be acknowledged”.

Acknowledgements

Firstly, I would like to acknowledge and thank Vance Ikezoye, formally the CEO of Audible Magic, who originally accepted my suggestion of undertaking research into video matching methods at Audible Magic. Vance provided the initial impetus for this study which helped sustain me through the early years of this project.

Secondly, I would like to thank the team at Audible Magic for continuing to support my research work through challenging times. In particular Erling Wold and Zafar Rafii from the Research Team who have provided valuable suggestions and support along the way. Audible Magic continues to deliver innovative products to the media industry and I am proud of my contribution to their success.

Thirdly, I would like to thank my PhD supervisor Professor Toby Breckon and Durham University for supporting me through my PhD programme.

And finally, I would like to thank Kay Martin, David Boulderstone, Bella (Elizabeth) Boulderstone and Ed (Edward) Boulderstone for being there when I needed them.

Contents

Abstract	ii
Declaration	iii
Acknowledgements	iv
List of Figures	x
List of Tables	xv
Dedication	xvii
1 Introduction	1
1.1 Video Usage Constraints - Copyright	1
1.2 Study Objective	2
1.3 Visual Matching	3
1.4 Digital Watermarking	3
1.5 Video Distortions	4
1.6 Sequence of Images	4
1.7 Audible Magic Identification Services	5
1.8 Summary	11

2	Database and Literature Review	13
2.1	Video Databases	14
2.1.1	VCDB: A Large-Scale Database for Partial Copy Detection in Videos	14
2.1.2	Synthetic Databases	15
2.2	Studies on Video Matching	15
2.2.1	Studies Using Traditional (Non-Deep Learning) Matching Techniques	16
2.2.2	Studies Using Deep Learning Techniques	17
2.3	Video Retrieval	19
2.4	Video Matching in Television Broadcasting	20
2.5	Other Video Matching Related Topics	20
2.6	Critical Summary	21
3	Experimental Methodology and Video Databases	22
3.1	Video Signature Construction	22
3.2	Methods Used In Video Matching Algorithms	24
3.2.1	Locality Sensitive Hashing	24
3.2.2	Feature Vectors and Dimensionality	25
3.2.3	Hash tables, Hash Functions and Distance Measure	26
3.2.4	Locality Sensitive Hashing Based on P-stable Distributions	27
3.2.5	P-Stable Distribution	28
3.2.6	Probability of Hash Table Collisions	29
3.3	Databases Used in This Study	30
3.3.1	Primary Database	31
3.3.2	Synthetic Databases	31
3.4	Summary	35
4	The GRID algorithm	36
4.1	Introduction	36
4.2	GRID Video Signature	37
4.3	Matching Process	37

4.3.1	GRID Algorithm Parameters	38
4.3.2	Blank (a.k.a. Black) Frames	40
4.3.3	Match Candidate Processing	42
4.4	GRID Algorithm Variants	44
4.4.1	Centre of Illumination	44
4.4.2	Phase Correlation	45
4.5	Datasets Used to Test GRID Algorithm Parameters	46
4.5.1	Simulated Distortions	46
4.5.2	Real-World Distortions	47
4.6	Results	48
4.6.1	Simulated Distortion Test	48
4.6.2	Cropping Test	51
4.6.3	Score Match Threshold	52
4.6.4	Frame Rate	53
4.6.5	Grid Size	54
4.6.6	Match Duration and Frame Sequence Match Algorithm	55
4.6.7	Centre of Illumination Test	57
4.6.8	Phase Correlation	59
4.6.9	VCDB Test	62
4.7	Discussion	63
4.8	Throughput	63
4.9	GPU Memory Requirements	64
4.10	Summary	65
5	Spatial Points of Interest	66
5.1	Signal to Noise Ratio	67
5.2	SNR Analysis on Sports Dataset	68
5.3	Baseline for Spatial Points of Interest Algorithms	70
5.4	Summary	73
6	Temporal Points of Interest - The PIXL Algorithm	75
6.1	Computation of the PIXL Video Signature	76

6.2	Matching Process	78
6.2.1	PIXL Algorithm Matching Parameters	82
6.3	Datasets Used to Test PIXL Algorithm Parameters	84
6.3.1	Simulated Distortions	85
6.3.2	Real-world Distortions	85
6.4	Results using Simulated Distortions	86
6.5	Results using Real-world Videos	89
6.5.1	Hash Code Length	89
6.5.2	Number of Bits Set in Hash Code	90
6.5.3	Number of Bit Errors	92
6.5.4	Area of Central Region to Scan	92
6.6	Throughput	93
6.7	GPU Memory Requirements	95
6.8	Summary	95
7	Pre-trained Convolutional Neural Networks	98
7.1	Use of Pre-trained Models in Video Matching	99
7.1.1	Resnet-18	99
7.1.2	ResNet-50	100
7.2	Computation of the Pre-trained Neural Network Video Signature	101
7.3	Matching Process	104
7.3.1	Blank Frames	105
7.3.2	Frame by Frame Matching	105
7.3.3	Multiple Frames per Vector	107
7.4	Results	108
7.4.1	Simulated Distortions	108
7.4.2	ResNet-18	110
7.4.3	ResNet-50	113
7.4.4	Using a Variety of Pre-Trained Networks	116
7.4.5	Multiple Frames per Vector	121
7.4.6	Matching with the Feature Film Dataset	121
7.5	Summary	124

8	Training Convolutional Neural Networks with Triplet Loss	125
8.1	Reference Video Dataset	126
8.1.1	Blank Frames	126
8.2	Creating Triples	127
8.2.1	Alignment	127
8.2.2	Training	129
8.3	Matching	130
8.3.1	Matching Statistics	133
8.4	Results	133
8.4.1	Video Matching Experiments	136
8.5	Summary	139
9	Hybrid Approaches	140
9.1	Combined Algorithm with Feature Film Dataset	141
9.2	Combined Algorithm with Sports Dataset	142
9.3	Combined Algorithms with VCBD Database	146
9.3.1	VCDB Comparison to Other Studies	150
9.4	Throughput	153
9.5	Summary	155
10	Conclusion	157
10.1	Areas for Future Research - Existing Algorithms	159
10.2	Areas for Future Research - Other Algorithms	160
A	Auxilliary Results for Pre-Trained Neural Network Algorithm	170
	Appendix	170
B	Auxillary Results for Training Convolutional Neural Networks with Triplet Loss	183
C	The Sports Dataset	191
D	The Feature Film Dataset	203

List of Figures

1.1	Audible Magic content identification services	5
1.2	Standard distortions types used in this study	9
1.3	Complex distortions types used in this study	10
1.4	Video matching objectives	12
3.1	Processing path for matching reference and unknown videos	23
3.2	P-Stable distribution projections	28
4.1	Computation of GRID feature vector	37
4.2	GRID algorithm, simulated unknowns, standard distortions match performance	49
4.3	GRID algorithm, simulated unknowns, complex distortions match performance	50
4.4	GRID algorithm, simulated unknowns, frame cropping evaluation	52
4.5	GRID algorithm, simulated unknowns, score threshold evaluation	53
4.6	GRID algorithm, simulated unknowns, selected frame rates	54
4.7	GRID algorithm, simulated unknowns, grid size evaluation	54
4.8	GRID algorithm, simulated unknowns, match duration comparison	56
4.9	GRID algorithm, simulated unknowns, frame sequence match algorithm comparison	57

4.10	GRID centre of illumination algorithm	58
4.11	GRID algorithm, simulated unknowns, phase correlation match accuracy, 8 × 8 and 16 × 16	60
4.12	GRID algorithm, simulated unknowns, phase correlation match accuracy, 24 × 24 and 32 × 32	61
4.13	GRID match score threshold on VCDB database	62
4.14	GRID algorithm, real-world videos, processing time vs reference database size	64
4.15	Grid algorithm parameters table	65
5.1	Harris corner detector match results against $\log_{10}(SNR)$ on sports database	71
5.2	Shi-Tomasi points of interest algorithm match results against $\log_{10}(SNR)$ on sports database	72
5.3	SIFT detector match results against $\log_{10}(SNR)$ on sports database . . .	73
6.1	Pixel frame locations for reference PIXL signature	78
6.2	PIXL reference hash table construction diagram	79
6.3	PIXL algorithm, reference pixel locations	80
6.4	PIXL algorithm, reference pixel locations for a rotated image	81
6.5	PIXL algorithm parameters diagram	84
6.6	PIXL algorithm, simulated unknowns, standard distortions match results .	87
6.7	PIXL algorithm, simulated unknowns, complex distortions match perfor- mance	88
6.8	PIXL algorithm, real-world videos, hash code length evaluation	90
6.9	PIXL algorithm, real-world videos, number of bits to match evaluation . .	91
6.10	PIXL algorithm, impact of bit errors on matching accuracy	92
6.11	PIXL algorithm, real-world videos, fraction of central region to scan eval- uation	93
6.12	PIXL algorithm, real-world videos, processing time vs reference database size	94
6.13	PIXL algorithm optimal parameters table	96
7.1	ResNet-18 model structural diagram	100

7.2	ResNet-50 model structural diagram	102
7.3	Pre-Trained neural network, image insertion technique	103
7.4	Pre-Trained neural network, image stretch technique evaluation	104
7.5	Pre-Trained neural network ResNet-18, distance and match score for single match	106
7.6	Pre-Trained neural network ResNet-18, match score threshold vs euclidean distance	109
7.7	Pre-Trained neural network ResNet-18, simulated unknowns, standard distortions match performance	110
7.8	Pre-Trained neural network ResNet-18, simulated unknowns, complex distortions match performance	112
7.9	Pre-Trained neural network ResNet-50, simulated unknowns, standard distortions match performance	113
7.10	Pre-Trained neural network ResNet-50, simulated unknowns, complex distortions match performance	115
7.11	Pre-Trained neural network models, summary results for sports dataset . .	117
7.12	Pre-Trained neural network ResNet-10, simulated unknowns, standard distortions match performance	119
7.13	Pre-Trained neural network ResNet-10, simulated unknowns, complex distortions match performance	120
7.14	Pre-Trained neural networks, feature film dataset matching performance evaluation	122
7.15	Pre-trained neural networks, simulated unknowns, matching performance at 1fps evaluation	123
8.1	Neural network training and triplet construction	128
8.2	Neural network feature computation and matching with reference database using Faiss index	132
8.3	ResNet-18 neural network with a trained final layer, match performance using triples with a frame lag to the negative component of 1,000 frames .	134
8.4	ResNet-18 neural network with a trained final layer, match performance using triples, example 13	135

8.5	Video matching using the ResNet-18 neural network	137
8.6	ResNet-50 trained neural network, match performance	137
8.7	ResNet-18 trained network using frame size of 32×18 , match performance	138
8.8	ResNet-18 trained network using frame size of 16×9 , match performance	138
9.1	PIXL match score threshold versus combined match rate	142
9.2	Combined algorithm, sports database, standard distortions	143
9.3	Combined GRID and PIXL algorithms, sports database, complex distortions	144
9.4	Combined GRID and PIXL algorithms with various match score thresholds	146
9.5	VCDB database, distribution of the number of vidoes in each set	147
9.6	GRID and PIXL match score thresholds on VCDB database	148
9.7	[1] Precision-recall curves for different methods on VCDB dataset.	152
9.8	Combined GRID and PIXL algorithms average processing time	154
A.1	ResNet-18, bi-cubic interpolation, standard distortions, match performance	171
A.2	ResNet-18, bi-cubic interpolation, complex distortions, match performance	172
A.3	ResNet-34, bi-linear interpolation, standard distortions, match performance	173
A.4	ResNet-34, bi-linear interpolation, complex distortions, match performance	174
A.5	DenseNet-121, bi-linear interpolation, standard distortions, match performance	175
A.6	DenseNet-121, bilinear interpolation, complex distortions, match performance	176
A.7	ResNet-10, colour planes @4fps, bi-linear interpolation, standard distortions, match performance	177
A.8	ResNet-10, colour planes @4fps, bi-linear interpolation, complex distortions, match performance	178
A.9	ResNet-10, colour planes @1fps, bi-linear interpolation, standard distortions, match performance	179
A.10	ResNet-10, colour planes @1fps, bi-linear interpolation, complex distortions, match performance	180
A.11	SqueezeNet, bi-linear interpolation, standard distortions, match performance	181

A.12 SqueezeNet, bi-linear interpolation, complex distortions, match performance	182
D.1 View video sidecar application - The Abyss (1989)	204
D.2 View video sidecar application - Blinded by the Light (2019)	204
D.3 View video sidecar application - The Darjeeling Limited (2007)	205
D.4 View video sidecar application - Deadpool 2 (2018)	205
D.5 View video sidecar application - Dream Horse (2020)	206

List of Tables

3.1	Standard video distortions	33
3.2	Complex video distortions	34
5.1	Sports dataset, average SNR for simulated distortions	69
6.1	Computation of PIXL bitmap	77
7.1	Composition of ResNet-50 model	101
7.2	Pre-Trained neural network models, summary results for sports dataset . .	116
7.3	Mutliple frames per vector results	121
8.1	ResNet-18 trained neural network, best discriminator model parameters .	136
9.1	Match statistics for VCDB dataset using combined GRID and PIXL algo- rithms	149
9.2	[2] Comparison with existing methods on VCDB dataset	151
9.3	[3] Comparison of performance of our proposed method at segment-level in the VCDB core dataset.	153
9.4	Combined GRID and PIXL results on different datasets	155
9.5	Combined GRID and PIXL results on different datasets	155
10.1	Overall video matching results by algorithm	159

C.1	Examples of distorted images	202
-----	--	-----

Dedication

To my wife, Kay Patricia Martin, without whom I would not have completed this thesis.

CHAPTER 1

Introduction

Video content is being created, stored, distributed and consumed at an unprecedented rate. Various sources estimate that several hundred hours of original video are being uploaded to YouTube every minute [4]. Combined with other video indexing and sharing services an estimated 42,000 petabytes of new video content are created every month [5].

With more than 2 billion logged-in monthly users YouTube is the most popular online platform in America [6] a service that did not even exist at the start of the millenium. Since YouTube is a platform that exclusively provides access to video content there is a clear growing demand and usage of this type of content.

A survey of online marketers [7] found that 60% of businesses use video as a marketing tool, 36% of marketers make videos a few times a week and 94% of marketers who use video, plan to continue.

1.1 Video Usage Constraints - Copyright

While many content creators encourage free, unconstrained use and redistribution of their video content, other content owners want to control the use of their video content for financial reasons. In particular the film industry is a multi-billion dollar industry that

ferociously protects its creative content [8, 9].

The rights of content owners are enshrined in copyright legislation enacted by virtually all countries around the world [10]. In the UK, films are protected for 70 years from the end of the calendar year in which the principal director, author or composer dies [11].

However, significant revenues of copyright protected works are lost to piracy [12–14] every year. The social networking platforms (Facebook, Google, TikTok, &etc) that allow their users to upload video¹ content are required to identify and remove copyrighted content or risk legal action [15]. Copyright legislation generally protects the original work and any derivative works where little or no additional intellectual effort has been expended in the derivative works creation [16]. Therefore, it is incumbent on the social networking platforms to identify copyright protected video content that their users are attempting to upload and take appropriate action. These actions include refusal to host the content, seeking a licence from the content owner or charging the user a royalty payment, in all cases identification of the underlying copyrighted video content is required.

Increasingly content owners are seeking to protect shorter and shorter video clips particularly as platforms such as TikTok encourage users to upload sub-10 second video clips[17].

1.2 Study Objective

The overall objective of this study is to create and validate video matching algorithms that meet the quality, performance, scale and stability criteria required to create a commercial video matching service. The definition of these requirements is described in detail below.

This study is only concerned with the visual component of video matching. It is assumed in this study, that the audio channel of a video clip is either unavailable or has been replaced and is therefore not useful as part of the matching process. The typical scenario for this type of visual matching are situations where the audio channel has been replaced with commentary in a different language such as for a sporting event or a film that has been dubbed into another language. Commercially there are also opportunities

¹The term video is used in this study to cover all audio-visual content including feature films, television shows, instructional videos and other visual media.

where the customer requires the audio and visual portions of the video to be matched separately.

This study is driven primarily by commercial considerations. The work is sponsored by the employer of the author (Audible Magic Corporation). The expectation of Audible Magic is that a commercially viable video matching service will be created based on the work presented here.

1.3 Visual Matching

This study will focus on the visual rather than the audio content contained within videos. While matching the audio track of a video is generally easier than matching the visual content this does not provide the copy protection assurance that video content owners require - it is a relatively simple matter to replace, eliminate or distort the audio track of a video so that it cannot be matched with the audio track from a reference video. The audio track of a video can be replaced or distorted for both legitimate or illegitimate reasons including foreign language commentaries of sporting events; foreign language dubbing of feature films; secondary dialogue included for instructional, humorous, critical or other reasons and obfuscation to avoid copyright controls. In this study all mention of video, film or feature films refers to the visual part of the media only unless otherwise stated.

1.4 Digital Watermarking

Digital watermarking is a technique used to embed a digital fingerprint or watermark into a piece of media (in this case video) so that ownership of the content can be determined. Generally content owners do not share information about digital watermarks in their content, even if present (for obvious reasons), so a digital watermarking approach to video content matching is not a viable solution to the challenges posed by illicit use of copyrighted material [18].

1.5 Video Distortions

Matching an unknown video clip to a specific reference video from a large video database is challenging because of both the size of the video reference database and the combination and variety of distortions that an unknown video clip may have. Each reference video maybe several hours in duration with copyright owners expecting notification of any 30 second clip that is reproduced by a third party. Reference video clips maybe distorted for legitimate or illegitimate reasons. The frame size, frame rate, colour model or other video features may be changed to conform to an internal publishing standard, while these same features may be manipulated to avoid copy detection algorithms.

Video clips can be distorted in a variety of ways enabled by the ubiquity of software tools that support video manipulations (ffmpeg [19–22] & etc). Example distortions that can defeat simple copy detection algorithms include the introduction of noise, blurring, cropping, rotations, overlays, mash-ups, spliced and/or stitched to form derivative works.

1.6 Sequence of Images

The visual content of a video can be thought of as a continuous sequence of images or frames. Most current video content is created at 25 or 30 frames per second (fps) in colour. The pixel differences between consecutive frames of a video are generally quite small since only around 40 milliseconds separates one frame from the next. However, larger changes between consecutive frames can occur when scene changes are made in feature films or when separate cameras are used at sporting events. The frame where this large visual change occurs is often called a key frame. Many video matching techniques rely on finding these key frames and using them for the basis of matching algorithms. Unfortunately the mathematical models created to define key frames are not precise and do not necessarily produce key frames at the rate required to match a short video [23]. Therefore this study will sample the video at regular time intervals rather than attempting to find key frames in the reference videos.

While a number of techniques have been developed for image matching over the last few years (Scale Invariant Feature Transform (SIFT) [24], Speeded Up Robust Features (SURF) [25], BLOB [26], Template matching [27], BRIEF [28], ORB [29], etc. [30])

these techniques are computationally expensive particularly for large image databases. This is because a typical reference image database might contain tens of thousands of images while a reference video database might also contain tens of thousands of videos, but this may represent over a billion images.

1.7 Audible Magic Identification Services

Audible Magic customers are primarily internet platform companies, for this discussion an internet platform is a company that allows it users to upload media onto the internet platform web site. These internet platforms typically provide services around this uploaded content that allow users to share, annotate, comment on, rank, modify and otherwise use this content. The process of providing access to this uploaded media to other users is effectively a publishing process that attracts the interest of the rights holders of the underlying media. Many rights holders object to the unsolicited use of their content as it may cause them to lose revenue. To avoid conflict with the rights holders many internet platforms choose to screen this content prior to making it accessible on their web site. Audible Magic provides services that match audio and video media against a vast database of registered audio and video digital signatures derived from the underlying media.

Audible Magic Content Identification Services

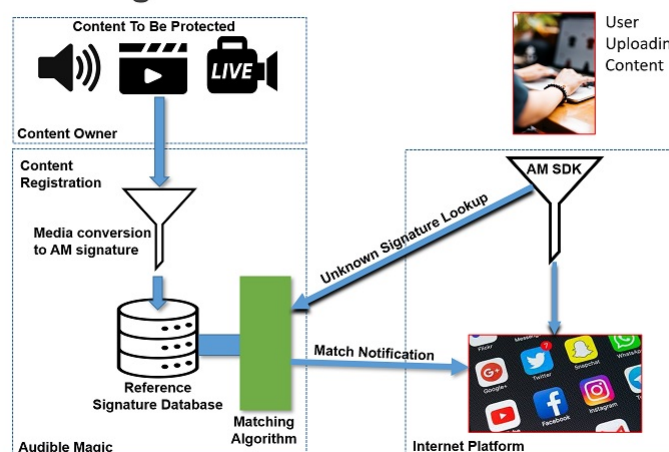


Figure 1.1: Audible Magic content identification services

Figure 1.1 illustrates how Audible Magic processes audio and video content. The content owners (who do not pay Audible Magic) register audio tracks and video content

(films, videos, broadcast streams). This content is converted into Audible Magic signatures as the content owners generally do not want Audible Magic to hold copies of the original media files. An Audible Magic signature is a highly condensed representation of the underlying content. For audio content the signatures typically contain less than 100 bytes per second of data to represent the underlying audio stream. The signatures are stored in a reference signature database. Depending on the service specification, a set of servers will read a sub-set of these reference signatures into local high-speed storage. These servers provide the matching service/algorithm as shown in the diagram to determine whether uploaded content is contained within the reference signature database.

When a user, of an internet platform that uses Audible Magic (AM) services, attempts to upload a media file the AM matching service is invoked. Firstly, the media file is converted into an AM signature using a local AM software development kit (SDK), the resulting signature is then sent to the matching service at AM where the matching algorithm attempts to match it against the AM reference signature database. Any matches that are found are reported back to the internet platform. If a match is found the internet platform determines the appropriate course of action, from upload refusal (citing protected content) to quarantine (if uploaded claims ownership) otherwise the content is posted onto the internet platform service.

Scale and Throughput Objective

The requirement for the throughput rate of the matching process is that the time taken for an average lookup transaction should be less than a second, which translates into a throughput rate of at least one transaction a second. This transaction rate should be achieved against a reference database of approximately 50,000 hours of video. The 50,000 hours of video is a relatively arbitrary database size but provides sufficient capacity to store approximately 25,000 films (assuming an average of 2 hours duration). The current approximate global output of feature films is 4,000 per year according to Stephen Follows, Film Data and Education web site [31].

The query video clip, described as the unknown in this document, is the video clip that is matched against the reference database. This query/unknown clip is assumed to be at least 30 seconds in duration. Typically, the middle 30 seconds of the unknown clip is used

to perform the lookup in the reference database. The assumption is that this 30 seconds is representative of the entire unknown clip - some customers are more demanding and require iterative lookups across the entire unknown; however the general case is to perform one lookup for each unknown clip. This lookup process constitutes a single transaction. The one transaction per second rate must be achieved on a standard Linux-based server with 8 GPU cards at a cost of approximately \$25,000. This requirement implies that the 50,000 hours of video signatures must also be accommodated in a single server. Since 50,000 hours of video is 180,000,000 seconds and the available GPU memory capacity for each system is approximately 64 GBytes - the maximum available GPU memory is approximately 500 bytes per second of video. A typical video may have a frame size of 640×480 with 3 colour channels and a frame rate of 30 fps (frames per second) these dimensions would create a raw video size of approximately 28 Mbytes per second of video, depending on the video encoding method. Therefore, the ratio between the size representation of the video in GPU memory and the raw video size is approximately 1:50,000. Therefore only matching techniques that can use very compact representations of the video can be considered. For example, a typical feature vector of an image from a pre-trained neural network (Resnet-18) contains 512 features coded as 32 bit floating point numbers and occupies 2,048 bytes - meaning that one of these feature vectors could only be stored in GPU memory for every 4 seconds of video.

The transaction rate objective described here is primarily an internal Audible Magic cost issue as the longer each transaction takes the more servers will ultimately be required to service all the customer transactions. In a real-world service the transactions are not distributed evenly throughout the day and normally peak in the early evening. However, queuing of transactions to reduce the overall transaction rate is discouraged as the overall customer (end-user) perception will be of a slow, unresponsive service.

AM customers have some sensitivity to transaction time as generally the end-user will be uploading content interactively and will have to wait for the combined upload and AM matching process to complete before the uploaded content is registered with the internet platform.

While meeting the throughput and scale requirements represents the desired case, some flexibility in meeting the required transaction rate is available. For example, if it

can be shown that higher match quality can be achieved at half the transaction rate this may be an acceptable trade off.

Match Quality Objective

The video matching process (lookup process) between an unknown video clip (the uploaded clip) and the reference video database should return more than 90% true positives (matches) and fewer than 0.1% false positives (type 1 errors). This is based on a 50,000 hour reference video signature database and a 30 second unknown video clip. This allows 10% false negatives (type 2 errors) as these errors are less important to most customers.

Matching an unknown video clip to a specific reference video from a large video database is challenging because of both the size of the video reference database and the combination and variety of distortions that an unknown video clip may have. Each reference video may be of several hours duration with copyright owners expecting notification of any 30 second clip that is reproduced by a third party. Reference video clips may be distorted for legitimate or illegitimate reasons. The frame size, frame rate, colour model or other video features may be changed to conform to an internal publishing standard, while these same features may be manipulated to avoid copy detection algorithms. However, the video matching algorithm does not need to match clips shorter than 30 seconds duration or clips where frames are missing. While it would be desirable to match this type of fragmented video clip against the reference database this is not a current commercial requirement and this study does not include techniques to match such fragmented clips.

Video clips can be distorted in a variety of ways enabled by the ubiquity of software tools that support video manipulations. Example distortions that can defeat simple copy detection algorithms include the introduction of noise, blurring, cropping, rotations, overlays, mash-ups, spliced and/or stitched to form derivative works.

The standard distortion types used in this study are shown in Figure 1.2.

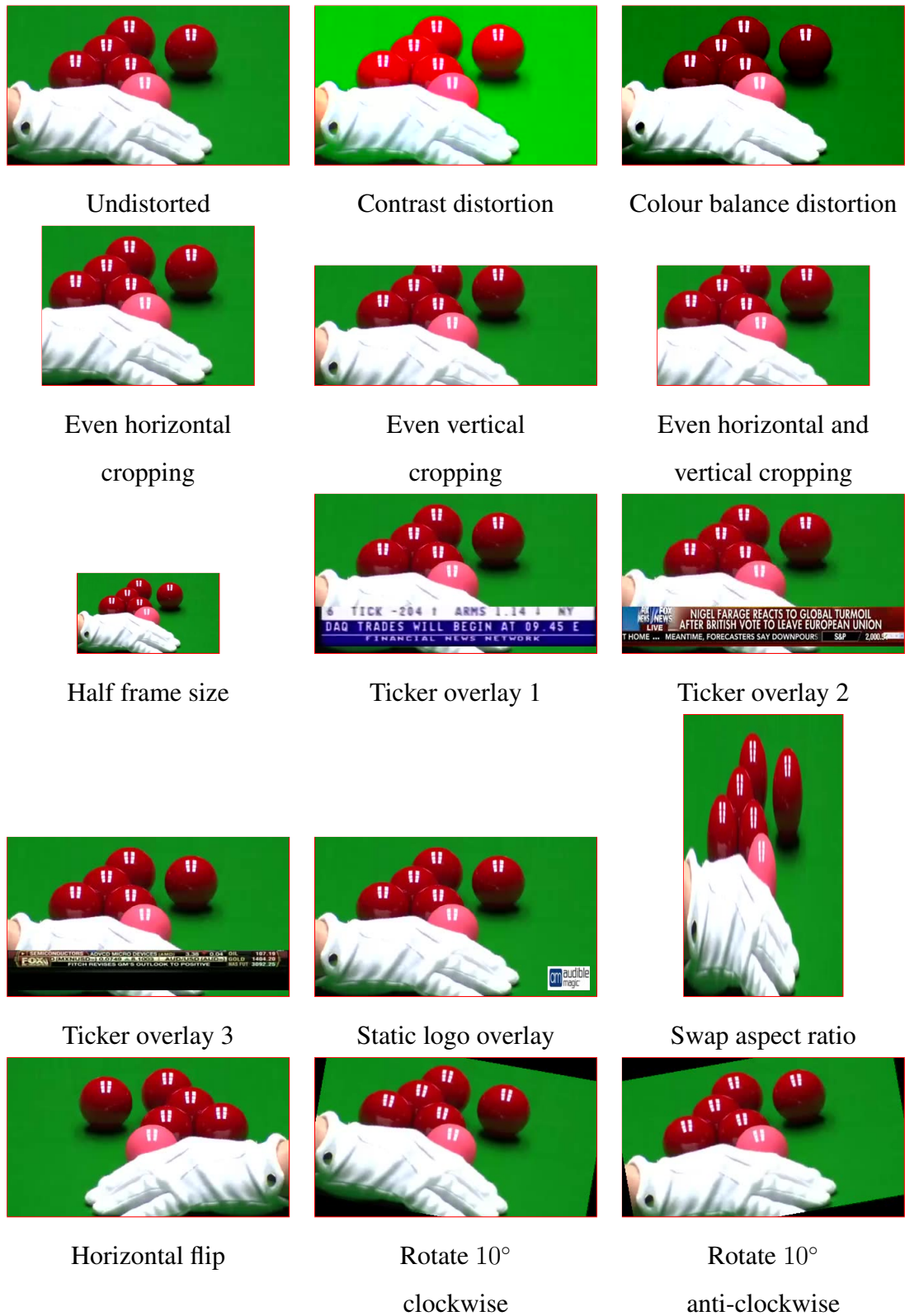


Figure 1.2: Standard distortions types used in this study

The complex distortion types used in this study are shown in Figure 1.3.

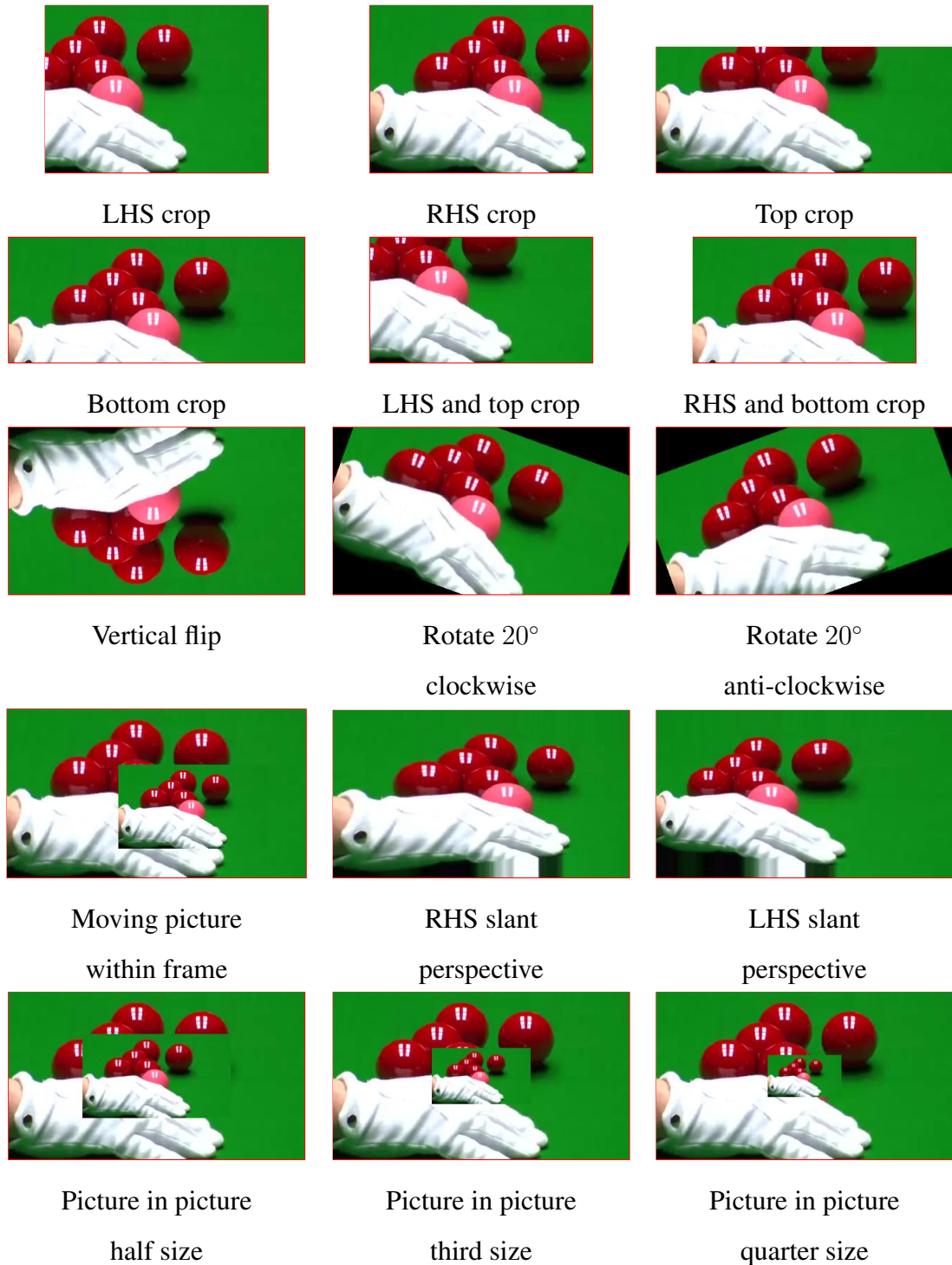


Figure 1.3: Complex distortions types used in this study

The video matching process should be robust to the standard distortions described in Figure 1.2. These standard distortion types will be included in the match performance

statistics. We are not concerned in this study with distortions that make the resultant video unwatchable. The primary goal is to detect partial copies of a reference video that might be reasonably consumed by another user, not videos that have been distorted beyond the point where they are recognisable or painful to the eyes to watch. However, it is also desirable that the matching algorithm find the complex distortions shown in Figure 1.3. These complex distortions will also be used in this study to determine the precision of the matching process.

Video Content Type

Since this study is driven primarily by commercial considerations the video types used will be those which have financial value and which the content owner wants to protect. These include feature films including animation, recent sporting events and some broadcast television shows.

Stability Objective

The video matching service will operated 24 hours a day 7 days a week. A service up-time of 99.99% is guaranteed to AM customers with financial penalties for failure to meet this standard. The 99.99% standard represents less than one hour of downtime per year. Scheduled maintenance is allowed once a month and does not contribute to service downtime.

1.8 Summary

All requirements are based on a reference video database of approximately 50,000 hours of video. Query videos (unknowns) must be at least 30 seconds duration or longer. However, only one lookup of the unknown will be made against the reference database. The lookup will use the middle (continuous) 30 seconds of the unknown video clip. The match statistics are based on finding the 30 seconds of unknown video within the reference video database.

The objectives of the video matching algorithm are presented in Table 1.4.

Objective	Desired Standard	Minimum Acceptable Standard
Database size	50,000 hours	10,000 hours
Hardware	Linux server with 8 GPUs	Linux server with 8 GPUs
Unknown video clip duration also lookup duration	>30 seconds	>40 seconds
Throughput rate	<1 second per transaction	<1 second per transaction
GPU memory per second of video	<500 bytes	<1,000 bytes
True positive rate	>90%	>80%
False positive rate	<0.1%	<2%
Distortions	all distortions	standard distortions
Uptime	>99.99%	>99.99%

Figure 1.4: Video matching objectives

A distinction is drawn here between the objectives that are desired (in the second column of the Table 1.4) and those that represent a minimum acceptable standard (in the third column of the Table 1.4).

These objectives are drawn from product marketing expertise within Audible Magic, based on knowledge of potential customers, and may be ultimately incorrect. The competitive landscape may also change, during the course of this study, which may also shift the target for a viable service. However, this set of objectives frames the scale of the challenge for Audible Magic to create a competitive service and will be used in this study as the benchmark to be achieved rather than the state of the art described in the scientific literature.

CHAPTER 2

Database and Literature Review

Over the last few years significant advances in the areas of image classification, object detection and image segmentation have been made [32–34]. These have been driven by a new range of neural network models (AlexNet [35], VGGNet [36], ResNet [37], etc.), the availability of large image classification databases (ImageNet [38] with 15 million images over 22,000 classes) and the emergence of relatively low-cost high-performance hardware, particularly Graphics Processing Units (GPU).

Using these developments in image analysis, efforts have been made to leverage these results into the area of video analysis [39, 40].

However, these advancements in video analysis have come at the cost of computational performance, achievable only through the aggressive use of the latest parallel processing technology. Since computational performance is a key objective of this study, as described in Section 1.4, both contemporary deep learning and traditional (non-deep learning) approaches to video matching will be explored.

In addition, few studies in the literature have attempted to scale their results to databases that contain more than 1,000 hours of reference video. While no studies were found that attempted to search a video database of 10,000+ hours, which is the minimum acceptable database size required by this study as described in Section 1.4. The size of the reference

database is important because performance tends to decline, at best, linearly with database size. Therefore, the existing literature does not provide video matching methods that meet the scale or transactional performance objectives of this study.

Large reference database sizes also have an effect on the precision of video matching. As the reference database gets larger, the number of false positives returned by a match algorithm generally increases as a proportion of the number of true positives returned. This leads to a decline in matching precision because:

$$precision = \frac{true\ positives}{true\ positives + false\ positives} \quad (2.1)$$

Therefore, the precision of the match results reported in studies that use small reference databases cannot be easily compared with the accuracy of results performed on a much larger database.

2.1 Video Databases

The primary objective of this study is to detect video clips (a contiguous set of frames, or, as described in the literature - partial copies) extracted from a reference video at a scale and quality that meets the demands of a commercial video matching service. The video clips are assumed to be distorted in a limited way so that the resultant clips are perceived as reasonable high-quality versions of the original reference video. The acceptable distortions by this definition include addition of some noise, colour model changes, contrast changes, cropping of frames, frame rate changes, horizontal flip and aspect ratio changes.

2.1.1 VCDB: A Large-Scale Database for Partial Copy Detection in Videos

A number of studies, focused on the evaluation of video copy detection techniques, use the publicly available VCDB [41] database as the basis of the performance of their method [1–3, 42]. Unfortunately this database does not contain definitive reference videos, rather it contains 28 sets of distorted videos. Where each individual set of videos contains a number of distorted versions of the same event. The videos in each set are not always

in the same frame sequence and some do not have a continuous 30 second duration that matches across all the videos. For example in the `bill.clinton.apology.speech` set, one video only contains 5 seconds of the speech common to the other videos in the set, also in the `dove.evolution.commercial` video, one clip has the same content but is played backwards. Therefore, the match statistics obtained from this database are difficult to compare with match statistics from more consistent datasets. The objective of this study is to identify any 30 second unknown clip from a definitive reference database. Since the VCDB database does not contain consistent continuous 30 second video clips only image matching techniques applied to each frame are likely to achieve high match accuracy rates.

The VCDB database was used in this study but clips which did not contain a full 30 seconds of video common to both reference and unknown videos were discarded.

2.1.2 Synthetic Databases

According to the study [43] most publicly available datasets for partial video copy detection have a small size, have an unbalanced distribution of positive and negative videos and many have synthetic distortions rather than distortions resulting from real-world noise and transformations. Some authors [3, 43] have criticised the partial video copy detection results obtained from these synthetic databases, where near perfect results have been obtained for video matching applications, as not being representative of actual distortions found in the wild. Clearly it is easier to design a video matching algorithm where the types of distortion are known and limited, compared with a matching algorithm that cannot rely on these constraints.

2.2 Studies on Video Matching

In the last ten years there has been a dramatic shift towards deep learning techniques for video analysis in the literature. However, these deep learning techniques have not yet been proven to be superior to non-deep learning (traditional) methods for all video analysis applications. Therefore a brief survey of relevant traditional video matching techniques is presented followed by a survey of deep learning approaches.

2.2.1 Studies Using Traditional (Non-Deep Learning) Matching Techniques

The study [44] attempts to address the scalability and localization issues associated with matching against large video databases (2,316 hours). A compact video descriptor occupies 22 bytes per frame at 25 fps in the reference video database. Local descriptors are initially extracted from individual video frames and aggregated into a single vector for a video segment. This single vector is further reduced in size by multiplication with a sparse matrix or by principle component analysis. The reference vectors are arranged into indexes so that at match time only a subset of the indexes are searched for the query vectors. In the large scale test described in the study an average precision of 0.53 was obtained in a database of 2,316 hours using their modified version of the TRECVID [45] database. The transaction rate obtained from this large scale test was 23.5 times real time (which is taken to be 23.5 times longer than the duration of the unknown video) on non-GPU hardware. All of these performance statistics are considerably outside the minimum acceptable standard required for this study.

A study using hash keys and inverted indexes is [46]. This study uses a frame rate of 10 fps and frame size of 176×144 to extract temporally informative representative images (TIRI) for each frame. These TIRI are binarized and hash codes using a local region mean are extract from them. The hash codes are used for the matching process with the reference video hashes loaded into an inverted file index. At match time, hash codes for the unknown video clip are calculated and match candidate matches are found by searching the inverted index. Match candidates are further analysed to ensure a temporal alignment between the reference and unknown video clips. Ordering and time sequencing are used to reject false positives. A threshold is applied to the final match candidates and a match is declared if the score for the match is higher than the threshold. A sample from the TRECVID 2011 dataset (100 videos) was distorted in 11 different ways to produce a test database. The experimentation produced an average F1 score of 97.8% on this synthetically distorted video database with a transaction rate of around 5 lookups per second. A further experiment using the VCDB database with a set of distraction videos (1855 hours) achieved a precision and recall of 81% and 53% respectively with an average

transaction time of 11.79 seconds.

2.2.2 Studies Using Deep Learning Techniques

Deep learning techniques are widely used in the field of computer vision for matching and classification tasks [47].

Video matching has taken advantage of these advances in a number of ways. Convolutional neural networks (CNN) [48] have been used to derive compact deep global descriptors for translation, scale and rotational invariance [49]; copy detection methods have also used CNN features to encode image content [50] as features to be matched and real-time copy detection has used key-frame selection techniques to match distorted video streams [51]. However comparison between methods is difficult because different studies use a range of video databases (particularly real-world and synthetic), do not always report performance information and have different criteria for success. The following review of individual studies that use neural networks for video matching has been selected from the most recent works in this area.

The study [2] contains many of the ideas explored in this study. These ideas include uniform linear extraction of video frames with time rather than the use of key frames; feature generation for each extracted frame separately; creation of a global feature database to store reference frame feature vectors; off-line computation of reference frame features; the use of FAISS [52] index software for fast retrieval of near feature vector matches; the use of the VCDB [41] database as a matching standard; use of hard negative samples in the neural network training. The work uses several neural network layers to achieve matching using a ResNet base layer to compute the frame-level feature vectors, a transformer encoder was used to improve the temporal representation and a temporal network localisation layer to localize the copy segment.

However, the approach used by W. Tan et al [2] is not practical to achieve the goals for this study as the computational performance, described as the running speed is acceptable, close to real-time does not meet this study target of one second per transaction. Further, the match statistics are presented as F1 scores which hides the individual precision and recall statistics with an F1 score on the VCDB database of 0.88 well below this study objective. Although the limitations of the VCDB database are noted in the section on

datasets used in this study.

An earlier work [1] firstly uses a pre-trained ResNet architecture to extract frame-level features and subsequently used a SiameseLSTM (Siamese Long Short Term Memory) unit to temporally combining the frame-level feature vectors. Video clips were feed into the ResNet model to obtain 2,048 length feature vectors for each frame. Frame-level features were combined by the spatial-temporal fusion provided by the SiameseLSTM network with a contrastive loss function. The VCDB database was used to evaluate the results and an additional 100,000 *distraction video clips* were added to the reference dataset to make the matching test more realistic. This appears to be one of the few attempts to make the overall reference database large enough to tackle real-world problems rather than producing results of purely academic interest. Results were reporting using precision and recall statistics. The matching statistics from this study show that at a 80% precision a 70% recall is obtained for the VCDB database. However, to achieve these results the computer processing time is close to the 60 second query duration, far longer than the minimal acceptable transaction time for this study.

Another study [3] that uses the VCDB database makes the point that simulated videos are relatively easy to match and lists a number of papers that have used simulated data boasting high match rates. The authors point out that simple distortions of a reference video can be easily detected with a range of matching algorithms, while real-world distortions are much harder to match against. The video matching method in this paper uses key frames to generate two descriptors for each short video sequence. These descriptors consist of the intensity of pixels in the downsized key frames (30×30) and binary vectors representing the spatial and temporal information. The temporal descriptor is 98 bits and the spatial descriptor is 26 bits in length. To match the descriptors a Q-learning algorithm is used which is a temporal-difference technique of Reinforcement Learning. Only the VCDB database is used in the experiments (528 video segments) with a transaction time of 0.3 seconds. However, scaling this approach up to the minimum requirements for this study is likely to lead to extreme performance challenges. The method claims a precision recall match rate of 0.88 and 0.74 with only 916 bytes per video shot (2 seconds) which is within our scope. This shows the value of extracting features from a video segment rather than individual frames.

A further study [42] claims the state of the art performance on the VCDB matching task as ($F1 = 0.90$). This study describes the Video Similarity and Alignment Learning (VSAL) approach, combining spatial, temporal and alignment information using a neural network to model spatial similarity. However, no information is provided about the scalability of the method and insufficient detail is provided to reproduce the experiment and results shown. As such this paper is only included for completeness.

2.3 Video Retrieval

A recent literature review of video retrieval techniques [53] highlights some of the differences and similarities between video retrieval and video matching. The paper maps the relationship between information retrieval and its various sub-branches down to near duplicate video retrieval (NDVR) which is shown as a super-set of partial video copy detection which is the focus of this study. The review notes the issues with comparing different copy detection methods and cites the TRECVID artificially generated video dataset which only contains 101 videos for a total duration of 100 hours. The discussion goes on to explain that the difference between near duplicate video retrieval (NDVR) and near duplicate video detection is that NDVR focuses more on accuracy and NDVD focuses on efficiency. In a comparison of traditional and deep-learning methods the review notes that traditional methods have not been proven to express videos very well nor meet the needs of big data processing whereas deep-learning based approaches are more effective. The various traditional image matching techniques (Colour histogram, SIFT, SURF, LBP etc.) are described along with the benefits of CNN [48], RNN [54] and LSTM [55] architectures. The review also points out that most techniques extract features from individual frames rather than from video segments. The remainder of the study is mainly focused on video retrieval methods and databases including inverted indexes, hashing methods and similarity measurements. The size of test datasets is also discussed with the largest being significantly less than 1,000 hours of video content.

2.4 Video Matching in Television Broadcasting

[56] develop an unsupervised technique to detect commercials in broadcast TV. This work computes shot boundaries to detect the beginning of a commercial in a broadcast TV channel, they detect abrupt transition in video streams using a thresholded colour histogram difference technique. Features are extracted from shots using a pre-trained Inception-V3 model trained on ILSVRC-2012 image data sets that yields 2,048 features per frame. Four sets of features from the shot are considered using equidistant sampling of the frames. Finally a support vector machine is trained to classify shot features into commercial and non-commercial video clips. Using this system the authors claim a precision and recall of 93% and 95% respectively.

Another work that deals with TV broadcasting [57] is concerned with finding query video segments in real-time broadcast TV streams. This study uses the Normalised Cross-Correlation to find frames within videos and hence to identify video segments. Frames are rescaled to 32×24 to perform cross-correlations between unknown and reference video segments. A selection from the SVD dataset is used for testing with a set of 27,000 reference videos and 1,000 query videos used for matching. Some synthetic distortions (blurring, compression and contrast) were applied to the dataset which are considered appropriate for TV detection. The study reports an F score of approximately 99%.

2.5 Other Video Matching Related Topics

[58] uses frame level descriptors to train a temporal neural network to accurately align videos in the time domain. The paper points out the state of the art for video matching is achieved through frame-level feature vectors. Their approach is to Fourier transform the feature vectors so that the resultant vectors can more easily be aligned in the time domain. While this work was performed on small datasets the notion of using frame level descriptors for video matching will be used in this study.

2.6 Critical Summary

The literature review has shown that very few studies have worked with a reference video database of more than 1,000 hours of video. This makes a comparison of match accuracy and transaction rates against the objectives of this study relatively meaningless.

As noted, several authors have pointed out that matching synthetically distorted video clips is significantly easier than matching real-world distorted clips. However, synthetically distorted clips can provide a lower limit of match accuracy and will be used in this study for this purpose.

No studies have been identified where lookup transactions take less than one second on a database larger than 200 hours. As the minimum acceptable standard for this study is a lookup transaction rate of less than 3 seconds per transaction on a database of 25,000 hours of video, high performance techniques beyond the scope of the current literature on video matching will have to be investigated.

Virtually all recent studies on video matching have used deep-learning techniques but with relatively small video databases. Deep-learning based studies are primarily focused on frame-level matching at around one frame per second with a typical feature vector of around 512 values per vector. This feature vector size may be incompatible with the objectives of this study (no more than 500 bytes per second).

Overall, the literature provides some partial ideas about how to achieve the objectives of this study as set out in Section 1.4.

CHAPTER 3

Experimental Methodology and Video Databases

This chapter describes the experimental methodology and the video databases used in this study. The following chapters use this methodology to analyse individual video matching algorithms. While the video matching algorithms described in each chapter are unique, a common experimental methodology is used to analyse these algorithms so that the algorithms can be compared. This methodology includes the processing path for reference and unknown video files, the high-performance techniques used to compare feature vectors, as well as, the databases used in this study.

3.1 Video Signature Construction

To match an unknown video clip with a reference video database, each video is first converted into a video signature. The exact format of these signatures is dependent on the specific video match algorithm used. In this study, frames are taken from the reference and unknown videos at a rate of 4 frames per second (4fps) with representative feature vectors computed for each frame. These feature vectors can be a set of floating point numbers or a set of bits that represent the visual content of the frames. The matching algorithms compare the feature vectors, rather than the raw video files, to determine whether a match

has been found.

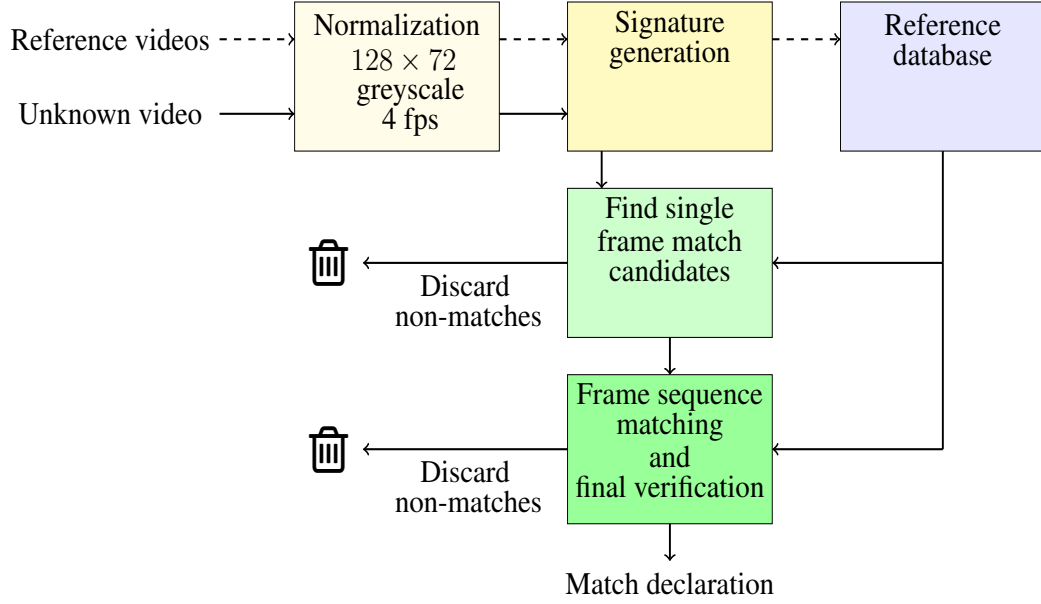


Figure 3.1: Processing path for matching reference and unknown videos

The processing path for matching unknown video clips and reference videos is shown in the Figure 3.1. All videos are first normalised to a frame size of 128×72 pixels, 8 bit greyscale at 4 fps. This normalisation reduces the size and dimensionality of the raw video data making them suitable for video signature generation. The feature vectors are then calculated for each frame and concatenated into video signatures (fingerprints). The reference video signatures are computed off-line (i.e. prior to match time), and are stored in the reference video database. The signature for the unknown video clip is computed just prior to the matching process.

At match time, a two step process is used to identify video matches. In the first step, candidate matches are obtained from a similarity search between the feature vectors of each unknown frame, within the 30 second lookup window, and the database of reference feature vectors. If the distance measure between these individual feature vectors is below a pre-defined threshold, a match candidate is obtained. In the second step, each of the match candidates, from the first step, are used as an alignment point (in time) between the unknown and reference videos. The corresponding reference and unknown feature vectors are compared from the alignment point for the duration of the lookup window. A total distance measure, normally the Euclidean distance, is calculated between the corre-

sponding feature vectors. A match is declared if the total distance between reference and unknown is below a certain threshold.

3.2 Methods Used In Video Matching Algorithms

Video matching algorithms must process large quantities of multi-dimensional data since the reference video database in a commercial setting will have thousands of hours of video. Using the minimum standard for database size in this study of 10,000 hours, as described in the objectives Section 1.4 of this study. The size of the reference database would be 36M seconds of video or 144M frames assuming a frame rate of 4 fps used throughout this study. Using the study objective of 500 bytes per frame for feature vectors, yields a total storage requirement of 72GBytes plus whatever storage overhead is required to structure and search these feature vectors. To achieve high-performance matching with this size of database requires techniques that only examine a small fraction of the total population of feature vectors.

3.2.1 Locality Sensitive Hashing

Locality Sensitive Hashing (LSH) is an approximate nearest neighbour search technique that groups similar multi-dimensional data items together. For matching short duration video clips to a reference database, each video (both reference and unknown) can be regarded as a six-dimensional data item. The six dimensions are the three dimensions of pixel colour (whether coded RGB, YUV or in some other colour model), the two dimensions of pixel location within a frame and the additional dimension of time of the frame within the video. In this study, to reduce data volume and computational load, the three colour dimensions are combined into a single greyscale value reducing the number of dimensions and the scale of the data to process. Each video matching algorithm described in this study uses additional data reduction techniques to further reduce the dimensionality of the data. The algorithms use either spatial or temporal feature vectors at each frame location. The matching process involves comparing these feature vectors within a certain match duration.

In the LSH technique, one or more hashing algorithms are used to map high dimen-

sional data items onto a hash index in N dimensions [59]. If the items are similar, the items are mapped to the same hash index with high probability, if the items are significantly different the items are mapped to widely separated hash indexes also with high probability. Using this approach an initial set of match candidates can be identified for an unknown video clip.

LSH uses hash tables and employs hash functions that put similar items into the same hash bucket. High performance similarity search can be achieved by judicious selection of hashing algorithms, size of hash table and the selection of an appropriate distance measure between the data items. Using LSH can dramatically speed up the matching process in multi-dimensional data sets where the scale, dimensionality and distance calculation would make exact distance matching between an unknown data item and reference data set computationally impractical. LSH finds application in many fields involving high dimensional data including video [60, 61] and image analysis [46, 62].

3.2.2 Feature Vectors and Dimensionality

A feature vector is a vector that contains information describing the important characteristics of an object, each characteristic can be thought of as a element of the vector. In image processing, features can take many forms. A simple feature representation of an image is the raw intensity value of each pixel. However, more complicated feature representations are also possible. The SIFT (Scale Invariant Feature Transform) descriptor features, as described by [24], are often used for matching points of interest in an image. These SIFT features capture the orientation, scale and intensity of a particular location in an image.

For similarity search with low dimensional (2 or 3 dimensions) feature vectors a number of algorithms can be used (k -means, k -d tree, pruning algorithms) to partition data points. However, at higher dimensions these (tree-based) algorithms have poor performance. [63] showed that under the assumptions of data uniformity and independence, all traditional tree-search, clustering and partitioning algorithms were out performed by a simple sequential scan for dimensions greater than 10. This analytical result was corroborated by experimental results using various tree based data structures. The issue of high dimensional search is often called the curse of dimensionality in the literature, where sub-linear search performance (search performance that does not directly scale with dataset

size) cannot be achieved by traditional segmentation-based approaches [64].

3.2.3 Hash tables, Hash Functions and Distance Measure

A hash table is a data structure that implements a list of key-value pairs, a structure that can map keys to values. A hash table uses a hash function to compute an index, also called a hash code, into an array of buckets or slots, from which the desired data item can be found.

Hash functions for LSH can either be data-independent or data-dependent. Data-independent hash functions do not use training data (unsupervised) and rely on the mathematical properties of the data items to construct hash indexes. Consequently data-independent algorithms are robust to data variations as they rely on specific rules without a training process [65]. Data-independent hash functions are created using a randomly generated vector selected from a specific distribution which allows an arbitrary number of hash functions to be generated. Therefore, multiple hash tables can be created to boost the recall rate. However, such methods require a large number of dimensions to reduce the false positive rate. This increases the storage costs and degrades query efficiency.

Data-dependent techniques learn the hashing functions from the training data. As with all supervised methods there is a danger that the resulting hash functions will not generalise well to other datasets. Also, data-dependent techniques do not allow the same richness of hash functions that data-independent methods allow. This richness of hash functions is generated from the data (as it changes) rather than a smaller constrained static training set in a data-dependent technique [66, 67].

Once similar items have been identified through hash table lookup, a distance measure is required to compare the query and hash table entries for similarity (match candidate). The feature vector of the query and match candidates are compared to determine their relative distances. The match candidates can then be sorted into relative distance order to return the closest matches to the query vector.

Generally, the Euclidean distance is chosen as the distance measure for floating point feature values, while the Hamming distance is used for binary bit pattern feature vectors.

More formally, given a set of N items $\mathbf{P} = \{\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_N\}$ from a d -dimensional space \mathbb{R}^d , the nearest neighbour to a query item \mathbf{q} is $N(\mathbf{q}) = \arg \min_{\mathbf{p} \in \mathbf{P}} D(\mathbf{q}, \mathbf{p})$,

where $D(\mathbf{q}, \mathbf{p})$ is the multi-dimensional distance between \mathbf{q} and \mathbf{p} . A typical distance metric is given by the ℓ_s norm, where $s \in (0, 2]$ and can be written as

$$\ell_s = \|\mathbf{q} - \mathbf{p}\|_s = \left(\sum_{i=1}^d |q_i - p_i| \right)^{\frac{1}{s}} \quad (3.1)$$

Typically values for $s = 1$ (the Manhattan distance) or $s = 2$ (the Euclidean distance) are used but other distance measures such as cosine similarity distance, Hamming distance for bit fields and others are also used. As the Euclidean distance uses a distance squared sum it penalises features that are significantly different compared to the Manhattan distance which just sums the absolute differences. The Manhattan distance and cosine similarity may therefore be more appropriate for high-dimensional data where the difference between individual features is less important than the similarity of the overall feature vector. [68].

3.2.4 Locality Sensitive Hashing Based on P-stable Distributions

In this study [69], a hash index is computed from the dot product of the input vector \mathbf{x} and random vector \mathbf{a} drawn from a specific distribution. This dot product $\mathbf{x} \cdot \mathbf{a}$ is a projection of \mathbf{x} onto a one-dimensional line. Since \mathbf{a} is randomly selected from a specific distribution the projection will be randomly distributed onto the line. For locality sensitive hashing the goal is to select a distribution that will project similar input vectors so that they are close together on the line and dis-similar input vectors so that they are far apart.

The one-dimensional line is cut into equally sized segments of size ω . Each line segment represents a hash index that points at a bucket in the hash table. Similar input vectors are therefore placed into the same hash bucket while dis-similar vectors are placed in different buckets. Therefore the hash function becomes:

$$h(\mathbf{x}) = \left\lfloor \frac{\mathbf{x} \cdot \mathbf{a}}{\omega} \right\rfloor \quad (3.2)$$

An arbitrary offset can be added to the hash function to align the resulting hash index to a zero base. Multiple hash functions of this type can be used to increase the probability of finding near neighbours to the input vector \mathbf{x} [69] [70]. Each hash function provides

a hash index that points to a bucket in a hash table. Each bucket has a list of candidate matches and each candidate match from all the buckets is tested for proximity to the original input vector \mathbf{x} .

The Figure 3.2 shows the use of two hash functions that point at buckets which contain the candidate matches $\mathbf{z}_2, \mathbf{z}_4, \mathbf{z}_8$. These candidate matches would be compared with the input vector \mathbf{x} using the distance function D to determine which if any was a match.

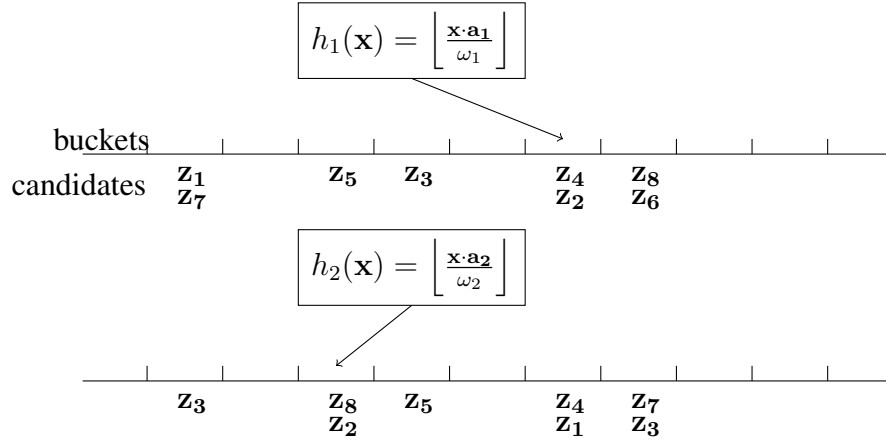


Figure 3.2: P-Stable distribution projections

3.2.5 P-Stable Distribution

The random vector \mathbf{a} that creates the dot product comes from a P-Stable distribution. The two well known examples of P-Stable distributions are:

$$\begin{aligned} \text{Cauchy distribution } f_{\text{Cauchy}}(x) &= \left(\frac{1}{\pi} \right) \left(\frac{1}{1 + x^2} \right) \\ \text{Standard normal distribution } f_{\text{norm}}(x) &= \frac{1}{\sqrt{2\pi}} \exp \left(-\frac{x^2}{2} \right) \end{aligned} \quad (3.3)$$

A distribution is called P-stable if for any real numbers a_1, a_2, \dots, a_n and independently identically distributed variables X_1, X_2, \dots, X_n with the P-stable distribution, the random variable $\sum_i a_i X_i$ has the same distribution as the random variable $(\sum_i |a_i|^p)^{\frac{1}{p}} \mathbf{X}$. Since this $(\sum_i |a_i|^p)^{\frac{1}{p}} \mathbf{X}$ random variable has the same form as the distance measure for

the similarity of two vectors, we can conclude that when we multiple the input vector by a randomly selected vector from the P-stable distribution the resultant scalar product will map similar vectors close together and dis-similar vectors further apart, as required by LSH.

3.2.6 Probability of Hash Table Collisions

Here, we consider the probability that similar items are hashed into the same bucket. The probability that two data points \mathbf{x}_1 and \mathbf{x}_2 are mapped into the same bucket is that the hash indexes computed from the hash functions are the same, that is:

$$\mathbb{P}_{\mathcal{H}} \{h(\mathbf{x}_1) = h(\mathbf{x}_2)\} = \mathbb{P}_{\mathcal{H}} \left\{ \left\lfloor \frac{\mathbf{x}_1 \cdot \mathbf{a}}{\omega} \right\rfloor = \left\lfloor \frac{\mathbf{x}_2 \cdot \mathbf{a}}{\omega} \right\rfloor \right\} \quad (3.4)$$

Assuming the Manhattan distance, a collision can therefore occur when:

1. The distance between \mathbf{x}_1 and \mathbf{x}_2 is smaller than the bucket size ω . Using the Manhattan distance this condition becomes $(|\mathbf{x}_1 \cdot \mathbf{a} - \mathbf{x}_2 \cdot \mathbf{a}| < \omega)$.
2. The points do not get inserted into adjacent buckets due to integer rounding of the hash index.

Due to P-stability, the condition $(|\mathbf{x}_1 \cdot \mathbf{a} - \mathbf{x}_2 \cdot \mathbf{a}| < \omega)$ can be written,

$$|(\mathbf{x}_1 - \mathbf{x}_2) \cdot \mathbf{a}| < \omega \iff |c\mathbf{X}| < \omega \quad (3.5)$$

Where \mathbf{X} has the same p-stable distribution as \mathbf{a} . Assuming c is positive this becomes,

$$\mathbf{X} < \frac{\omega}{c} \implies p(c) = \int_0^{\frac{\omega}{c}} f(u) du \quad (3.6)$$

where \mathbf{X} has the same distribution as the chosen P-stable distribution.

The probability that the boundary lies between \mathbf{x}_1 and \mathbf{x}_2 is

$$\frac{|(\mathbf{x}_1 - \mathbf{x}_2) \cdot \mathbf{a}|}{\omega} = \frac{|c\mathbf{X}|}{\omega} \quad (3.7)$$

and the probability that the boundary does not lie between \mathbf{x}_1 and \mathbf{x}_2 is:

$$1 - \frac{|c\mathbf{X}|}{\omega} \implies p(c) = \int_0^{\frac{\omega}{c}} \left(1 - \frac{|c\mathbf{X}|}{\omega}\right) du \quad (3.8)$$

Assuming these probabilities are independent, we can multiply to obtain the probability of a collision is:

$$p(c) = \int_0^{\frac{\omega}{c}} f(u) \left(1 - \frac{|c\mathbf{X}|}{\omega}\right) du \quad (3.9)$$

If we now substitute the equation with $t = c \cdot u$ we obtain,

$$p(c) = \int_0^{\omega} \frac{1}{c} f_p\left(\frac{t}{c}\right) \left(1 - \frac{t}{\omega}\right) dt \quad (3.10)$$

which is the equation quoted in the literature [68] [69, 71] for the probability of a collision in a hash table. This equation shows that the probability of collision decreases monotonically with $c = \|\mathbf{x}_1 - \mathbf{x}_2\|_p$.

3.3 Databases Used in This Study

The primary objective of this study is to detect video clips (a contiguous set of frames, or, as described in the literature - partial copies) extracted from a reference video at a scale and quality that meets the demands of a commercial video matching service. The video clips are assumed to be distorted in a limited way so that the resultant video clips are perceived to be reasonable quality versions of the original reference video from which

they were derived. The acceptable distortions by this definition include addition of some noise, colour model changes, contrast changes, cropping of frames, frame rate changes, horizontal flip and aspect ratio changes.

3.3.1 Primary Database

The primary source material for this project is the 50,000+ hours of video collected from Audible Magic content providers specifically contributed to Audible Magic for copy detection purposes. This video is provided as a raw binary file containing video frames of 8-bit greyscale, with a frame size of 128×72 , at 4 frames per second. This downgraded form of the reference videos is provided in this way to reduce the size of the videos delivered to Audible Magic and also to limit the security risk associated with holding original, high definition reference copies. This content includes television shows (episodes of Big Brother, The Simpsons, late night U.S. chat shows and a variety of other material), around 150 American feature films and some sporting events.

To match against this reference database, a set of film clips from the full feature films contained within the reference database were identified on various Internet sites. These film clips had various distortions (compared to the reference material) including cropping, static overlays, sub-titles, teasers, trailers and added frames. Approximately 200 of these films clips were initially identified. These 200 films clips will provide the unknowns that should match the reference database and will provide the true positive evaluation. In addition, a large random collection of recent videos distorted in a variety of ways will be used for the false positive evaluation.

3.3.2 Synthetic Databases

According to Van-Hao Le et al. [43] most publicly available datasets for partial video copy detection have a small size, have an unbalanced distribution of positive and negative videos and many have synthetic distortions rather than distortions resulting from real-world noise and transformations. Some authors [3, 43] have criticised the results from these synthetic databases, where near perfect results have been obtained for video matching applications, as not being representative of actual distortions found in the wild. In

particular the TRECVID [45] competition has been criticised in these studies for the use of synthetic content.

However, finding, creating and labelling large video databases with real-world distortions is both time consuming and expensive. Therefore, a number of video datasets, with synthetically created distortions, were created for this project. These datasets cover specific areas of interest for which content copy protection is required by the customers of Audible Magic. These areas of interest include sports, feature films and pop-videos. To create the video datasets, individual videos were generally downloaded from Youtube using the 4K Video Downloader by [72]. The standard format for downloads from this tool (Normal Quality) is MPEG4 with 240p using h.264 encoding. Other videos were supplied by content owners or obtained from others sources at much higher pixel densities and frame rates.

To reduce the dimensionality, complexity and format differences between the videos an initial pre-processing step was used to create a uniform or normalised set of videos. This pre-processing step reduces the frame size, frame rate and simplifies the colour model of each video for both reference and unknown. While these changes remove a huge quantity of information the resultant normalised videos can still be recognised and matched by eye.

To create distorted unknowns from the reference videos each video was cut into consecutive 5 minute video clips. Each 5 minute video clip was then distorted in all the ways described in the Table 3.1 to create the set of distorted unknown videos that will be matched against the reference database.

The standard video distortions used in this project are described in Table 3.1.

Distortion	Description
Color change 1	Changes the brightness and colour saturation of each frame pixel
Color change 2	Increases the contrast between light and dark pixels
Even cropping 1	Evenly crops one eighth of the frame width from both sides of the frame
Even cropping 2	Evenly crops one eighth of the frame height from top and bottom of the frame
Even cropping 3	Evenly crops one eighth of the frame width and height
Frame rate reduction 1	Reduces the frame rate to 15 frames per second before pre-processing to 4 frames per second
Frame rate reduction 2	Reduces the frame rate to 10 frames per second before pre-processing to 4 frames per second
Add news ticker	Add news ticker as overlay onto frame
Add static watermark	Adds static image to all video frames
Change aspect ratio	Changes dimensions of video frame so that the width and height of the video frames are reversed which changes the video aspect ratio
Flip video horizontal	All pixels in the video are flipped across the central vertical line
Rotate 1	Rotate all frames 10 degrees clockwise
Rotate 3	Rotate all frames 10 degrees anti-clockwise

Table 3.1: Standard video distortions

The complex video distortions used in this project are described in Table 3.2.

Distortion	Description
Uneven cropping 1	Crops one eighth of the width from the left hand side of the frame
Uneven cropping 2	Crops one eighth of the width from the right hand side of the frame
Uneven cropping 3	Crops one eighth of the height from the top of the frame
Uneven cropping 4	Crops one eighth of the height from the bottom of the frame
Uneven cropping 5	Crops one eighth of the width from the left hand side and one eighth height from the top of the frame
Uneven cropping 6	Crops one eighth of the width from the right hand side and one eighth height from the bottom of the frame
Flip video vertical	All pixels in the video are flipped across the central horizontal line
Rotate 2	Rotate all frames 20 degrees clockwise
Rotate 4	Rotate all frames 20 degrees anti-clockwise
RHS slant perspective	Simulates looking at the frame at an angle from the right hand side.
LHS slant perspective	Simulates looking at the frame at an angle from the left hand side.
PIP half size	Overlay the center of each frame with a half size version of the original frame
PIP third size	Overlay the center of each frame with a third size version of the original frame
PIP quarter size	Overlay the center of each frame with a quarter size version of the original frame

Table 3.2: Complex video distortions

While the distortions described in Table 3.1 and Table 3.2 are not an exhaustive list of video distortions, the tables contain typical distortions observed in copyrighted video content available on various video sharing web sites.

3.4 Summary

This chapter described the experimental methodology and the video databases used in this study. The subsequent chapters use this methodology and databases to analyse individual video matching algorithms. While the algorithms used for video matching span the range from simple non-machine learning to convolutional neural networks, the data preparation, vector comparison and match statistics techniques are common, allowing a fair comparison to be made between the algorithms.

As with the matching algorithms, the datasets used in this study are common. Unfortunately, very few large publicly available datasets for partial video copy are available. This makes comparing the quality of the matching algorithms presented in this study with those described in the literature difficult. Therefore, the primary method of assessing the quality of the matching algorithms in this study will be to compare the algorithms with each other. Once a high-performing algorithm has been identified, an attempt will be made to compare it with other algorithms described in the literature.

CHAPTER 4

The GRID algorithm

4.1 Introduction

The GRID video matching algorithm was invented by the author and is designed to match unknown video clips to a reference video where the unknown has limited spatial distortion. The algorithm works by averaging the pixel intensity over a simple grid for each frame. The algorithm should be robust to non-spatial distortions, such as frame-level noise, changes in the colour model and contrast. The algorithm is simple to implement and should have a high match transaction rate. The major anticipated weakness of the algorithm is poor match performance on cropped and/or rotated video clips. The performance of the simple version of this algorithm along with methods to overcome these weaknesses are explored in this chapter.

Algorithms that use ordinal measures of regional illumination have appeared in previous studies [73] but no papers were identified that use average pixel intensity. This is probably due to the current focus on deep-learning methods, the inherent weaknesses in the algorithm and the lack of focus on computational performance in most scientific studies.

The term reference video (or just reference) refers to the original or definitive video

that is assumed to contain all the undistorted video content (e.g. a full feature film). The term unknown video clip (or just unknown) refers to the video clip that we are attempting to match.

4.2 GRID Video Signature

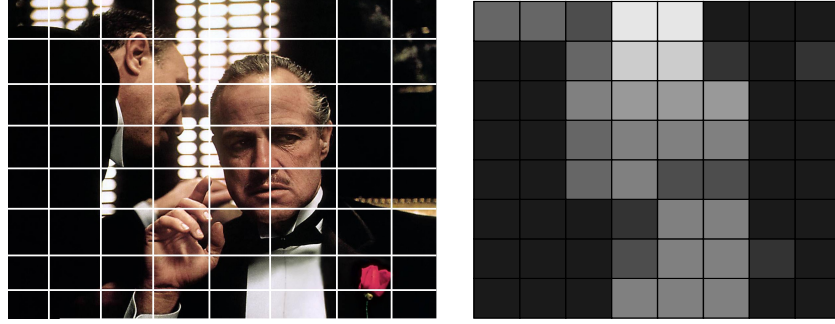


Figure 4.1: Computation of GRID feature vector

The GRID feature vector is computed by applying a simple grid over each frame and averaging the pixel intensity (box filter) for each cell in the grid (see Figure 4.1). These averaged pixel intensities (brightness) are concatenated together to form a feature vector for the frame. The grid dimensions are small integers with 5×5 being typical, resulting in a feature vector of 25 values for each frame. The feature vectors are generally standardised (zero centered and normalised to unit variance) to minimise changes in colour/contrast effects, the resulting features can be used directly or converted into a difference bitmap. A difference bitmap is constructed by comparing adjacent feature values and assigning a one bit if the next feature value is greater or equal to the preceding value or a zero bit if less. These bits are then concatenated together to form a bitmap (24 bits for a 5×5 frame). The feature vector or difference bitmap is used to match individual frames to obtain match candidates.

4.3 Matching Process

The feature vectors for the reference frames are loaded into an in-memory database and indexed by the FAISS software [52]. The FAISS software provides a high-performance

nearest match interface that can be used to obtain match candidates from the query video. The reference video signatures are computed off-line and inserted into the FAISS index prior to match time.

At match time, a two step process is used to identify video matches. In the first step, candidate matches are obtained from a similarity search process on each unknown frame within the lookup window; in the second step, the match candidates are used as an alignment point (in time) between the unknown and reference videos and a Euclidean distance of the GRID vector is calculated between the corresponding sequence of unknown and reference frames. A match is declared if the total Euclidean distance between reference and unknown is below a certain threshold. This two step process provides good accuracy and performance as the first step quickly finds potential matches at a low computational cost while the second step provides a more rigorous comparison between the relatively small set of candidate matches identified.

4.3.1 GRID Algorithm Parameters

The basic GRID algorithm uses a simple box filter on each frame to produce a 5×5 grid and hence a 25 element feature vector for each frame. The video is sampled at a frame rate of 4 fps and the feature vectors of the reference and unknown are compared over a match duration of 30 seconds (120 frames). These parameters were selected from initial experience of using the algorithm. Experiments will be used to determine the optimal value of these parameters to achieve the best possible match statistics for this algorithm.

Parameters in the GRID algorithm that will be varied in this study include:

1. Score match threshold - the match score obtained in the matching process above which a match is declared.
2. Frame rate for the reference and unknown, range to explore 1 fps to 4 fps.
3. Grid size, chosen to be in the range 4×4 to 6×6 .
4. Match duration, number of seconds of video to use in the match process.
5. Frame sequence match algorithm.

Score Match Threshold

The GRID algorithm uses a match score threshold in the final processing step to determine whether the match candidate is an actual match or not. The match score itself is determined from the inverse of the mean Euclidean distance of the feature vectors from the reference and unknown.

$$D_{ru} = \max \left(\frac{1}{N} \sum_{i=1}^N \left(\sqrt{\sum_{j=1}^n (f_r(i, j) - f_u(i, j))^2} \right), D_{min} \right) \quad (4.1)$$
$$M_s = \frac{1}{D_{ru}}$$

Where D_{ru} is the Euclidean distance between the reference $f_r(i, j)$ and unknown $f_u(i, j)$ feature vectors for N frames, n is the number of elements in the feature vectors, D_{min} is an artificial lower bound of the D_{ru} distance to avoid division by very small values in the match score calculation. M_s is the match score which is capped by the value of D_{ru} .

The match score threshold has to be set where it both maximises the true positive rate and minimises the false positive rate. The match score threshold will be varied to determine the optimal value for match accuracy.

Frame Rate

Both reference and unknown videos are generally provided at 25 to 30 fps. At this frame rate the videos contain significant redundant information, GRID frame feature vectors are therefore only computed at a fixed rate of 4 fps. To accomplish this down sampling, the original video frames are down-sampled in time by averaging pixel values of consecutive frames to obtain an average rate of 4 fps prior to computation of the GRID feature vector. Values between 1 fps and 4 fps will be tested to determine the optimal value for this parameter.

Grid Size

The grid size determines the number of individual cells used to form the feature vector for the GRID algorithm. Since this parameter determines the number of cells in the grid it also determines the number of pixels used to form each vector element. The choice of grid size therefore impacts the robustness of the algorithm to cropping and rotational distortions.

Match Duration

The match duration is the number of seconds over which the reference and unknown video clips are compared. Generally the middle 30 seconds of the unknown video clip is used as the query against the reference video database to perform the match process. A shorter duration is preferred, as shorter video clips can be detected, with the desired and minimum acceptable standard of 30 and 40 seconds respectively as shown in study objectives in Table 1.4.

Frame Sequence Match Algorithm

Once candidate matches have been found, a frame sequence match algorithm is used to determine whether each candidate is a true positive or not. This frame sequence algorithm compares time aligned frames or allows some variation in the temporal alignment between the video frames. This parameter attempts to compensate for the inevitable time misalignment between reference and unknown video frames.

4.3.2 Blank (a.k.a. Black) Frames

Blank frames (frames of a uniform colour often just black) must be ignored for matching purposes as they could appear in any video, will match with other blank frames and are likely to cause false positives. Also, sequences of frames with the same content (i.e. static content) can also cause false matching within a video and are detected and ignored by the algorithm.

The pseudo code for the blank frame detector function is shown in Algorithm 1.

Algorithm 1 Pseudo code for the blank frame detector

```
pixelThreshold = 0.05;                                ▷ Blank pixel threshold
blankFrameList  $\leftarrow$  false(numberOfFrames)          ▷ Set blank frame list to false
currentFrame = 1;

while currentFrame < numberOfFrames do              ▷ Process all frames
    ▷
    ▷ Compute a map (frameWidth * frameHeight) of the pixels that have changed
    ▷ more than the threshold, if any pixels has changed more than the threshold,
    ▷ set a flag in that pixel location in the map
    ▷
    map  $\leftarrow$  zeros(frameWidth, frameHeight)
    SetThresholdMap(map, currentFrame, currentFrame, pixelThreshold)
    ▷
    ▷ Scan map from each border (top, bottom, right, left)
    ▷ to locate first non-zero element in map to determine active area of frame,
    ▷ if either less than (frameHeight/2) rows or
    ▷ less than (frameWidth/2) columns are not blank, then
    ▷ declare a blank frame
    ▷
    findRowBorders(map, &borderTop, &borderBottom)
    findColBorders(map, &borderLHS, &borderRHS)
    if (frameHeight - borderTop - borderBottom) < (frameHeight/2) then
        blankFrameList[currentFrame] = true
    else if (frameWidth - borderLHS - borderRHS) < (frameWidth/2) then
        blankFrameList[currentFrame] = true
    end if
    currentFrame = currentFrame + 1
end while
```

The blank frame detector is applied to both reference and unknown frames. In the matching software, a bitmap is used to record whether a frame is blank and when comparisons are made between videos, blank frames are ignored.

4.3.3 Match Candidate Processing

As described in the matching process for the GRID Algorithm 4.3, the first step in the matching processing is to find match candidates. These match candidates are obtained by finding reference feature vectors that are closest to the unknown feature vectors of the frames over the match period (typically, 30 seconds). The reference feature vectors are stored in a FAISS [52] index which provides typically 4 reference frames that have the lowest Euclidean distance between their feature vector and the feature vector of the unknown frame. For the entire 30 second period of the unknown video clip each frame is used to create a set of 4 candidate matches. Therefore at 4fps a total of 480 match candidates is produced ($4 \frac{\text{frames}}{\text{second}} * 4 \frac{\text{candidates}}{\text{frame}} * 30 \text{seconds} = 480 \text{candidates}$) for each match transaction. Duplicate match candidates, which produce the same alignment location between a specific reference and unknown are discarded.

The second phase of the match process, which is required to remove false positives, uses the match candidates as an alignment point in time between the unknown and reference frame sequences. From this alignment point, feature vectors for the next 30 seconds from the reference and unknown in the match candidate are compared calculating a total Euclidean distance for the 30 second sequence. This total Euclidean distance is compared against an empirically determined threshold fed_{thres} to decide whether the candidate is a match.

Through experimentation it was determined that some variation to the frame matching process within the GRID algorithm may yield better results (i.e. higher true positives and lower false positives). The primary variation is to compare feature vectors from the unknown with feature vectors from the reference a few frames either side of the temporal alignment point. The pseudo code for this algorithm is shown in Algorithm 2

Algorithm 2 Grid Frame Matching Algorithm

$frameDistanceTotal = 0.0$ ▷ Zero total Euclidean distance

$frameIndex = 0$

▷ Find minimum distance for all $numFramesToCompare$ frames

while $frameIndex < numFramesToCompare$ **do**

$frameDistanceMin = \infty$

$frameOffset = -fos$ ▷ Initialise frameOffset

while $frameOffset \leq fos$ **do**

▷

▷ $referenceFrameBeg$ is frame index where a match candidate was located

▷ $unknownFrameBeg$ is the corresponding frame index in the unknown

▷

$referenceFrame = referenceFrameBeg + frameIndex + frameOffset$

$unknownFrame = unknownFrameBeg + frameIndex$

▷

▷ Compute Euclidean distance between reference and unknown feature vectors,

▷ using the $referenceFrame$ and $unknownFrame$ respectively.

▷

$frameDistance = Distance(referenceFrame, unknownFrame)$

if $(frameDistance < frameDistanceMin)$ **then**

$frameDistanceMin = frameDistance$

end if

$frameOffset += 1$

end while

$frameDistanceTotal += frameDistanceMin$

$frameIndex += 1$

end while

$frameDistanceMean = frameDistanceTotal / numFramesToCompare$

if $(frameDistanceAverage < fed_{thres})$ **then**

$match = true$

end if

Within the matching algorithm, the frame offset size f_{os} parameter, controls the number of frames either side of the match alignment point, where a minimum Euclidean distance is sought. This parameter will be varied in the parameter tests to determine its optimal value for match accuracy.

4.4 GRID Algorithm Variants

A limitation of the GRID algorithm is that it uses the frame boundaries to create the grid. Therefore it is anticipated that the match performance will be poor on cropped and rotated video clips. This is because the feature vector for a cropped or rotated frame would be computed from a different set of pixels than the corresponding reference frame. Hence the unknown feature vector will be different from the reference feature vector for the same frame and will not match.

4.4.1 Centre of Illumination

One approach to overcome the limitation that the GRID algorithm is computed from the frame boundaries is to use another spatial alignment point between reference and unknown frames. This section describes the computation of a centre of illumination that could be used to spatially align the reference and unknown. The method chosen here is to compute a centre of illumination (location weighted illumination) for each frame. The centre of illumination is likely to be in a similar visual location in both the reference and a cropped unknown since the brightness of the pixels will follow approximately the same distribution horizontally and vertically even if changed by contrast or by scaling. (Although pixels will be missing from the cropped frame so this assumption maybe incorrect for some distortions).

The centre of illumination is computed for each ordinate separately as follows:

$$\hat{x}_r = \frac{\sum_{x,y} I_r(x,y)x}{\bar{I}_r} \text{ and } \hat{y}_r = \frac{\sum_{x,y} I_r(x,y)y}{\bar{I}_r} \quad (4.2)$$

where \hat{x}_r and \hat{y}_r are the computed centre of illumination and $I_r(x,y)$ is the intensity (brightness) of the pixel at point (x,y) in the reference and \bar{I}_r is the average pixel intensity

over the entire frame.

If we assume a simple distortion model (spatial distortion only) for the unknown frames where each ordinate has been linearly transformed in the distorted frame,

$$\begin{aligned} x &\mapsto ax + b \\ y &\mapsto cy + d \end{aligned} \tag{4.3}$$

where a , b , c and d are linear distortion coefficients. We have two frames (reference and unknown), I_r and I_u with,

$$I_r(x, y) \approx I_u(x', y') = I_u(ax + b, cy + d) \tag{4.4}$$

where I_r and I_u are the intensity of the pixels in the reference and unknown frames respectively, x , y are the coordinates and a and c are the x and y scale factors and b and d are the x and y offset values to represent cropping.

The centre of illumination for the unknown frame $I_u(ax + b, cy + d)$ becomes,

$$\widehat{x}_u = \frac{\sum_{x,y} I_u(x, y)(ax + b)}{\bar{I}_u} \quad \text{and} \quad \widehat{y}_u = \frac{\sum_{x,y} I_u(x, y)(cy + d)}{\bar{I}_u} \tag{4.5}$$

$$\widehat{x}_u = \frac{a \sum_{x,y} I_u(x, y)x}{\bar{I}_u} + b \quad \text{and} \quad \widehat{y}_u = \frac{c \sum_{x,y} I_u(x, y)y}{\bar{I}_u} + d \tag{4.6}$$

Since the scale factors a and c are generally close to 1.0 for cropped frames, the centre of illumination for the unknown in this simple distortion model is just a linear translation of the centre of illumination of the reference frame.

To use this analysis, the modified GRID algorithm uses the computed centre of illumination as the centre of a 5×5 grid to calculate the feature vector. The original frame width and height are used which will generally map the grid to some pixels that are no longer on the frame - these off-frame points are ignored for the purposes of the calculation.

4.4.2 Phase Correlation

Another method to spatially align video frames that are miss-aligned is to use a phase correlation. A phase correlation between two images will estimate the relative transla-

tional offset between them. The phase correlation can be calculated using a 2D Fourier transform of both images and computing the cross-power spectrum [74].

Both the centre of illumination and phase correlation variants of the GRID algorithm will be explored in Section 4.6.7.

4.5 Datasets Used to Test GRID Algorithm Parameters

A number of video datasets have been collected for this project. These datasets cover specific areas of interest for which video matching services are provided by Audible Magic. The areas of interest include sports, feature films, pop-videos and animated content.

Two types of video datasets were used for testing the GRID algorithm. Firstly, a simulated dataset using sports events as the reference videos with simulated distortions of these videos for the unknowns to determine the lower bound of match accuracy. Secondly, a real-world dataset using reference videos from Audible Magic customers with unknowns taken from feature film clips obtained from internet web sites.

4.5.1 Simulated Distortions

As discussed in the literature review in Section 2.1.2, simulated datasets, where the unknowns are directly derived from the reference videos by a simple distortion operation, tend to produce unrealistically good match statistics. However, real-world databases that contain both high-quality definitive reference videos and distorted clips of these references are not available. Simulated unknowns can provide a lower-bound of the quality of matching that might be available from real-world data. If poor match performance is achieved with simulated unknowns, we can be reasonably certain that real-world data will produce even poorer results. Hence, simulated unknowns provide good evidence for situations where a particular match algorithm will be weak. Also, the match results provide an indication of the difficulty in obtaining a low false positive score.

4.5.2 Real-World Distortions

The references for the real-world distortions dataset was taken from video signatures obtained from Audible Magic (AM) content owners. This content includes around 200 feature film titles, 40 animated episodes (The Simpsons) as well as a number of television series and sporting events amounting to a total of around 150,000 items of varying duration from several hours to several seconds. In all cases, a degraded version of the original video (128×72 , greyscale, 4fps) was used as the basis of the reference signature. This is because this degraded version is the only form of the original video that is available for signature generation. The original videos are not delivered to AM.

To facilitate experimentation, a subset of the full AM reference database was randomly selected for analysing the GRID algorithm parameters. This subset has 20,000 video signatures which represents around 8,000 hours of video or one eighth of the total. The use of the subset is justified by the reduced time taken to create and match signatures, as well as, the relative nature of this analysis. The objective is to determine the relative value of the algorithmic parameters rather than the absolute match quality.

Two sets of unknowns were used for these experiments. Firstly, for the true positive test, a set of film clips downloaded from YouTube where the full feature film is known to be contained within the reference dataset. Secondly, for the false positive test, a set of sports events, cut into 5 minute clips, known not to be in the reference dataset. The true positive and false positive test are performed for each set of GRID algorithm parameters which result in ROC plots of false positive against true positive rates.

VCDB Dataset

As described in Section 2.1.1 a number of studies, focused on the evaluation of video copy detection techniques, use the publicly available VCDB database as the basis of the performance of their method [1–3, 42].

While VCDB is publicly available, the database only contains 28 sets of distorted videos and does not contain a definitive reference in these sets. Therefore, this database does not align well with the objectives of this study.

However, since real-world reference databases for video copy detection are very scarce, an attempt is made to use this database as an external validation of the GRID algorithm.

4.6 Results

The results shown in this section explore the simple GRID match algorithm parameters, as well as the centre of illumination and phase correlation matching methods.

4.6.1 Simulated Distortion Test

For this test a reference database of 20,000 video items was used representing around 7,750 hours of content. This reference database was supplemented with 263 original sports clips of 5 minutes duration. The unknowns were created from these reference sports clips by distorting in a variety of different ways. Figure 4.2 shows the true positive and false positive match rates for this test. The first graph shows the standard distortions where the goal is a true positive rate of at least 90% and a false positive rate of not more than 1%. The second graph show complex distortions which it would be desirable to match. The dotted lines in each graph show the minimum acceptable standard for true positive and false positive rates as described in Section 1.4.

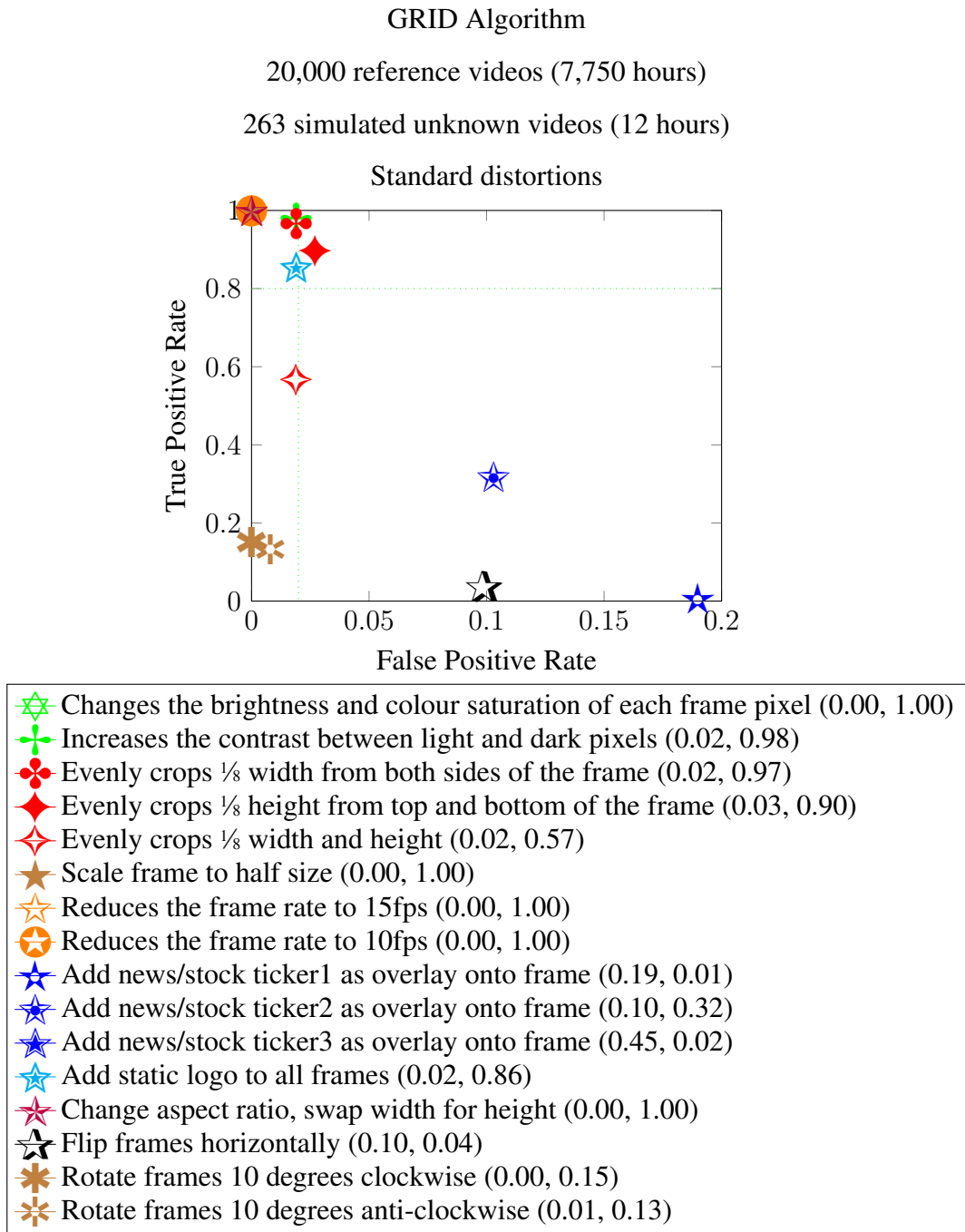


Figure 4.2: GRID algorithm, simulated unknowns, standard distortions match performance

The standard distortions shown in Figure 4.2 shows that distortions due to colour model changes such as brightness and contrast, along with scale changes that retain all the original pixels and frame rate changes, produce a high true positive and low false positive scores as expected. The cropping distortions provide more mixed results, cropping just

the frame width or height has little effect on the match rate while a combined width and height crop dramatically reduces the match rate from 90+% to 57%. News tickers also cause matching issues with a large number of false positives recorded for some types. The horizontal frame flip and rotations significantly effect the match score as expected for this algorithm.

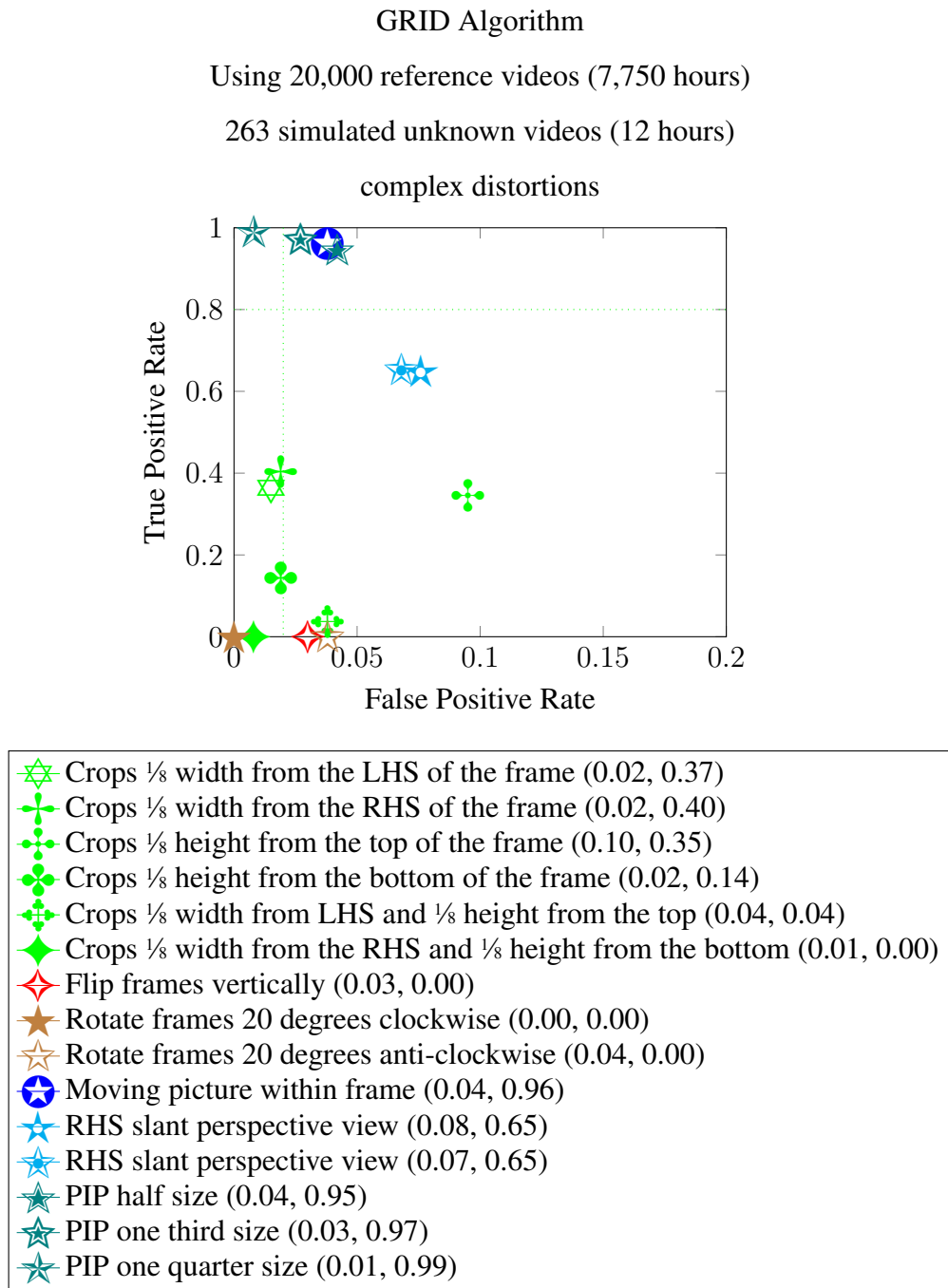


Figure 4.3: GRID algorithm, simulated unknowns, complex distortions match performance

The results for complex distortions are shown in Figure 4.3. This shows that many of the videos distorted by these methods have poor matching characteristics. However, PIP (picture in picture distortions) and moving picture within frame are exceptions. This is probably due to the border detection algorithm that is used to ensure that a static or black border in each frame is ignored prior to creating the feature vector. This approach ensures that only the active area of the frame is used in computing the feature vector. The results also show the relatively poor match performance of the uneven cropping distortions along with the more aggressive frame rotations.

4.6.2 Cropping Test

A further test was performed using the same reference dataset matched against frames that were progressively cropped in the horizontal direction (evenly cropped on both sides of the frame); as this is a common distortion found on various video sharing sites. The results shown in Figure 4.4 show that significant loss of match rate only occurs after one quarter of the frame has been cropped i.e. one eighth from either side of the frame. These results are based on a GRID size of 5×5 and assume even horizontal cropping rather than uneven or vertical cropping.

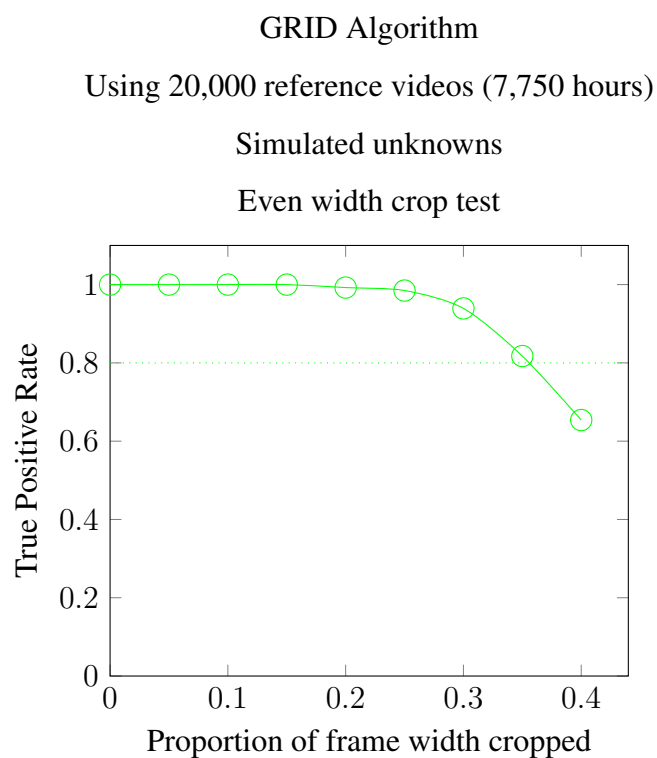


Figure 4.4: GRID algorithm, simulated unknowns, frame cropping evaluation

4.6.3 Score Match Threshold

The GRID algorithm uses a score match threshold in the final processing step to determine whether the match candidate is an actual match or not. This score match threshold has to be set where it both maximises the true positive rate and minimises the false positive rate. In this experiment the score match threshold is varied to determine the best value for the basic GRID algorithm.

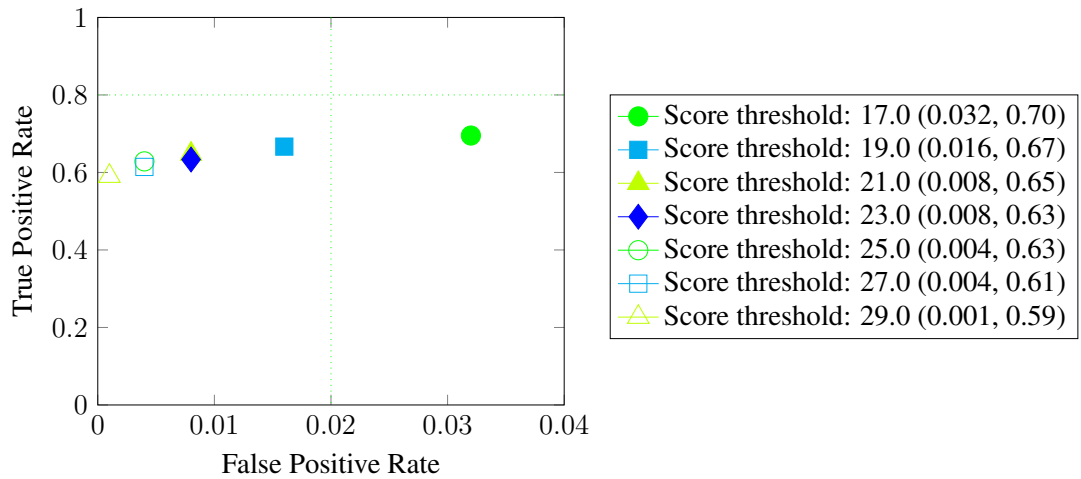


Figure 4.5: GRID algorithm, simulated unknowns, score threshold evaluation

Figure 4.5 shows that a match threshold score of 21.0 produces a true positive score of 65% with a false positive score of 0.8%. While the true positive rate is some way short of the minimum acceptable standard (>80%) for this study as described in Section 1.4, the GRID algorithm is computationally inexpensive, has a high-performance algorithm that is able to identify almost $\frac{2}{3}$ of the true positives with an acceptable false positive rate. Further analysis showed that the candidate finding algorithm (first step in the match process) was responsible for missing approximately 15% of the matches while the remainder of the non-matches were due to the second part of the match process.

4.6.4 Frame Rate

In this experiment the reference and unknown frame rate were varied. The original video frame rate of both the reference and unknown are at least 25 fps. The frame rate used in the GRID algorithm is less than this raw frame rate to reduce the amount of data held and to eliminate data redundancy. However, the reference video signatures received from AM customers have all been down sampled to 4fps so only frame rates less than or equal to 4fps are available. It has been found that the best results are obtained by averaging corresponding pixel intensity values together from consecutive frames to form the frame at the lower frame rate. Each point on the graph represents the results of two experiments - the true positive test and the false positive test.

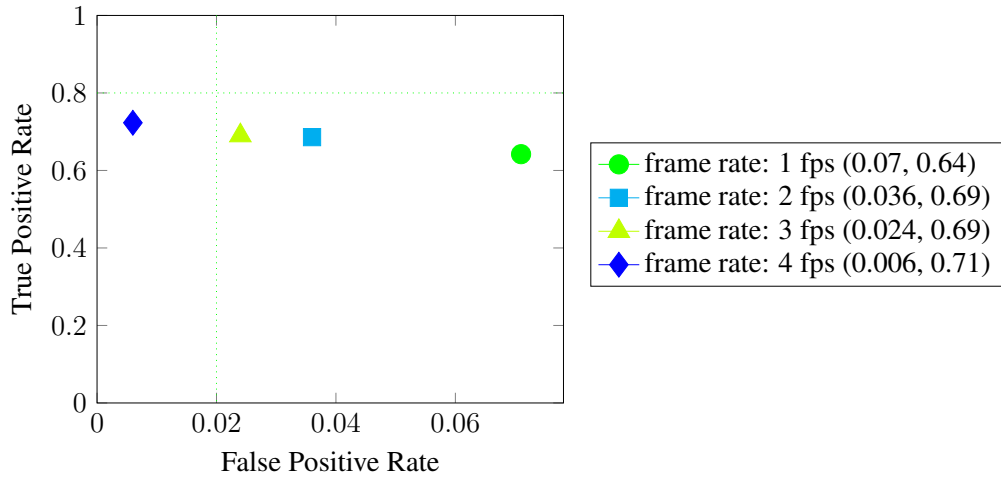


Figure 4.6: GRID algorithm, simulated unknowns, selected frame rates

Figure 4.6 shows that the higher frame rates produce the best match rate. The downside to higher frame rates is more data to analyse and hence poorer computational performance. However, the simplicity of the GRID algorithm means that the performance is acceptable at 4 frames per second.

4.6.5 Grid Size

In this experiment the size of the grid used to calculate the feature vector was varied to find the optimal value. As stated previously, each point on the graph represents the results of two experiments - the true positive test and the false positive test.

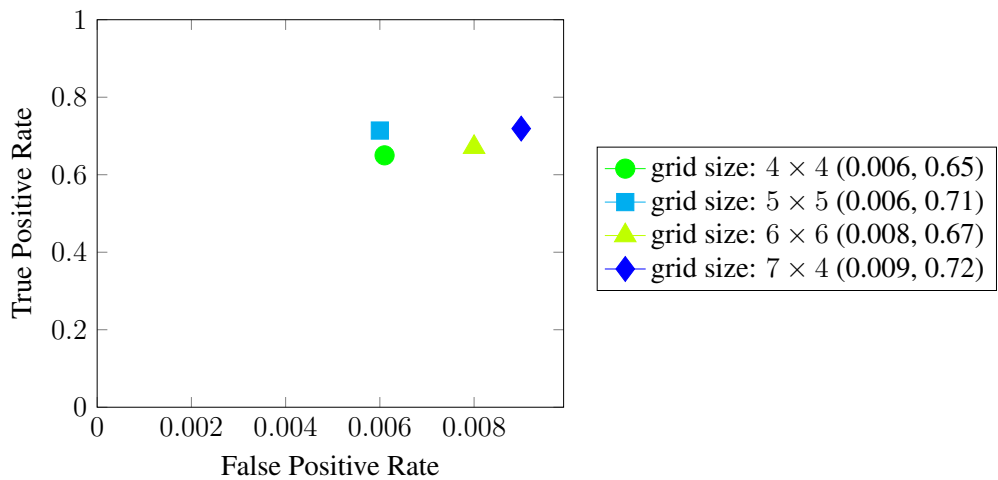


Figure 4.7: GRID algorithm, simulated unknowns, grid size evaluation

Figure 4.7 shows that a grid size of 5×5 produces the best results with the largest true positive and smallest false Positive Rate of the tests. It should be noted that the differences between the results are not that large, so the use of values close to 5×5 produce comparable results. Overall the results for the 5×5 test have a true positive rate of 0.71 and a false positive rate of 0.006.

4.6.6 Match Duration and Frame Sequence Match Algorithm

The match duration is the length of time over which an unknown video is matched against the reference database. The expectation is that a continuous unknown video clip will be matched with the same continuous portion of the reference video, generally the middle of the unknown video clip is used to match the reference over the match duration. The GRID algorithm makes no attempt to match fragmented clips such as feature film trailers or sections of films with added scenes that are shorter than the match duration. While the objective of this study is to have a match duration of 30 seconds or less it is worth considering a longer match duration. Therefore, a test was performed with a match duration of between 20 to 60 seconds. The frame sequence match algorithm provides time displacement jitter between the reference and unknown feature vectors. Feature vectors from the unknown are allowed to match reference feature vectors from surrounding frames to improve the match score.

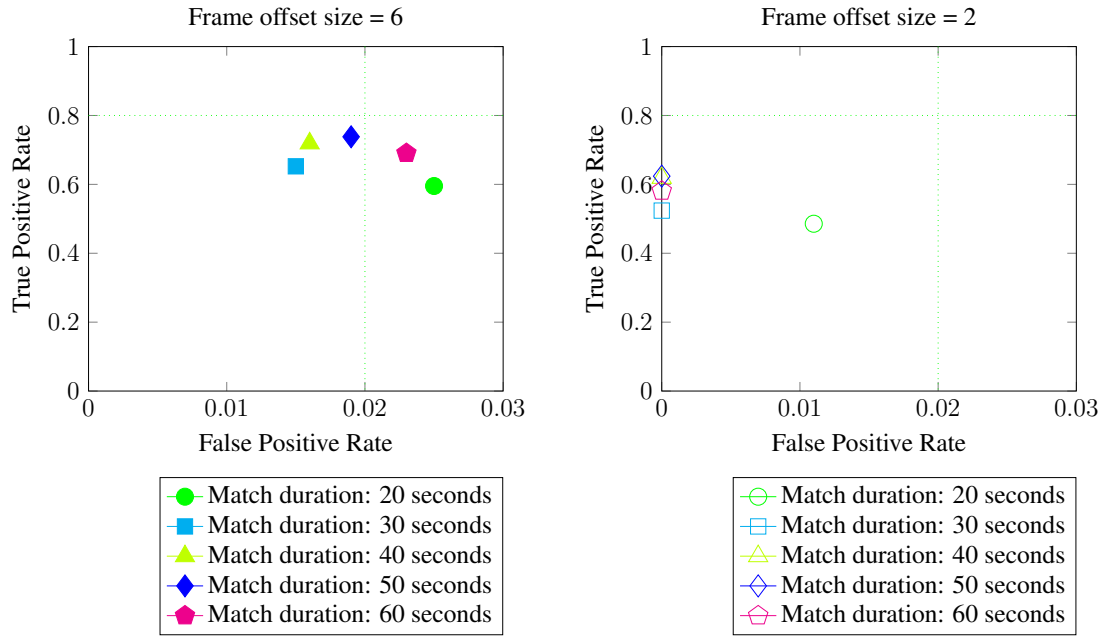


Figure 4.8: GRID algorithm, simulated unknowns, match duration comparison

Figure 4.8 shows two separate sets of experiments to determine the impact of the match duration and frame sequence match algorithm. The left-hand graph with solid shapes in Figure 4.8, used looser parameters in the frame sequence match algorithm (a frame offset size of 6) while the right-hand graph with the open shapes, used tighter parameters (frame offset size of 2). Both sets of results show the highest match rate in the 40 to 50 seconds range. The looser parameters produce a higher match rate along with a higher false positive rate. With the tighter frame sequence match algorithm the false positive score drops to zero - the overall match threshold score could then be raised with the likelihood of a higher match rate at the expense of an increase in the false positive score.

Overall a 40 second match duration would provide around a 10% increase in true positive rate compared with the results for a 30 second match duration on this dataset. This is probably because increased certainty about a match is obtained from a longer video sequence. However, at 60 seconds the true positive rate declines for the frame offset of 6. This declination of true positive rate suggests that for longer match durations at this frames offset size, feature vectors of dissimilar video sequences are being matched.

Further experiments were carried out on a dataset with 20,000 production reference videos that contained 7,750 hours of video. This experiment used various frame offset

size values.

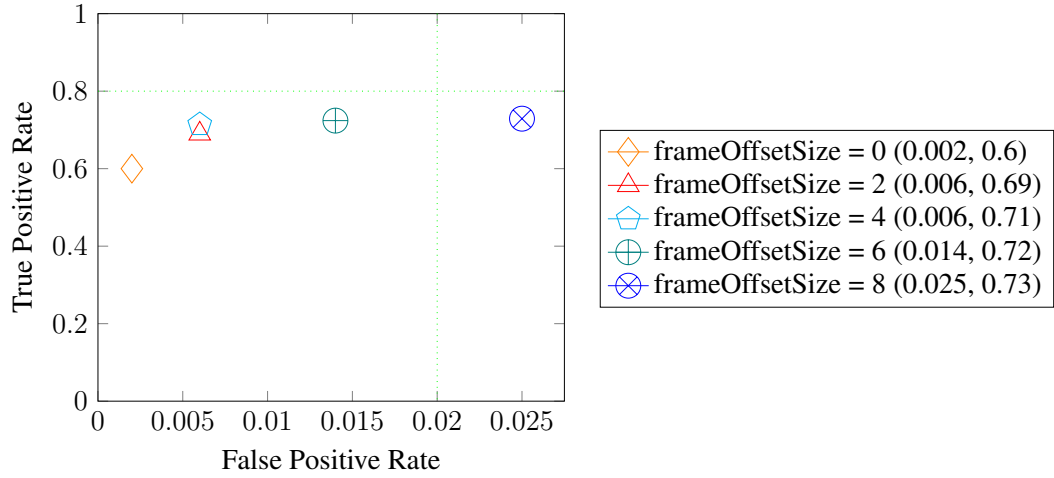


Figure 4.9: GRID algorithm, simulated unknowns, frame sequence match algorithm comparison

Figure 4.9 shows that the highest true positive rate was achieved with a frame offset size of 4 but a lower false positive score is achieved with a frame offset size of 0. A frame offset size of 2 is a compromise value as it balances the higher match rate with a small increase in false positive rate.

4.6.7 Centre of Illumination Test

As described in the GRID algorithm variants Section 4.4 of this chapter, spatially aligning the reference and unknown feature vectors should improve the overall match rate particularly when the unknown video clip has been cropped. To use this approach, the modified GRID algorithm uses the frame-based computed centre of illumination as the centre of a 5×5 grid to calculate the feature vector.

After some visualisation of the results from this algorithm it was noted that the centre of illumination moved jerkily from frame to frame. To smooth this jerky motion a number of experiments were performed using a simple moving average formula that averaged the location of the centre of illumination over a number of consecutive frames.

The results of this experiment are show in Figure 4.10.

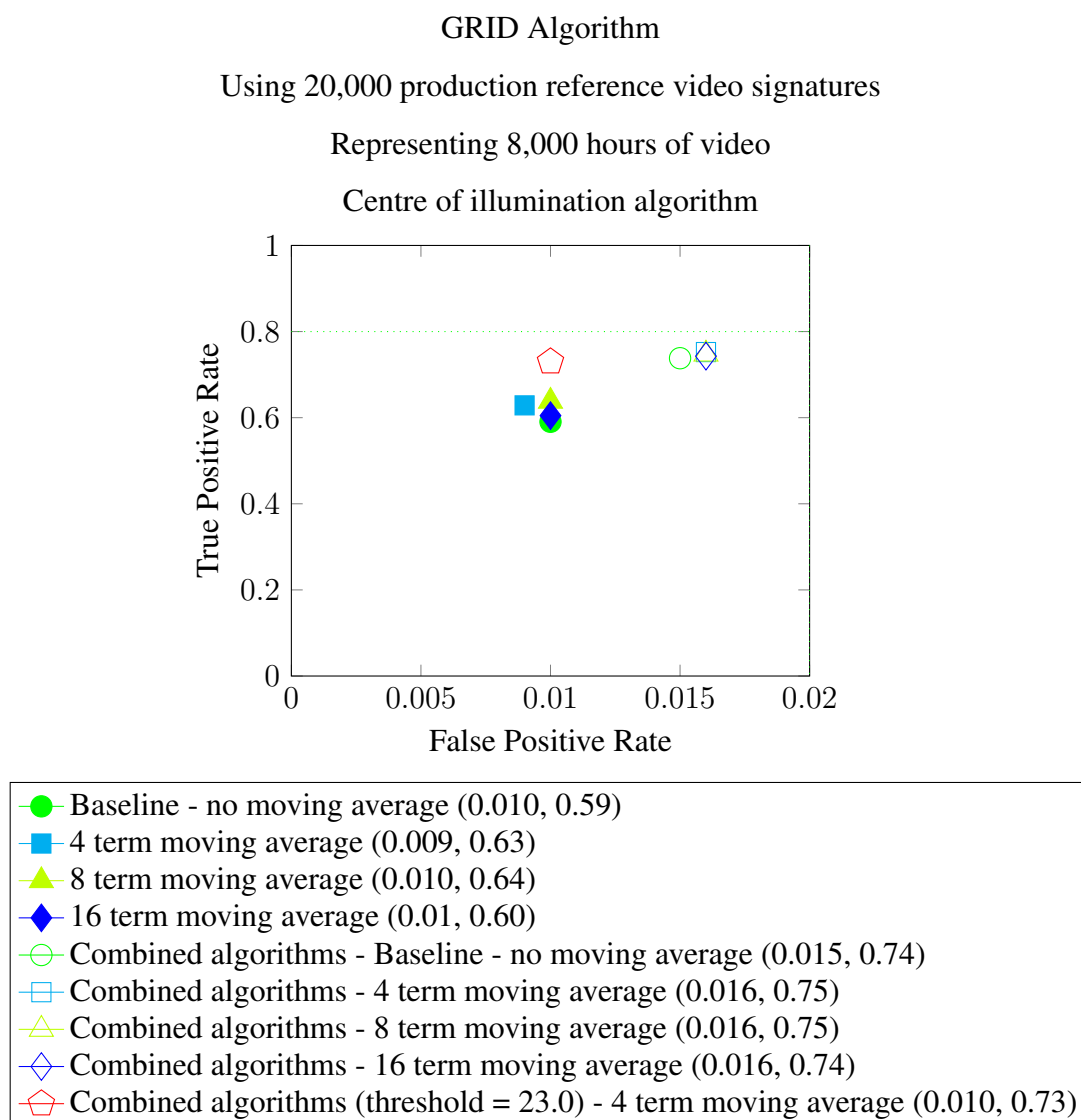


Figure 4.10: GRID centre of illumination algorithm

Figure 4.10 shows that aligning the grid on the centre of illumination (solid shapes) produces similar (but not as good) results as the basic GRID algorithm. Using a moving average to smooth the location of the centre of illumination does not have a significant effect.

Combining the results from this approach with the original GRID algorithm improves the match rate at the expense of a higher false positive rate with the additional processing cost of using two algorithms rather than one. The best match rate is obtained from the combined algorithms with a 4 term moving average. To reduce the false positive rate a final test was run with a higher threshold score of 23.0 (where as the other data points

were obtained with a threshold score of 21.0) this reduced the false positive rate to 1.0% (minimum acceptable level) while only increasing the true positive match rate to 73%. However, this result is only 2% better than not using the centre of illumination algorithm. Therefore, using the Occam's razor principle of less complexity is preferred, using this centre of illumination method in a combined algorithm is probably not worthwhile.

4.6.8 Phase Correlation

To test whether a phase correlation based approach would be useful in this study, an initial test was performed to determine the viability of the method. A random set of matching and non-matching video clips were selected (same number of items in each class). The matching video segments were time-aligned to ensure that matching should take place.

A phase correlation of the frames in each pair of matching and non-matching video clips was performed. The magnitude of the phase correlation was averaged over the length of the video clips (60 seconds). This average phase correlation was plotted against the variance of the translation distance. For a matching pair of videos it would be expected that the translation distance would be reasonably consistent, leading to a small variance in the translation distance over the duration of the video clip. As GPU memory per second of video is a constraint of this study (minimum acceptable standard <1,000 bytes per second) the frame size of the video clips were varied to determine the minimum frame size required to obtain a separation of matching and non-matching video clips. In the results in Figures 4.11 and 4.12 each point represents a single comparison between a pair of matched or non-matched video clips.

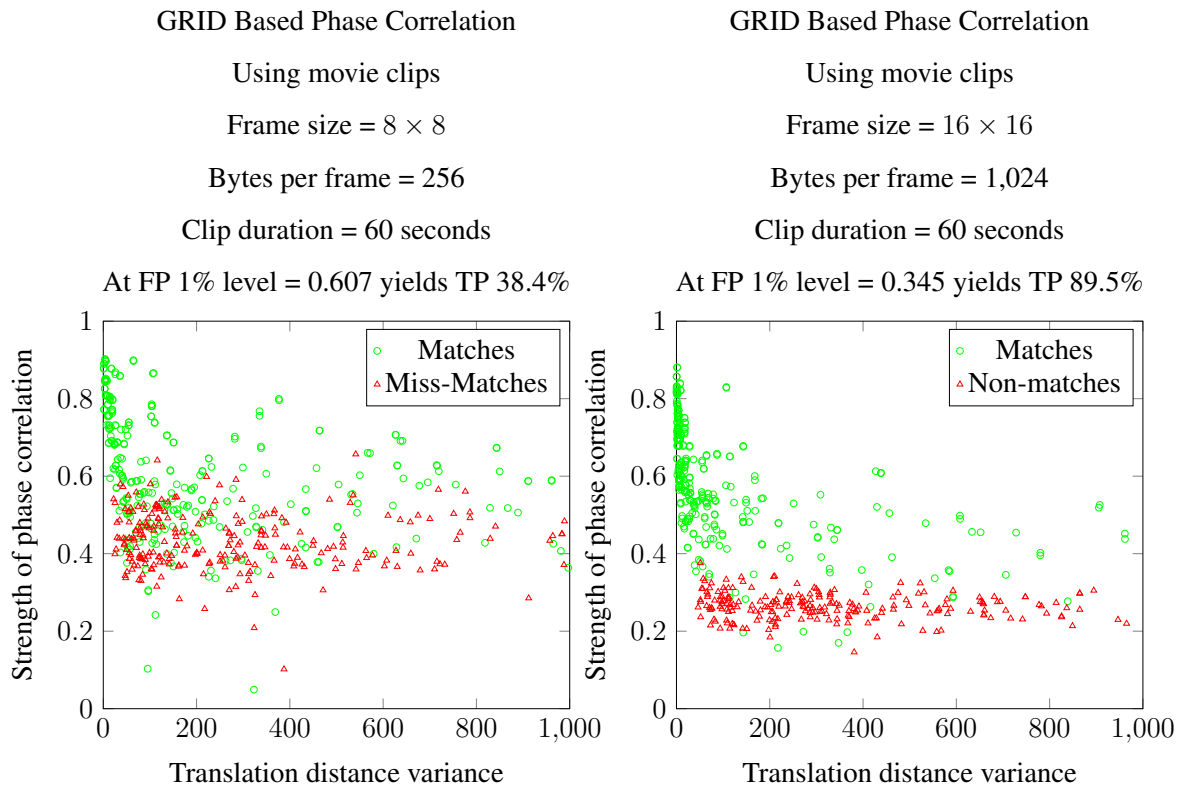


Figure 4.11: GRID algorithm, simulated unknowns, phase correlation match accuracy, 8×8 and 16×16

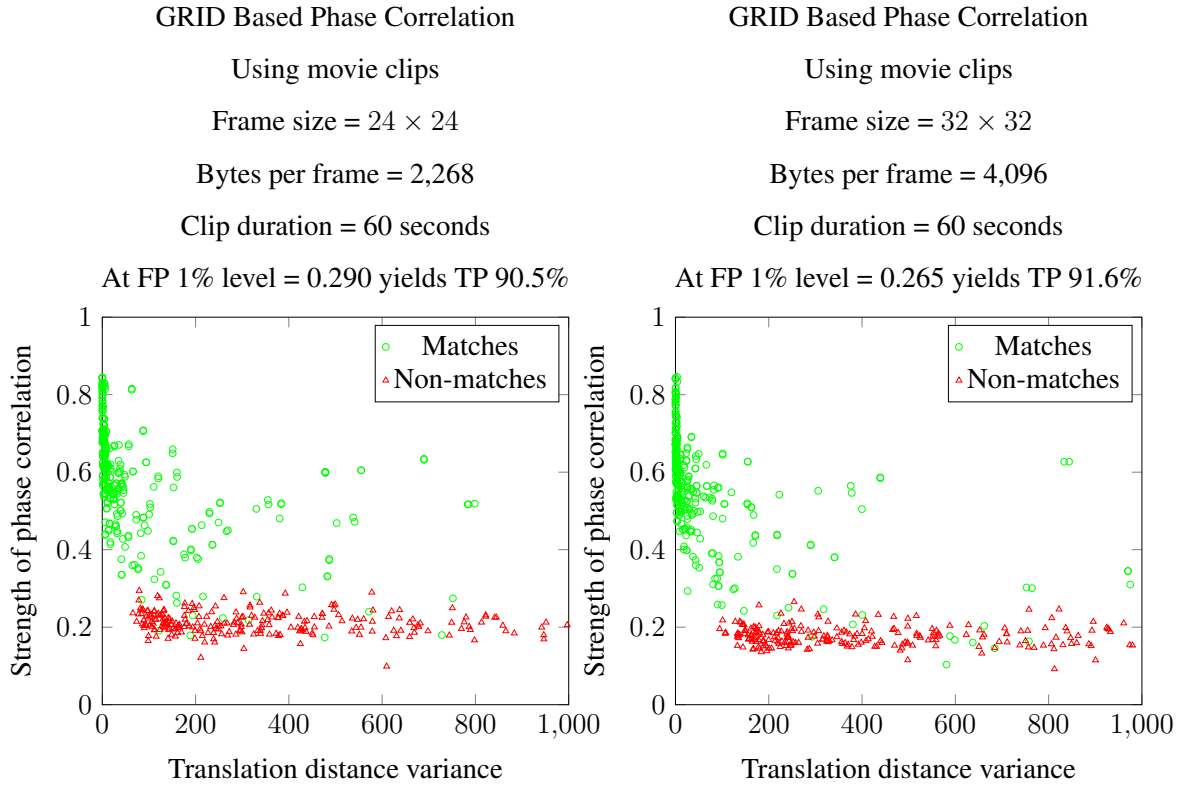


Figure 4.12: GRID algorithm, simulated unknowns, phase correlation match accuracy, 24×24 and 32×32

Figures 4.11 and 4.12 show that as the frame size increases the separation between matches and non-matches increases with the matching having a significantly lower translation distance variance than the non-matches as expected. At a frame size of 16×16 , accepting a false positive rate of 1%, a true positive rate of 89.5% was achieved with this dataset. Unfortunately, this dataset is not typical as the proportion of miss-matches to matches in a real-world dataset is likely to be in the thousands to one rather than in the one to one ratio as in this test set. Therefore a complete separation of matches and non-matches would be required, with this dataset, to make this a practical approach.

Other issues that probably make this approach unviable include the computational cost of the phase correlation algorithm (which is high), the lack of a low computation cost of selecting initial match candidates and the size of the data required per frame.

4.6.9 VCDB Test

In the VCDB test, a dataset was constructed that contained both the VCDB videos along with 20,000 noise videos (7,750 hours of sports) to represent a real-world environment. Unfortunately, some authors [42] use only the VCDB database for their comparison tests which produces unrealistically good results. The unknown videos used for the lookups are the second file listed within the annotation files contained within the vcdb annotation folder. The results for this test are shown in Figure 4.13.

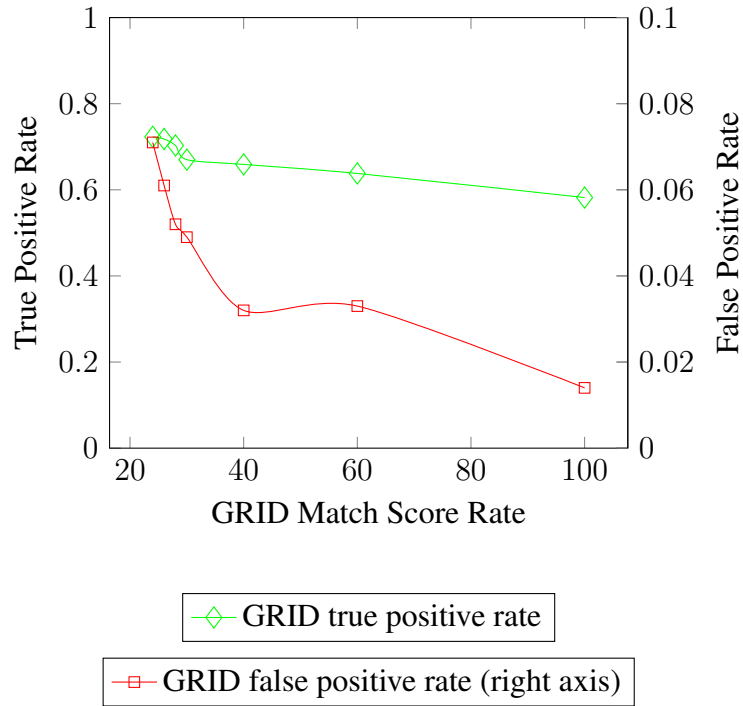


Figure 4.13: GRID match score threshold on VCDB database

In this test the GRID match score threshold was varied between 24.0 and 100.0 to determine the affect of this parameter on the true positive and false positive rate.

The previous results are reported in the study [42] where the best precision of 0.90 was achieved without noise videos. The comparable precision, with noise videos, ($precision = \frac{TP}{TP+FP}$) statistic for this test with a GRID match score threshold of 24.0 is $0.72/(0.72 + 0.071) = 0.91$

4.7 Discussion

The strengths of the GRID algorithm are that it is robust to certain types of noise, it is simple to implement and it provides reasonable match accuracy. The algorithm matches videos where the unknown video is distorted by variations in the colour model (contrast) and pixel noise, as the algorithm normalises and averages the pixel illumination within each frame. The algorithm is relatively simple to implement and has a fast match time, since it only requires a number of single frame lookups to find match candidates in the first stage of the match process. The algorithm also provides reasonable match precision across a variety of content types including sports videos, television shows, animation, etc.

The weaknesses of the GRID algorithm include poor matching with some distortion types, high dependency on the exact sequence of frames in reference and unknown, and failure to match under some types of distortion. The GRID algorithm does not match frames that have been significantly cropped or rotated since different parts of the reference and unknown frames are used to compute the feature vector and hence the feature vectors will not match - a coarse grained grid (i.e. 5×5) helps maintain the matching rate under limited cropping distortions. The algorithm is dependent on the frame sequence of the reference and unknown video clips being time aligned, as the second part of the match processes computes the overall Euclidean distance between the sequence of feature vectors from corresponding frames. If the frames of the unknown and reference are sequenced differently a large Euclidean distance will result which may breach the match score threshold causing a false negative. The algorithm is relatively easy to subvert - by uneven and severe even cropping; an unknown video would not be matched with its corresponding reference.

4.8 Throughput

As described in the objectives of this study in Section 1.7, the throughput objective is that the time taken for an average lookup transaction should be less than a second. This transaction rate should be achieved against a reference database of between 10,000 to 50,000 hours of video. The hardware available for this transaction rate is a standard Linux-based server with 8 GPU cards at a cost of approximately \$25,000 at 2024 prices.

For this throughput analysis the feature film dataset (described here in Appendix D) was bulked out with noise videos to create a reference dataset of 61,000 hours of reference video. The GRID algorithm lookup parameters were set to a score threshold of 21.0, a frame rate of 4fps and a grid size of 5×5 . Two thousand lookups were performed using a 50:50 ratio of items in the reference dataset and items not in the reference dataset. The results from this experiment are shown in Figure 4.14.

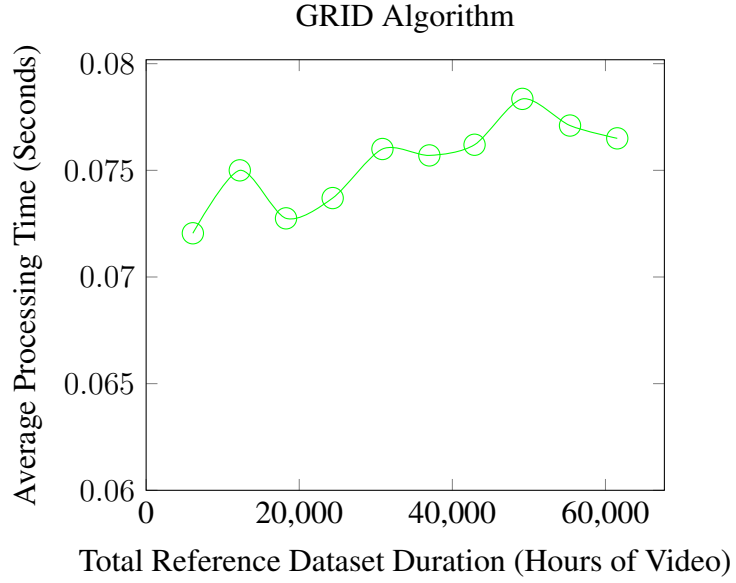


Figure 4.14: GRID algorithm, real-world videos, processing time vs reference database size

Figure 4.14 shows that the average processing time of the 2,000 lookup transactions with the GRID algorithm is a fairly weak function of the total duration of the reference dataset. This is probably because of the efficiency of FAISS software [52] in finding matching vectors. The graph shows that the throughput rate for the GRID algorithm is a little above $\frac{1}{0.08} = 12.5$ or 12.5 transactions per second.

4.9 GPU Memory Requirements

The GRID algorithm requires approximately $5 \times 5 \times 4 = 100$ (GRID frame size * size of 32 bit float) bytes per frame to construct the reference signature. Therefore the GPU memory requirement is approximately 420 bytes per second of video including the indexing and

other overheads. This is within the desired standard of GPU memory per second of video defined in the video matching objectives in Table 1.4.

4.10 Summary

The GRID algorithm has a number of parameters that can be varied to produce a higher match rate with an acceptable false positive score. The table below summarises the results from the tests from the preceding sections.

GRID algorithm parameter	Optimal value
Score threshold	21.0
Frame rate	4 fps
Grid size	5×5
Match duration	40 seconds
Frame sequence match algorithm	Frame Offset Size = 2

Figure 4.15: Grid algorithm parameters table

Using these values a match rate (true positive) of 70% with a false positive rate of 1.2% was achieved with the real-world feature film dataset at a throughput of 21 transactions per second. While this match rate is below the target for this study (90%), the performance of this algorithm allows computational room for other algorithms to be combined with the GRID algorithm to potentially reach the overall target.

CHAPTER 5

Spatial Points of Interest

The weaknesses of the GRID algorithm, described in Chapter 4 are primarily due to the algorithmic assumption that the bounding box of each frame is the same for the reference and unknown. For distortions where this is not the case, such as cropped, particularly unevenly cropped and rotated videos, poor match performance is the result. Instead, this chapter explores a matching technique that does not use this bounding box assumption - spatial points of interest.

Conceptually, points of interest algorithms appear to be well suited to matching images (or video frames) where the unknown has distortions around the edges of an image (such as news and stock tickers, static logos) but is faithfully reproduced near the image centre [75]. However, these methods have been exclusively focused on image rather than video matching in the literature. For images, points (of interest) can be selected such that they are near the centre and are therefore unaffected by distortions that only affect the periphery of the image. Furthermore, a point of interest algorithm, that is robust to image scaling, could potentially match points in cropped images (which are inherently scaled) where other matching algorithms are less successful.

One particular example of a points of interest algorithm is the SIFT (Scale-Invariant Feature Transform) [24]. This appears to be robust to distortions that defeat other match-

ing techniques including scale, rotation and orientation distortions. However, the SIFT algorithm is known to be computationally expensive even for image matching [30] - other variants of the SIFT algorithm (SURF [25], BRIEF [28] and ORB [29]) attempt to address this performance issue but few examples of these algorithms being used for video matching are available in the literature. Therefore, low complexity methods that involve a few points per frame will be the focus of this chapter.

5.1 Signal to Noise Ratio

To compare points of interest algorithms, a method is needed to evaluate the results of experiments that employ different algorithms.

Since point-of-interest algorithms are sensitive to noise (as noise can be mistaken for points of interest), a definition of the signal-to-noise ratio for images and videos is proposed. The assumption is that if the signal-to-noise ratio is too low, these algorithms will not identify sufficient common points between the reference and unknown video clips to match their respective video frames. In this scenario the video clips will not match and the objectives of this study will not be achieved. Signal to noise ratio is a preferred measure for the assessment of a points of interest algorithm as it provides a per pixel assessment of the noise level rather than an overall image quality that would be obtained from SSIM (Structural Similarity Index) or other perceptual quality measures.

The basic definition of signal to noise ratio (SNR) is:

$$SNR = \frac{Power_{signal}}{Power_{noise}} \quad (5.1)$$

In the video datasets used for testing in this project, both the reference and unknown videos are available for analysis. Starting with video frames (images), we can compute both $Power_{signal}$ and $Power_{noise}$ for all frames. Since there is no useful signal in an image which has all the pixels with the same illumination intensity (greyscale value), we can define $Power_{signal} = 0$ for this situation. Therefore we can define $Power_{signal}$ for an image as;

$$Power_{signal} = \frac{1}{Area} \sum_{x_r, y_r} (I_r(x_r, y_r) - \bar{I}_r)^2 \quad (5.2)$$

where $Area$ is the number of pixels in the image, $I_r(x_r, y_r)$ is the illumination intensity of the pixel located at (x_r, y_r) and \bar{I}_r is the mean pixel illumination intensity of the entire image. The subscript I_r is used to identify a reference.

For $Power_{noise}$ since we have the distorted image (assuming the distorted and reference frames in a video are both spatially and temporarily aligned), the power of the noise is:

$$Power_{noise} = \frac{1}{Area} \sum_{x, y} (I_r(x_r, y_r) - I_u(x_u, y_u))^2 \quad (5.3)$$

where $I_u(x_u, y_u)$ is the illumination intensity of the pixel located at (x_u, y_u) in the unknown image (distorted frame). (x_u, y_u) is the coordinate of the unknown that corresponds to the pixel (x_r, y_r) in the reference.

5.2 SNR Analysis on Sports Dataset

The sports dataset used in this study and described in detail in Appendix C consists of a set of reference videos along with 31 different types of simulated distortion.

The foregoing analysis of signal to noise ratio was applied to this dataset. For each distortion type the signal to noise ratio calculation was applied and the values averaged across the different sports represented in the dataset. To compute the signal to noise ratio on the cropped videos, the pixels in the distorted frames were mapped onto the corresponding pixels in the reference frames using bilinear interpolation. Using this methodology the following SNR results were obtained;

Code	Distortion	Average SNR
a1	Color change 1	2.8
a2	Color change 2	1.5
b1	Even cropping 1	1.6
b2	Even cropping 2	1.5
b3	Even cropping 3	1.1
c1	Uneven cropping 1	1.8
c2	Uneven cropping 2	1.7
c3	Uneven cropping 3	2.0
c4	Uneven cropping 4	1.4
c5	Uneven cropping 5	1.2
c6	Uneven cropping 6	1.1
d1	Reduced frame size	8.4
e1	Frame rate reduction 1	4.9
e2	Frame rate reduction 2	5.2
f1	Add news/stock ticker 1	0.33
f2	Add news/stock ticker 2	0.41
f3	Add news/stock ticker 3	0.36
g1	Add static watermark	2.6
m1	PIP half size	1.5
m2	PIP third size	2.6
m3	PIP quarter size	3.5

Table 5.1: Sports dataset, average SNR for simulated distortions

The results in Table 5.1 show that the uncropped distorted videos (codes: a1, a2, d1, e1, e2, g1, m1, m2, m3) have a SNR of between 8.4 and 1.5. Of these uncropped videos, the videos that have not been rescaled produce significantly higher SNR than those videos that have been cropped.

The cropped distorted videos produce SNR values in the range of 2.0 and 1.1. This

shows that rescaling introduces significant local noise for these fairly modest transformations. This noise is probably introduced by the interpolation artefacts inherent in any scaling operation.

The news/stock ticker distortions have the lowest SNR (between 0.41 and 0.33) as these distortions obscure part of the video so a high noise component in these areas is expected.

5.3 Baseline for Spatial Points of Interest Algorithms

To establish a baseline performance for the spatial points of interest algorithms, random frames were selected from the reference videos in the sports dataset. Each of these random frames was treated as a reference image and Gaussian noise of different intensities was added to these references to create the corresponding distorted frames. The points of interest calculation was run on both reference and distorted images with the number of points that matched between the reference and unknown recorded as a proportion of the total. The signal to noise ratio of the reference and unknown was calculated as described in the signal to noise Section 5.1 of this chapter. The following graph displays the results of these tests for the Harris Corner Detector, the Shi-Tomasi Corner Detector [76] and the SIFT point of interest algorithms. Each point has been calculated from one hour of video (3,600 seconds @4ps = 14,400 frames). The number of points of interest was limited to 100 to conform to the memory limitations described in the overall objectives of this study in Table 1.4.

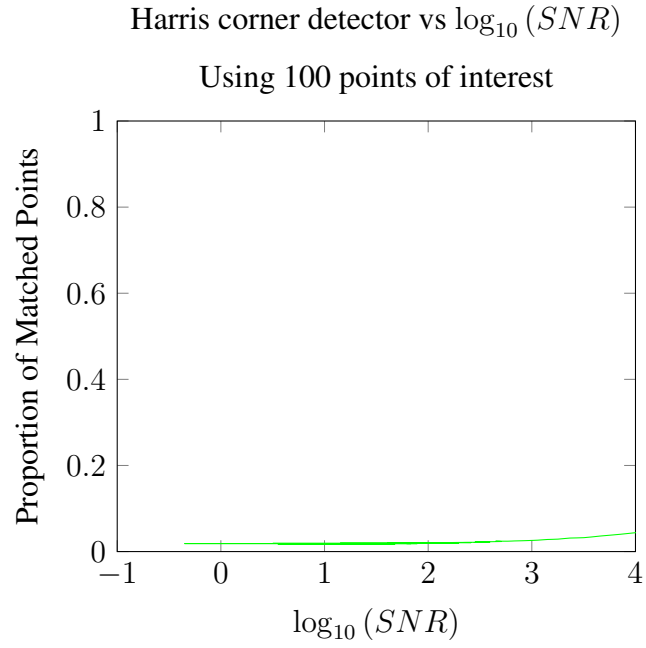


Figure 5.1: Harris corner detector match results against $\log_{10}(SNR)$ on sports database

The Figure 5.1 shows that the Harris corner detection algorithm produces few common matches between the reference and distorted unknown when the algorithm is limited to the top 100 points. The SNR has to be large to find common matched points between the reference and distorted unknown. This is therefore not a suitable detector for the video matching we are attempting in this study.

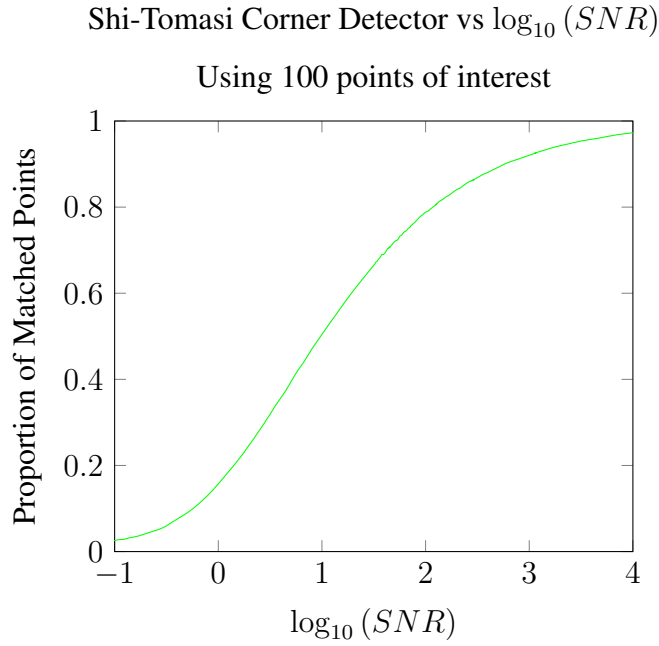


Figure 5.2: Shi-Tomasi points of interest algorithm match results against $\log_{10}(SNR)$ on sports database

The graph in Figure 5.2 shows that the Shi-Tomasi Corner Detector [76] (obtained from the `goodFeaturesToTrack` function in OpenCV) requires a relatively low SNR (compared with the Harris corner detector) to obtain common match points between reference and unknown. However, the analysis of the sports dataset shows that many common simulated distortions reduce the SNR to around 1.0 ($\log_{10}(1.0) = 0.0$). This translates to obtaining 20% of the total match points between reference and unknown with this level of Gaussian noise.

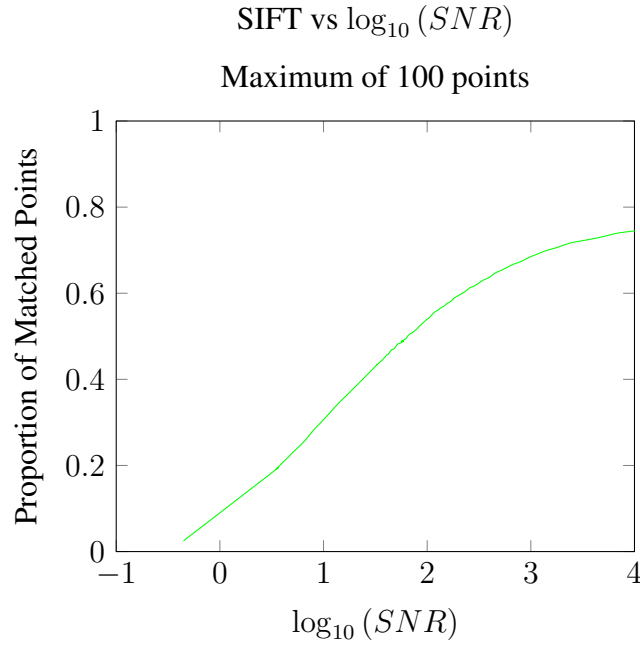


Figure 5.3: SIFT detector match results against $\log_{10}(SNR)$ on sports database

Using the SIFT algorithm [24] shown in Figure 5.3 yields an even lower number of match points than the Shi-Tomasi Corner Detector [76]. Therefore, the Shi-Tomasi Corner Detector [76] is the preferred detector for spatial points of interest matching based on this simple signal to noise ratio analysis.

5.4 Summary

From the experiments undertaken in this chapter the spatial points of interest algorithms are not resilient to small amounts of random noise or rescaling transformations, see Section 5.2. For the relatively simple synthetic distortions found in the sport dataset the SNR is around 1.0. Using the Shi-Tomasi Corner Detector [76] only 20 of the most significant 100 points of interest would correspond between the reference and unknown at this SNR, see Figure 5.2. Using these data points we can compute the probability of finding a match between an unknown frame and any other random reference frame. In the event that this probability is high, too many false positives would be produced to create a successful match algorithm.

Using F_s as the frame size in pixels, the probability of not finding a match with the 20

significant points of interest (spoi) for one random frame is,

$$\begin{aligned} P(\text{no match for 20 spoi in one frame}) &= \frac{\text{non-matching locations}}{\text{total locations}} \\ &= \frac{F_s - 20}{F_s} \end{aligned} \quad (5.4)$$

The probability of not finding a match for all 100 spoi in one random frame is,

$$P(\text{no match for all 100 spoi in one frame}) = \left(\frac{F_s - 20}{F_s} \right)^{100} \quad (5.5)$$

Assuming a video reference database of N frames, the probability of not finding a match for all frames in this database is,

$$P(\text{no match for all } N \text{ frames in database}) = \left(\frac{F_s - 20}{F_s} \right)^{100*N} \quad (5.6)$$

If we use a frame size of $128 * 72 = 9,216$ and a very small database of 1,000 frames, the probability of not finding a match to a random frame is,

$$\begin{aligned} P(\text{no match for all 1,000 frames in database}) &= \left(\frac{9,216 - 20}{9,216} \right)^{100*1,000} \\ &= 4.4 * 10^{-95} \approx 0 \end{aligned} \quad (5.7)$$

therefore, we are virtually certain that an unknown frame will match at least one random frame and produce false positives. This makes these spatial points of interest algorithms unviable for the match objectives for this study.

A study [77] has also shown that matching pairs of images by these methods is computationally expensive. Since the scale of matching required by this study is many orders of magnitude greater than simple pair-wise image matching performed here, these techniques do not provide a promising solution to the objectives of this study.

CHAPTER 6

Temporal Points of Interest - The PIXL Algorithm

This chapter introduces a novel algorithm, designed by the author called PIXL. This algorithm is designed to overcome the weaknesses of the GRID algorithm described in Chapter 4 which are primarily due to the algorithmic assumption that the bounding box of each frame is the same for the reference and unknown. The PIXL algorithm is based on temporal points of interest. This method uses changes of pixel intensity over time to create a unique bit pattern that can be used to match the reference and unknown.

Temporal points of interest algorithms were not found in the literature since most video matching techniques have been based on image matching [47]. Some studies have used temporally-aware indexing of feature vectors [44, 78] but these studies used relatively large spatial feature vectors.

The PIXL algorithm is designed to match distorted video clips with the underlying reference video, where the distortion primarily affects the outer region of the video frame rather than its centre (e.g. cropped frames). Distortions of this type are common in practice and can be caused by various types of cropping, enlargement and also by the use of overlays such as news and stock tickers, as well as, static logos.

The PIXL algorithm uses the variation (over time) of the pixel intensity in the central region of the frame to match video clips. Since the algorithm uses only pixel information

from the central region of the frame, it ignores the outer region, making it unaffected by scrolling tickers, logo overlays (typically found at the edge of the frame), certain types of cropping and other changes to the border. To mitigate the effects of spatial scaling, video conversion artefacts and other noise, a two-dimensional Gaussian filter is applied to the pixel values in the central region (spatially) and a one-dimensional Gaussian filter to the changes of pixel values over time (temporally).

6.1 Computation of the PIXL Video Signature

The reference PIXL signatures are computed off-line from the degraded videos provided by Audible Magic (AM) customers (128×72 , greyscale, 4fps) and stored in the reference signature database. The unknown video clip is converted into a video signature at match-time and matched against the reference video signature database.

For the reference video clips, PIXL bitmaps are computed at 9 specific pixel locations while for the unknown video clips PIXL bitmaps are computed at all pixel locations. This allows any pixel location from the unknown to be compared with the 9 specifically defined locations in the reference while keeping the size of the reference database to a minimum.

The reference signature is computed by comparing the smoothed pixel intensities (brightness) of the central and eight surrounding pixels (at predefined locations) in the current frame with the corresponding pixels in the subsequent frames. Each of the 9 pixel locations are considered separately when computing the pixel intensity changes. At each pixel location, as the smoothed pixel intensity increases or decreases from frame to frame, a bitmap of these changes is built up. A set bit is used to indicate an increase in intensity from frame to frame while a clear bit indicates a reduction in pixel intensity value. These bits are concatenated together to form a bitmap for each of the nine selected frame locations in the reference video signature.

Table 6.1 illustrates the creation of only the central pixel bitmap. This frame sequence is derived from the greyscale intensity value (pixel values: 0.0 to 1.0) at 4 frames per second.






frame sequence number	reference frame	intensity of central pixel	intensity difference	hash bit	hash value
1		0.47	-	-	-
2		0.64	+0.17	1 ₂	1 ₂
3		0.55	-0.9	0 ₂	10 ₂
4		0.76	+0.21	1 ₂	101 ₂
5		0.38	-0.38	0 ₂	1010 ₂
...

Table 6.1: Computation of PIXL bitmap

The process described in Table 6.1 is used to create all nine PIXL bitmaps for the reference video signature. The nine locations selected for this purpose are the central pixel and eight surrounding pixels which are one eighth of the frame width and one eighth of the frame height distance from the central pixel. As illustrated in Figure 6.1.

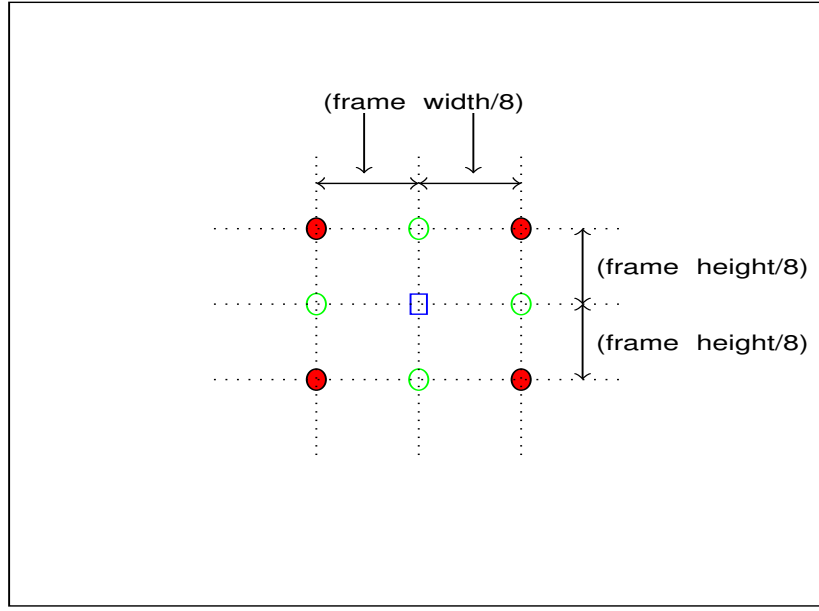


Figure 6.1: Pixel frame locations for reference PIXL signature

Figure 6.1 shows the nine pixel locations used to compute the PIXL bitmaps for each reference signature, with the open blue central square representing the central pixel and the solid red and open green circles showing the locations of the other pixels. The video signatures that include the PIXL bitmap for each of the nine locations is computed off-line and loaded into GPU memory prior to commencement of matching operations.

The same algorithm, to compute the PIXL bitmap, is performed on the unknown at every $(128 \times 72 = 9,216)$ pixel location so that the reference PIXL bitmap from the central location can be compared with any pixel location of the unknown.

6.2 Matching Process

The PIXL bitmaps for all the reference videos are computed and loaded into GPU memory prior to the start of the matching process. A reference hash table is built from the central PIXL bitmaps of each video, each hash code (hash table index) is composed of the n (typically 18) consecutive set of PIXL bits that starts at each of the reference video frames. The hash table uses a fixed-length hash code of n bits long. Each hash table index has a list of all the reference frames (identifier and frame number) that corresponds to the hash code (hash index). Each entry in this list contains a video reference identifier and frame

number. The hash table construction process is shown in Figure 6.2.

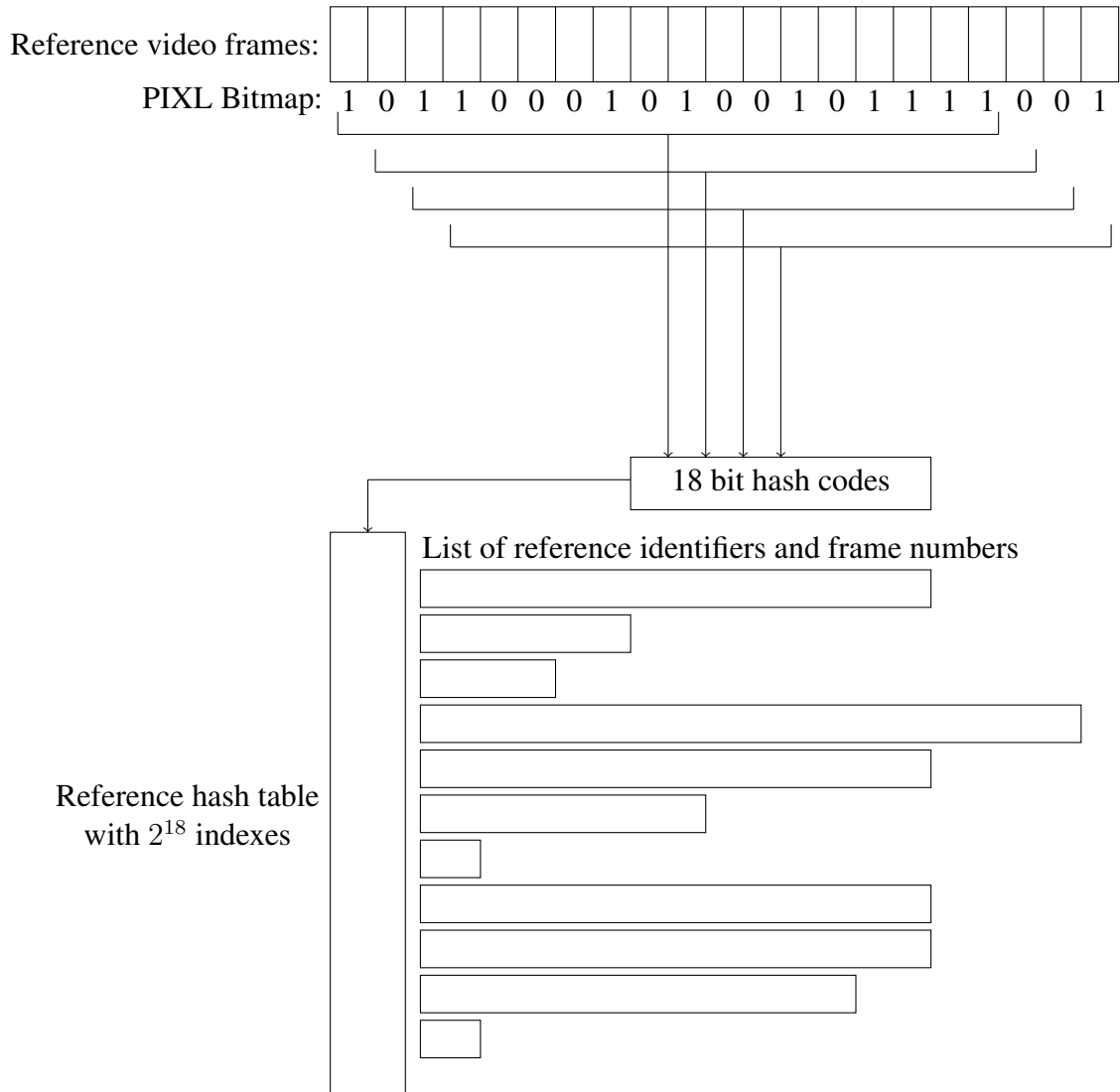


Figure 6.2: PIXL reference hash table construction diagram

At match time, a two step process is used to identify video matches. Firstly, PIXL bitmaps are computed for all the pixel locations in the unknown video over the match duration. Initial match candidates are obtained by performing hash table lookups of these computed unknown PIXL bitmaps over a defined match duration (typically 30 seconds). The number of these initial lookups is reduced (for performance reasons) by limiting the area of the unknown frame over which the lookup is performed - an area around the central pixel is chosen. This does not affect the match rate very much since the corresponding location of the reference central pixel in the distorted video is likely to be close to the

centre of the frame even when the distorted video has been cropped.

The second step of the lookup process is to eliminate the false positives from the match candidates. This is achieved by creating similar hash codes (set of consecutive PIXL bits) using the 8 locations surrounding the central pixel in the reference. As illustrated in the Diagram 6.3. The yellow circle represents the central pixel, the green circles represent the North, South, East and West locations, and the red circles represent the North East, North West, South East and South West locations.

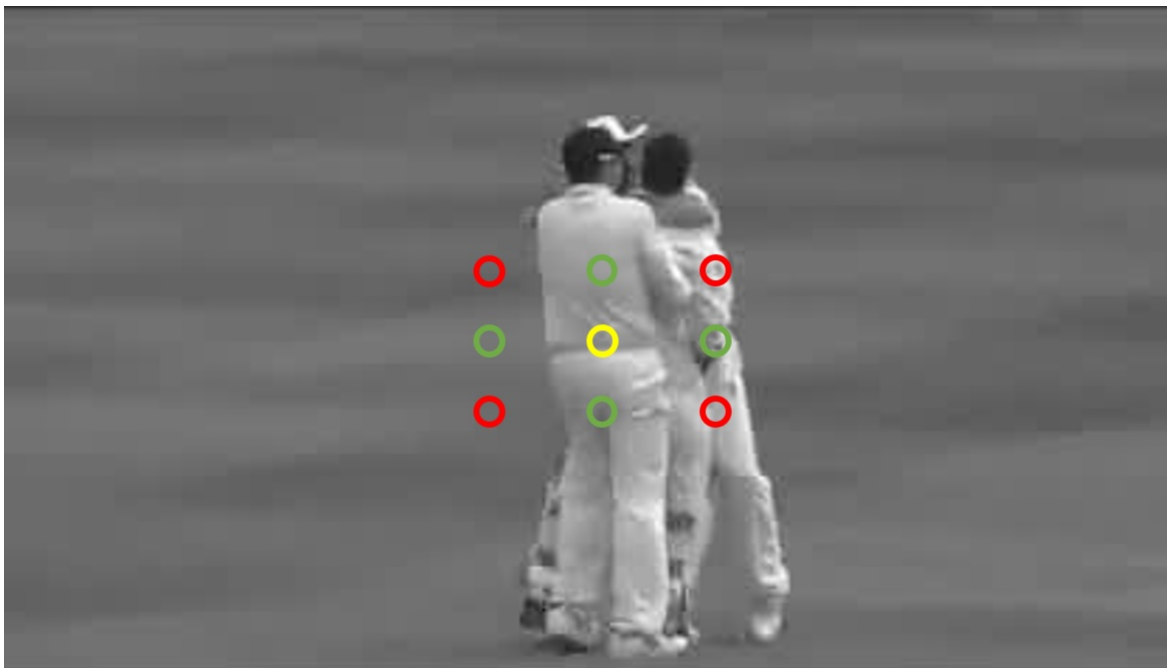


Figure 6.3: PIXL algorithm, reference pixel locations

The hash values for each of these 8 further locations are pre-computed (before match time) for each reference frame. At match time, once candidate matches have been found, the hash codes from the unknown frame are compared with the 8 pre-computed reference hash codes and the greatest correspondence (minimum Hamming distance) between the hash codes of the 8 reference locations and the hash codes for unknown pixel locations is determined. The pixel locations in the unknown with the minimum Hamming distance are then considered in opposite pairs (North & South, East & West, North East & South West, North West & South East; where North is vertically above the central pixel). The

intersection point between the lines joining North & South, and separately for East & West is computed - if this intersection point is close to the central pixel a match is deemed to have been made between the reference and unknown. In addition, the diagonal opposite pairs of points North East & South West and North West & South East are considered in the same way so that if the intersection point between the lines joining these points is close to the central pixel a match is deemed to have been made between the reference and unknown.

The algorithm is effective at removing false matches (false positives) even if the unknown has been re-scaled. This is because in a re-scaled video the intersection location between the two sets of matching points will remain the centre of the frame.

Figure 6.4 illustrates this matching process. This image is a rotated and flipped version of the image in Figure 6.3. While the 8 pixel matching locations in the unknown are different from the reference the intersection point for the opposite vertices still meet at the central pixel.

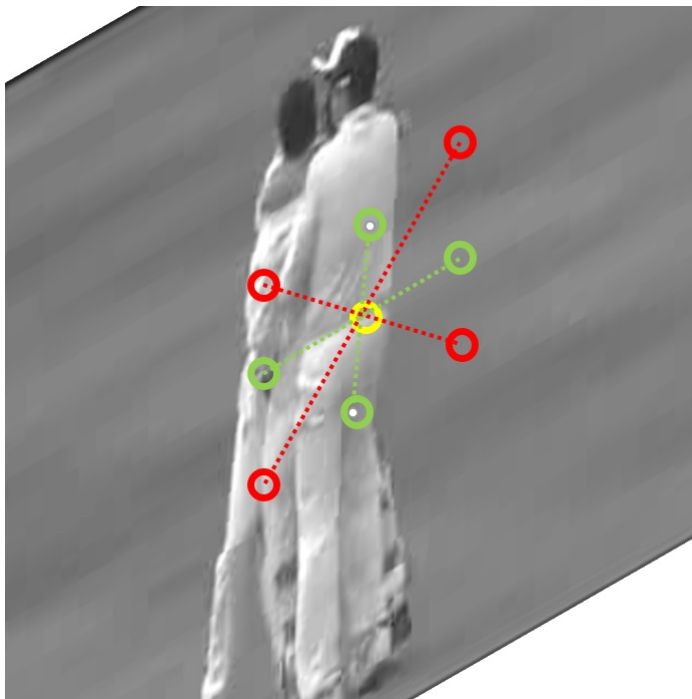


Figure 6.4: PIXL algorithm, reference pixel locations for a rotated image

As can be seen in Figure 6.4, in principle this algorithm will match a reference to an unknown even if the reference has been rotated and/or flipped (vertically or horizontally).

6.2.1 PIXL Algorithm Matching Parameters

The PIXL algorithm uses a number of parameters to control the matching process. As with the other video matching algorithms in this study, the videos are assumed to be formatted with a greyscale colour model, a frame size of 128×72 pixels at 4fps as this is the standard that these videos are stored by the Audible Magic ingest process.

The parameters that are used to control the PIXL algorithm are:

1. Hash code size for each hash table.

The hash code size affects the size of the hash table and the number of match candidates produced. Through initial experimentation a value of 18 bits has been found to produce a good trade-off between the number of match candidates and the number of actual matches found. Further experimentation will be performed to optimise this value for different types of video content.

2. Minimum and maximum number of bits set in hash code.

The PIXL match algorithm relies on the illumination intensity of the central region of the reference frames changing over time. With any static video clip the PIXL algorithm will produce a hash code with few to no bits set. A static video sequence can occur in any video - this increases the likelihood of false positive matches. Therefore, the video sequences where the reference and unknown produce PIXL hash codes with a small number of set bits or a large number of set bits are ignored. The values used for an 18 bit hash code are 3 and 16 for minimum and maximum bits required respectively.

3. Number of bit errors permitted in hash code for initial hash table lookup.

During the lookup process, the unknown hash codes produce a list of candidate matches from the reference hash table. Through experimentation it has been found that more correct matches are obtained if these initial match candidates are supplemented with match candidates with similar hash codes. Therefore, additional match candidates are obtained by flipping each bit of the unknown hash code in turn and looking up each hash code variation in the reference hash table. Only one bit is flipped at a time. If the number of bit errors is set to one with an 18 bit hash code,

19 lookups are performed to produce a set of match candidates. For performance reasons the number of bit errors is either set to zero or one.

4. Size of central region of frame to used to obtain initial match candidates.

To find match candidates, from an unknown video signature, a region based on the centre of the unknown frame is scanned (pixel by pixel) and PIXL hash codes are created at each pixel location within this central region. These hash codes are looked up in the hash table and match candidates are produced. This initial lookup process is designed to find the central pixel of the corresponding reference. The size of the region to scan is defined by the size of central region parameter and is a fraction between 0.0 (just use the central pixel) and 1.0 scan the entire frame). This is the proportion of the total area of the frame, centred on the central pixel of the unknown, that is scanned. Clearly a larger area will require more computation and produce more match candidates.

5. Lookup pixel increment X and Y

As described in the central region of the frame parameter, a region in the centre of the unknown frame is scanned to create PIXL hash codes. To improve computational performance of the match process, the lookup pixel increment can be used to skip pixels in the region. The default values in both X and Y directions is one which uses every pixel location. A value of two or more would skip pixels to reduce the computational load at the expense of potentially missing match candidates.

6. Initial match candidate score threshold.

The initial match candidate score threshold is the fraction of bits found to correspond between the reference and unknown PIXL hash codes for the central pixel over the entire match duration (typically 30 seconds). This parameter reduces the number of initial match candidates by setting a floor on the fraction of bits that are required to match over the entire match duration.

7. Match duration.

To perform a match the middle portion (in time) of the unknown is compared with the entire reference database. The duration of this middle portion is the match

duration. This match duration sets a floor on the overall size of the unknown that can be used in the match process. Videos shorter than the match duration are always reported as not matching.

Figure 6.5 illustrates the use of the PIXL match parameters.

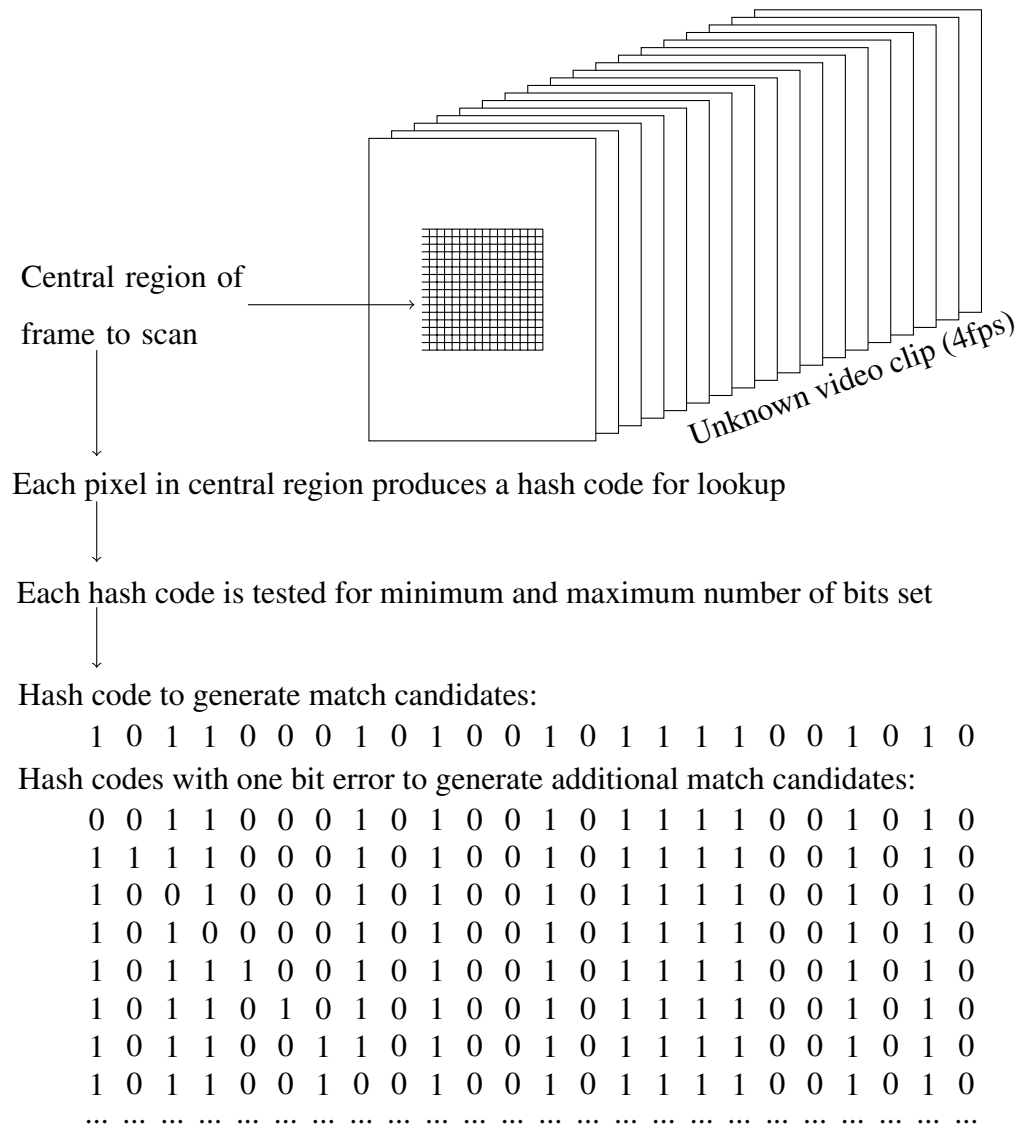


Figure 6.5: PIXL algorithm parameters diagram

6.3 Datasets Used to Test PIXL Algorithm Parameters

The datasets used to test the PIXL algorithm parameters are the same as those used for the GRID match algorithm in Section 4.5.

6.3.1 Simulated Distortions

As discussed in the literature review in Section 2.1.2 and in the GRID algorithm chapter in Section 4.5.1, simulated datasets, where the unknowns are directly derived from the reference videos by a simple distortion operation, tend to produce unrealistically good match statistics.

However, as with the GRID algorithm, simulated unknowns can provide a lower-bound of the quality of match accuracy that might be expected from real-world data. Poor match accuracy from simulated unknowns suggests that the match algorithm will not work well on real-world data.

As with the GRID algorithm, a reference database of 20,000 video items was used in the experiment, these video items represent around 7,750 hours of content. This reference database was supplemented with 263 original sports clips of 5 minutes duration. The unknowns were created from these reference sports clips by distorting in the methods described in the Experimental Methodology chapter in Table 3.1 and Table 3.2.

6.3.2 Real-world Distortions

As with the GRID algorithm experiments, the references for the real-world distortions dataset were taken from video signatures obtained from Audible Magic (AM) content owners. This content includes around 200 feature film titles, 40 animated television cartoon episodes (The Simpsons), as well as, a number of television series and sporting events amounting to a total of around 150,000 items of varying duration from several hours to several seconds. In all cases, a degraded version of the original video (128×72 , greyscale, 4fps) was used as the basis of the reference signature. This is because this degraded version is the only form of the original video that is available for signature generation. The original videos are not delivered to AM.

To facilitate experimentation, a subset of the full AM reference database was randomly selected for analysing the PIXL algorithm parameters. This subset has 20,000 video signatures which represents around 8,000 hours of video or one eighth of the total.

Two sets of unknowns were used for these experiments. Firstly, for the true positive test, a set of film clips downloaded from YouTube where the full feature film is known

to be contained within the reference dataset. Secondly, for the false positive test, a set of sports events, cut into 5 minute clips, known not to be in the reference dataset. The true positive and false positive test are performed for each set of PIXL algorithm parameters.

6.4 Results using Simulated Distortions

Using the simulated distortions database, a set of experiments was performed using standard and complex distortions described in Tables 3.1 and 3.2. The points on the graphs show the true positive and false positive rates from this test. The first Figure 6.6 shows standard distortions where the goal is a true positive rate of at least 90% and a false positive rate of not more than 1%. The second Figure 6.7 shows the results with the more complex distortions which would be desirable to match.

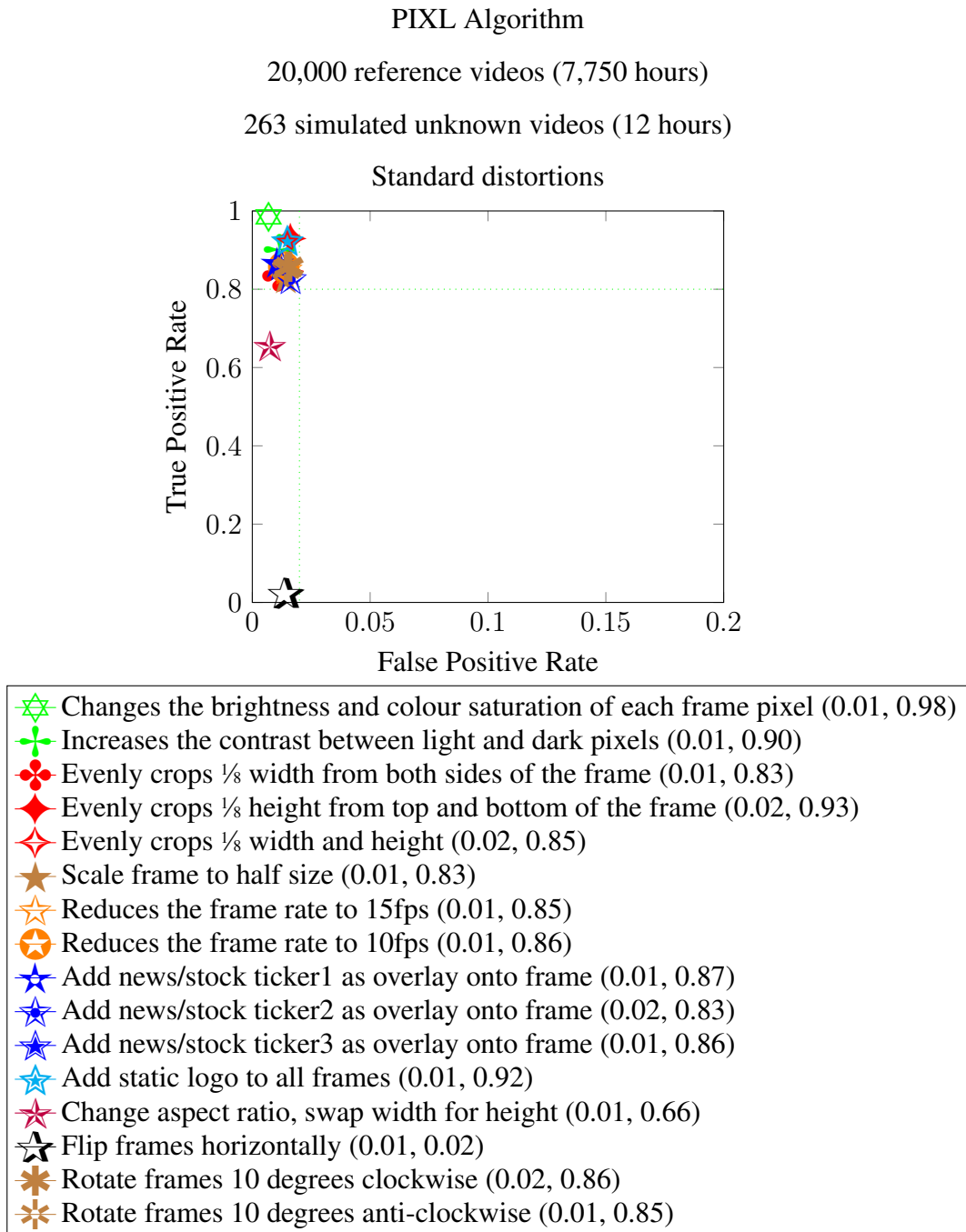


Figure 6.6: PIXL algorithm, simulated unknowns, standard distortions match results

The results for the standard distortions shown in Figure 6.6 shows that matching for most distortions produces a true positive rate of 80% or more and a false positive rate of 2% or less, these are the minimum acceptable bands of acceptability as described in Section 1.4. The two exceptions are the change aspect ratio and horizontal flip distortions. Conceptually the PIXL algorithm should be able to match the reference to these distortion

types but it is unclear why in this case poor match results are obtained. This may be due to pixel dislocation or temporal misalignment.

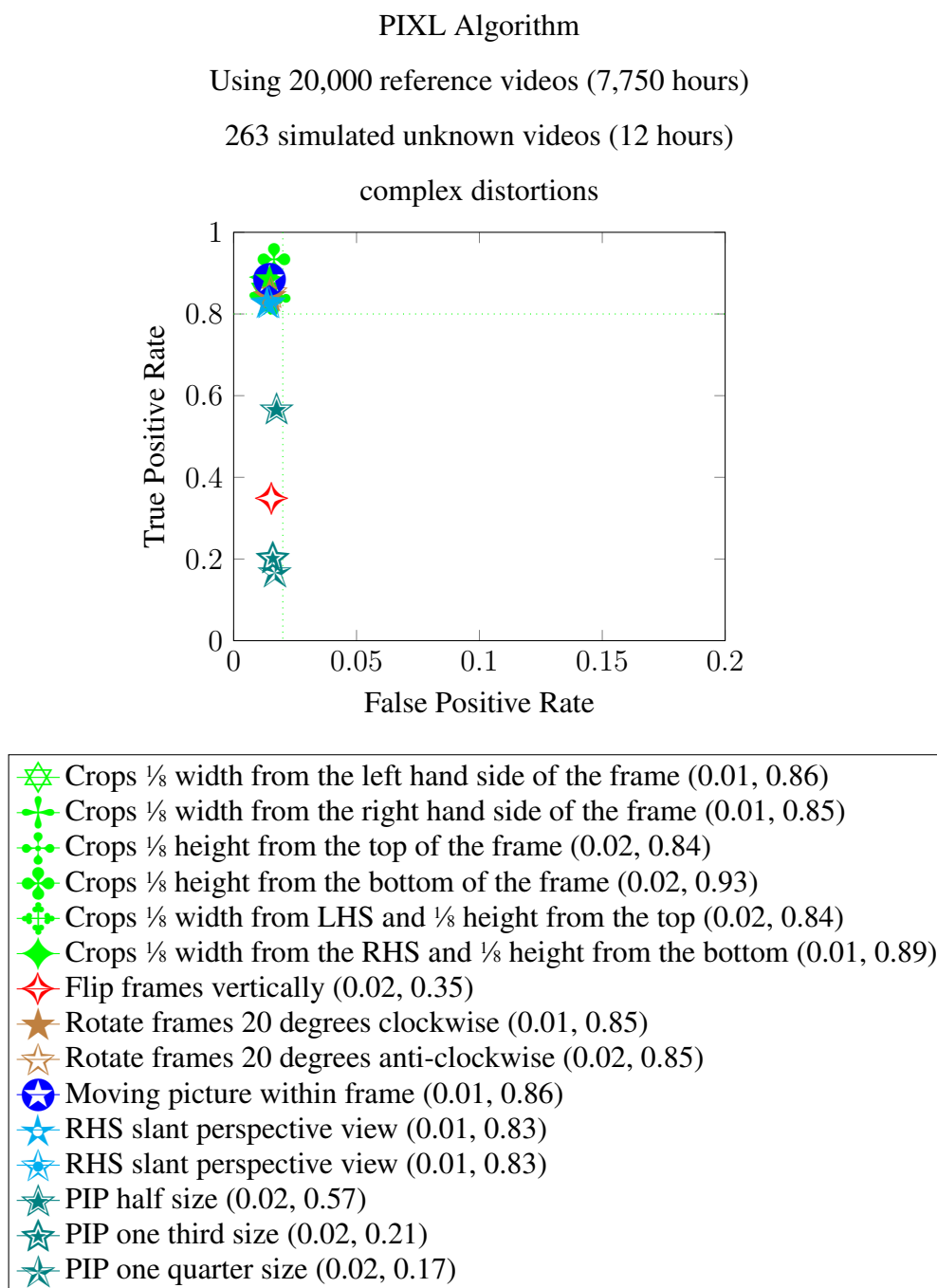


Figure 6.7: PIXL algorithm, simulated unknowns, complex distortions match performance

The results for complex distortions are shown in Figure 6.7. This figure shows that the unevenly cropped videos are matched within the minimally acceptable criteria while the

accuracy of the picture in picture (PIP) and the vertically flipped videos are significantly lower. The false positive rate does not exceed 2% for any of these distortions showing that the false positive rejection algorithm can potentially meet the minimal acceptable standard but is short of the desirable standard of 0.1%.

6.5 Results using Real-world Videos

The results in this section show the performance of the PIXL algorithm using the feature film database, described in the Appendix D. The references in this database are degraded versions of the original definitive videos (128×72 , greyscale, 4fps) while the unknowns are clips from these videos taken from various Internet websites.

This database will be used to optimise the values of the PIXL matching algorithm parameters described in Section 6.2.1.

6.5.1 Hash Code Length

The hash code length affects the size of the hash table and the number of match candidates produced. Through initial experimentation a value of 18 has been found to produce a good trade-off between the number of match candidates and the number of actual matches found. An experiment that used various values of hash code length and match duration was performed with the results shown in Figure 6.8.

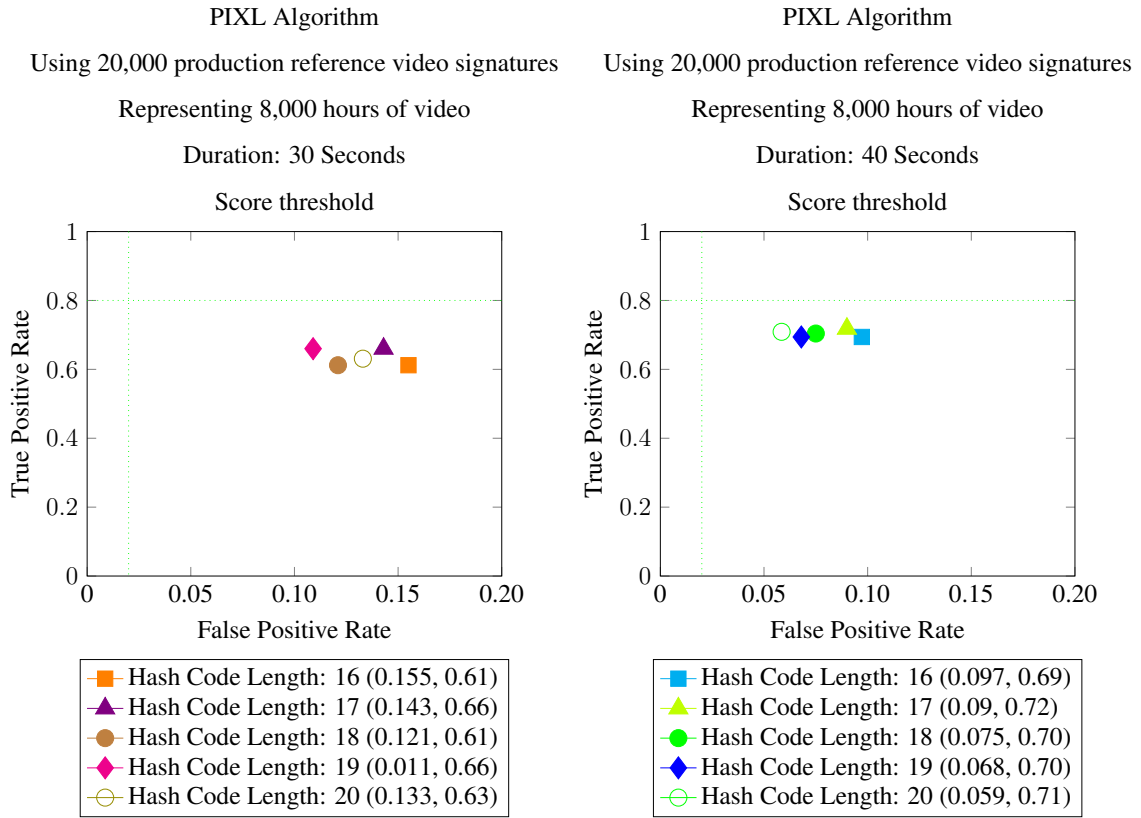


Figure 6.8: PIXL algorithm, real-world videos, hash code length evaluation

Two sets of results are shown in Figure 6.8 with a match duration of 30 and 40 seconds. Clearly the match duration of 40 seconds provides superior results as the true positive scores are higher and the false positive scores are lower for each hash code length. Therefore the following discussion will focus on a match duration of 40 seconds.

The results in Figure 6.8 with a match duration of 40 seconds show that the hash code length used for the hash table affects the proportion of false positives while leaving the true positive score largely unchanged. To minimise the false positive rate the longer hash code lengths are preferred, however a hash code length of 18 produces a true positive rate of 70% for a modest increase in false positive rate. Therefore, on the feature film dataset a hash code length of 18 with a match duration of 40 seconds produces the optimal results.

6.5.2 Number of Bits Set in Hash Code

The hash code length controls the size of the hash table that is used to create match candidates. However, static video clips are likely to produce hash codes with either very few or many bits set. This will likely cause false positive matches as many unrelated

videos could have the same bit pattern at the extremes of the hash code patterns. To reduce the probability of these unrelated videos creating match candidates an acceptable range for the number of hash code bits is defined to declare a match candidate. This section assumes a hash code size of 18 and varies the minimum and maximum acceptable number of bits set to determine the optimal value for this parameter.

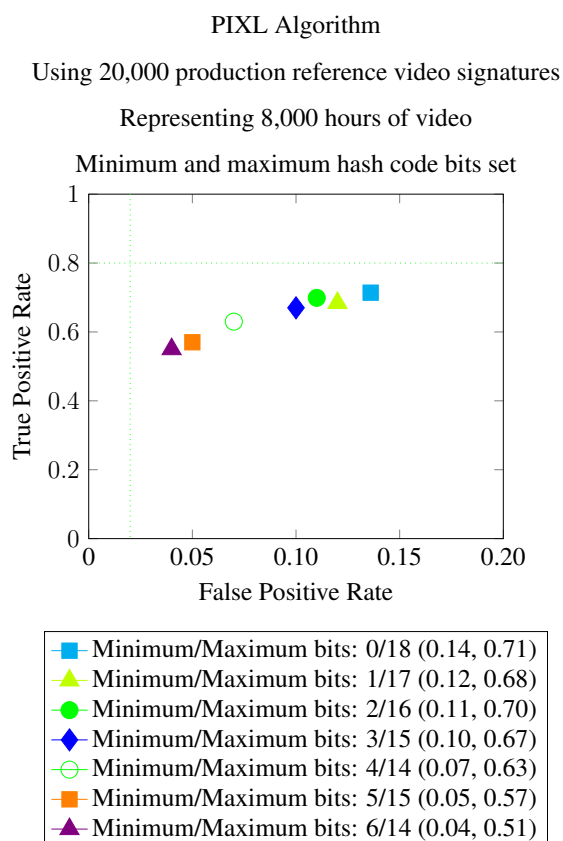


Figure 6.9: PIXL algorithm, real-world videos, number of bits to match evaluation

Figure 6.9 shows that reducing the accepted range between minimum and maximum number of hash bits has a modest affect on the true positive rate for the first few bits and also reduces the false positive rate slightly. The feature film dataset used for this test probably has few blank or repetitive frames so this filter has a limited impact on the matching performance.

6.5.3 Number of Bit Errors

The number of bit errors is a parameter that can increase the number of match candidates obtained from the hash table. If the number of bit errors is zero a single hash code is used for the hash table lookup. If the number of bit errors is one, a lookup is performed on the original hash code, as well as, all hash codes that are one bit different from the original hash code. Thus far, we have settled on the value of 18 bits for the hash code. Therefore with one bit error, 19 hash code lookups are performed rather than the single lookup that would be performed if this parameter were zero. Clearly there is a computational cost to using a non-zero bit error value as not only are significantly more lookups performed on the hash table but also these lookups produce additional match candidates that will require further processing.

The result of using zero and one bit errors is shown in Table 6.10.

Bit errors	true positive proportion	false positive proportion
0	0.723	0.058
1	0.704	0.150

Figure 6.10: PIXL algorithm, impact of bit errors on matching accuracy

Table 6.10 shows that using a bit error of 1 reduces the true positive rate and increases the false positive rate - we therefore select a bit error value of 0.

6.5.4 Area of Central Region to Scan

As described in the PIXL algorithm parameters Section 6.2.1, the area of the unknown to scan for central pixel matching is a critical parameter for the performance and accuracy of the match process. In particular, a video that has been unevenly cropped will displace the central pixel and thus a match will not be obtained unless the pixels around the central pixel are considered as potential match candidates.

A number of values for the fractional central area where chosen to explore this parameter and the results are shown in Figure 6.11. This fractional central area parameter ranges between 0.0 (only use the central pixel) and 1.0 (scan the entire frame). Clearly

the larger the area used, the higher the computational cost and the larger number of match candidates generated.

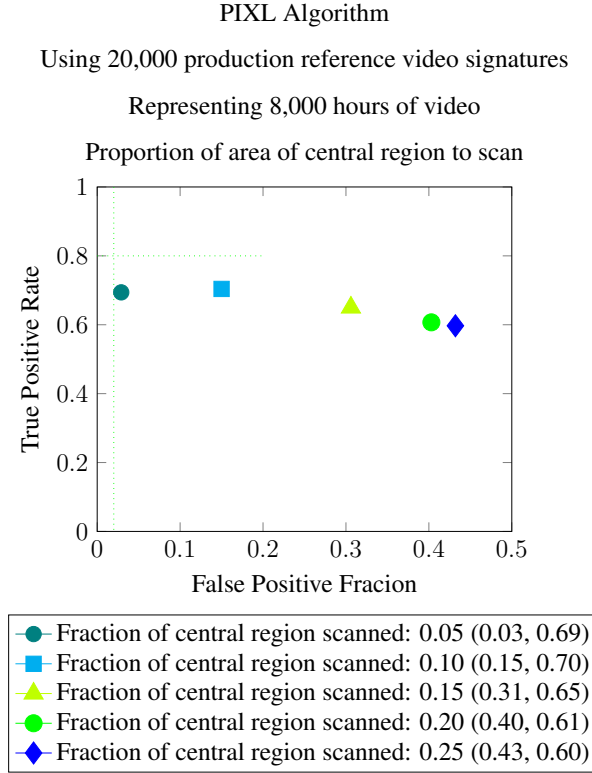


Figure 6.11: PIXL algorithm, real-world videos, fraction of central region to scan evaluation

Figure 6.11 shows that the true positive score peaks with an area fraction of 0.1 however a significant number of false positives are lost when using a smaller value for the area fraction. This smaller value for the area fraction would result in the match algorithm being less robust to uneven cropping since a relatively small area around the central pixel would be scanned for matches. To drive down the false positive rate a value of 0.05 is preferred with this dataset.

6.6 Throughput

As described in the objectives Section 1.7 of this study, the throughput objective is that the time taken for an average lookup transaction should be less than a second. This transaction rate should be achieved against a reference database of between 10,000 to 50,000 hours

of video. The hardware available for this transaction rate is a standard Linux-based server with 8 GPU cards at a cost of approximately \$25,000.

For this throughput analysis the feature film dataset (described here in Appendix D) was bulked out with noise videos to create a reference dataset of 61,000 hours of reference video. The PIXL algorithm lookup parameters were set with a hash code size of 17, min/max bits of 2 and 16 respectively, bit errors of zero and a match duration of 40 seconds. Two thousand match transactions were performed using a 50:50 ratio of items in the reference dataset and items not in the reference dataset. The results from this experiment are shown in Figure 6.12.

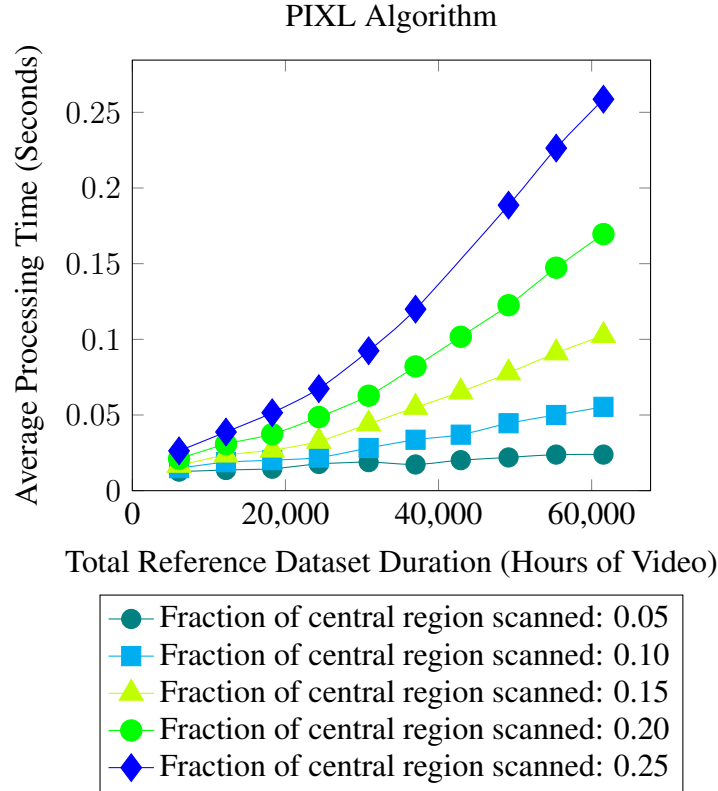


Figure 6.12: PIXL algorithm, real-world videos, processing time vs reference database size

Figure 6.12 shows the average time taken to perform a lookup transaction for different reference database sizes with a range of values for the fractional area of the central region to be scanned. This graph shows that the transaction duration increases as the scanned area increases. Clearly the experiments where a smaller central region is scanned produce much higher transaction rates.

Figure 6.12 also shows that the throughput rate for the PIXL algorithm with a 61,000 hour video database using a central scanned region of 0.05 is $\frac{1}{0.024} = 42$ or 42 transactions per second.

6.7 GPU Memory Requirements

The PIXL algorithm only requires 9 bits per frame to construct the reference signature. Therefore, the GPU memory requirement is less than 10 bytes per second of video including the indexing and other overheads. This compares very favourably with the GRID algorithm that needs at least 400 bytes per second for a $5 * 5$ grid. This PIXL memory requirement is well below the GPU memory per second metric as stated in the video matching objectives in Section 1.4. Assuming that a standard GPU card has 4 Gbytes of memory available for reference signatures, it would be possible using the PIXL algorithm to store over 100,000 hours of reference video signatures on a single GPU.

6.8 Summary

The strengths of the PIXL algorithm are:

1. The algorithm is not affected by changes on the edges of the frame (such as cropping) since it only requires pixels close to the centre of the frame to be present in the unknown to obtain a match, see Figure 6.1. These types of changes include overlaid tickers and static logos which are generally displayed at the bottom of the video frames.
2. The algorithm is resilient to rotations and flips as it uses the relative location of nine separate pixels rather than fixed locations within a frame to match frame sequences, see Figure 6.4.
3. The algorithm is also resilient to contrast changes since the PIXL hash code depends on pixel intensity differences rather than the absolute value of the pixel intensity, see Table 6.1.

4. The reference hash codes are small as they only require 9 bits per frame, see Table 6.1.

The weaknesses of the PIXL algorithm are:

1. The algorithm is sensitive to frame sequence and timing changes as it relies on the exact frame sequence and timing to construct the PIXL bitmap.
2. Distortions to the central region of the frame are likely to make matching with this algorithm difficult. However, it is unusual for distortions to be made to the central area of a video.
3. Using a small central region to scan for matches produces a high transaction rate with a low false positive rate, see Figure 6.11, however there is a danger that this approach will not match reference videos that have been cropped and thus produce a high false negative rate.
4. The algorithm can produce a large number of false positives as the algorithm uses relatively little information from the underlying references to obtain a match candidate. A more rigorous false positive detector could probably be devised that compares more information from the unknown and match candidates that would reduce the number of false positives at the penalty of additional compute capacity.

The PIXL algorithm has a number of parameters that can be varied to produce a higher match rate with an acceptable false positive score. A number of experiments were performed to determine the best values for these parameters on a reduced-sized dataset. The result of this analysis is as follows:

PIXL algorithm parameter	Optimal value
Hash code size	17
Min/Max bits to be set	2/16
Bit errors	0
Relative size of central region to be scanned	0.05
Match duration (seconds)	40

Figure 6.13: PIXL algorithm optimal parameters table

Using these values with the real-world feature film database, a true positive rate of 70% and a false positive rate of 3.0% was achieved with a throughput of 42 transactions per second. The match accuracy values are below the objectives for this study described in Table 1.4 however this algorithm could potentially be combined with other algorithms to get closer to the overall goal as the transactions per second and GPU bytes per second of video are well within the objectives of this study.

CHAPTER 7

Pre-trained Convolutional Neural Networks

Pre-trained convolutional neural networks such as ResNet [37], DenseNet [79], and EfficientNet [80] have been shown to accurately classify images into various object categories. While the objective of this study is to match video clips, these networks can be suitably modified to extract feature vectors from individual video frames. The feature vectors derived from each frame provide an abstract representation of the frame contents, making them potentially robust to various types of image distortion. This approach has the capacity to match distorted video frames to the reference video frames from which they are derived. A sequence of such video frame matches could then match a distorted unknown video clip to its corresponding reference video sequence. Therefore, these feature vectors can be used in this study to match video clips.

Feature vectors will be extracted from a set of reference videos to create a reference database. Similarly, feature vectors will be extracted from the frames of unknown video clips to be matched against the reference videos. The feature vectors from the reference and unknown video frames will then be compared, and a distance score will be calculated. This score will be accumulated across a set of consecutive frames from the reference and unknown videos over the match duration (typically 30 seconds). The accumulated distance score will determine whether the unknown and reference videos are a match.

For performance reasons, the videos will be sampled at a relatively low frame rate (4 fps), and the feature vectors will be compared using the high-performance FAISS [52] indexing software library. Once a near match between individual frames of the reference and unknown videos has been established, a candidate match will be obtained. This candidate match will be further analyzed by comparing the feature vectors from consecutive frames of the reference and unknown videos, using the initial frame match as an alignment point, to compute an overall match score.

To establish a match score threshold, definitive reference videos will be compared against distorted versions (unknowns) of the same reference videos. The threshold will be set at a point where no more than 1% of the comparisons yield a false positive match (i.e., the unknown is incorrectly matched to the wrong reference in 1% of cases). Once this threshold is established, any candidate matches with a score exceeding the threshold will be considered valid matches. The fraction of correct matches from a set of lookups will represent the true positive rate, which will measure the effectiveness of the matching process.

The PyTorch [81] machine learning framework provides several pre-trained neural network models, trained on the ImageNet dataset [82], which supports the classification of images into a thousand distinct classes. This chapter explores the use of these pre-trained convolutional neural networks for video matching.

7.1 Use of Pre-trained Models in Video Matching

The PyTorch [81] pre-trained models such as ResNet, [37], DenseNet [79], EfficientNet [80], MobileNet [83] and SqueezeNet [84] have been designed for image classification. However, by removing the final fully connected classification layer, a feature vector that represents the characteristics of the input image can be obtained.

7.1.1 Resnet-18

The summary of the ResNet-18 model in PyTorch [81] is shown in Figure 7.1.

```

ResNet(
  (features): Sequential(
    (init_block): ResInitBlock(
      (conv): ConvBlock(
        (conv): Conv2d(3, 64, kernel_size=(7, 7), stride=(2, 2),
          padding=(3, 3), bias=False)
        (bn): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
          track_running_stats=True)
        (activ): ReLU(inplace=True)
      )
      (pool): MaxPool2d(kernel_size=3, stride=2, padding=1,
        dilation=1, ceil_mode=False)
    )
    (stage1): Sequential(...)
    (stage2): Sequential(...)
    (stage3): Sequential(...)
    (stage4): Sequential(...)
    (final_pool): AvgPool2d(kernel_size=7, stride=1, padding=0)
  )
  (output): Linear(in_features=512, out_features=1000, bias=True)
)

```

Figure 7.1: ResNet-18 model structural diagram

To preserve the weights contained within the pre-trained ResNet-18 model, outlined in Figure 7.1, the final layer can be replaced with an identity layer. This replacement removes the final classification layer (which provides the class probabilities) with a non-processing layer which then provides access to the previous layers output. This output can be used as a feature vector to represent the overall input to the model [85]. The replacement identity layer directly transfers its inputs to its outputs. For ResNet-18 this results in a output vector of 512 features.

7.1.2 ResNet-50

To use the pre-trained ResNet-50 [86] neural network a similar adjustment to the model has to be made. In ResNet models, the convolutional layers use a window of size 3×3 ,

the number of filters increases from 64 in ResNet-18 to 2,048 in ResNet-50.

The weights used in this default model improve upon the results described in the original ResNet study [87] by using the TorchVision new training recipe [86]. The composition of the ResNet-50 model is shown in Table 7.1.

The ResNet-50 model can be modified in the same way as the ResNet-18 model.

Pre-trained Neural Network	Accuracy 1% on ImageNet-1K	Accuracy 5% on ImageNet-1K	Params	GFLOPS
ResNet-50	80.9	95.4	25.6M	4.09

Table 7.1: Composition of ResNet-50 model

As with the ResNet-18 model, the final linear layer maps the output of the network to a vector of probabilities for the 1,000 classes that the model has been trained on. As before an identity layer is used to over-write this final layer so that a 2,048 feature vector is produced. This feature vector is then used in the video clip matching process.

7.2 Computation of the Pre-trained Neural Network Video Signature

In keeping with the other matching methods used in this study, the neural network feature vectors are computed at a rate of 4 frames per second. The frame rate is arbitrary but attempts to balance the volume of output data against the timing accuracy (discretisation error) of the matching process.

The computational cost of computing a pre-trained neural network video signature (NNET) is high. This is because each frame (4 fps) has to be processed separately by the neural network to produce a feature vector. As with other video signatures the media file is first split into frames so that the individual frames can then be processed separately. The processing cost for the references is less important than for the unknown, as creation of the reference signatures can be performed off-line. However, the processing cost for unknown signatures is significant as this has to be performed at match time. To assess the performance of the lookup process, the time taken to create the video signature has to be

ResNet-50 Network

```
ResNet(  
  (features): Sequential(  
    (init_block): ResInitBlock(  
      (conv): ConvBlock(  
        (conv): Conv2d(3, 64, kernel_size=(7, 7), stride=(2, 2),  
          padding=(3, 3), bias=False)  
        (bn): BatchNorm2d(64, eps=1e-05, momentum=0.1,  
          affine=True, track_running_stats=True)  
        (activ): ReLU(inplace=True)  
      )  
      (pool): MaxPool2d(kernel_size=3, stride=2, padding=1,  
        dilation=1, ceil_mode=False)  
    )  
    (stage1): Sequential(...)  
    (stage2): Sequential(...)  
    (stage3): Sequential(...)  
    (stage4): Sequential(...)  
    (final_pool): AvgPool2d(kernel_size=7, stride=1, padding=0)  
  )  
  (output): Linear(in_features=2048, out_features=1000, bias=True)  
)
```

Figure 7.2: ResNet-50 model structural diagram

added to the overall time taken to perform the match itself adding to the overall response time of the system. A requirement of this study is to process each transaction (including signature creation) within one second, as described in the overall objectives of this study in Section 1.4.

The image size required as input to the pre-trained networks is $224 \times 224 \times 3$ (224 pixels wide, 224 pixels high with 3 colour channels), however the frame size of the videos provided by the Audible Magic content suppliers is $128 \times 72 \times 1$ (128 pixels wide, 72 pixels wide, greyscale (8 bit)). The degraded frame size is due to content security concerns of the customers - i.e. the distribution of a full size, full colour video would potentially undermine their business model. The conversion of the customer provided $128 \times 72 \times 1$ videos to $224 \times 224 \times 3$ videos can be performed in a number of ways. The methods explored here are:

1. Image insertion: The image insertion method simply copies the supplied 128×72 greyscale pixels into the centre of three colour planes of 224×224 leaving the surrounding pixels blank (zeros), see Figure 7.3.
2. Image stretching 1: The 128×72 supplied greyscale image is stretched by bilinear interpolation to fit into three identical 224×224 colour planes, see Figure 7.4.
3. Image stretching 2: The 128×72 supplied greyscale image is stretched by bicubic interpolation to fit into three identical 224×224 colour planes, see Figure 7.4.

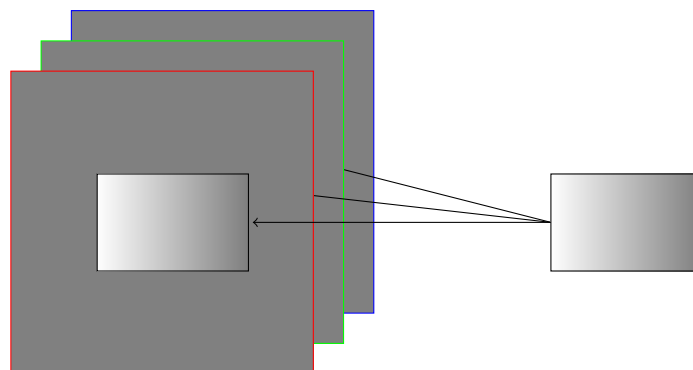


Figure 7.3: Pre-Trained neural network, image insertion technique

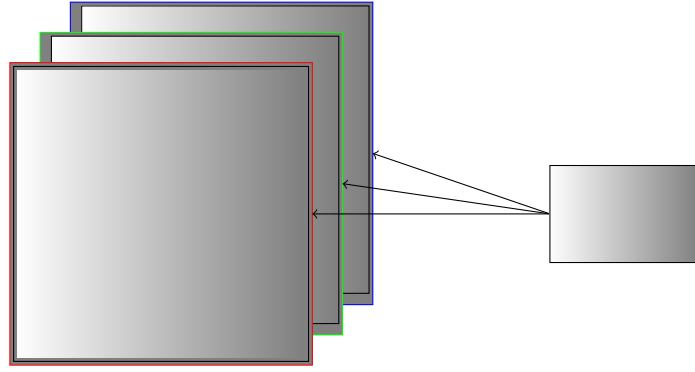


Figure 7.4: Pre-Trained neural network, image stretch technique evaluation

These image transformations are not desirable because either a significant area of the output image is unused in the case of image insertion, or a significant amount of interpolation is involved in the case of image stretching. However, the models were mainly trained on the ImageNet dataset [82], and data augmentation techniques such as cropping, rotating, and flipping were applied to the training images to avoid overfitting [88]. The transformations applied to the original training set for these pre-trained models provide some assurance that a representative feature vector will be produced for the video clips processed in this study.

7.3 Matching Process

Using essentially the same methodology as the GRID algorithm described in Chapter 4, the feature vectors for the reference frames are loaded into an in-memory database and indexed by FAISS [52]. FAISS provides a high performance nearest match interface that can be used to obtain match candidates from the query video. The reference video signatures are computed off-line and loaded into the FAISS index prior to match time.

At match time, a two step process is used to identify video matches. In the first step, candidate matches are obtained from a similarity search process using the feature vectors from each unknown frame within the lookup window; in the second step, the match candidates are used as an alignment point (in time) between the unknown and reference videos and a Euclidean distance of the feature vectors is calculated between the corresponding sequence of unknown and reference vectors. A match is declared if

the total Euclidean distance between reference and unknown is below a certain threshold. The overall match score is computed from the average Euclidean distance of the feature vectors over the match duration (typically, 30 seconds).

7.3.1 Blank Frames

Blank frames (frames of a uniform colour often just black) must be ignored for matching purposes as they could appear in any video, will match with other blank frames and are likely to cause false positives. The same blank frame detector is used for all the algorithms in this study, see Algorithm 1. Frame sequences that include blank frames will not be used in the feature vector comparisons.

7.3.2 Frame by Frame Matching

To illustrate the way that the matching process works, a reference and slightly distorted (contrast change only) unknown were used. ResNet-18 feature vectors were computed at 4 frames per second for both the reference and unknown videos. A thirty second section of the unknown was compared with a 30 second section of the reference. This 30 second comparison is performed at a number of consecutive reference frame offsets with a distance score between the reference and unknown calculated at each frame offset location. A frame offset of zero represents the alignment of the reference and unknown. For each frame offset an average Euclidean distance between the frame vectors for the 30 second section of the unknown and reference was computed. The results of this experiment are shown in Figure 7.5.

NNET Algorithm
ResNet-18 pre-trained neural network
512 Element vector
30 second comparison between reference and unknown

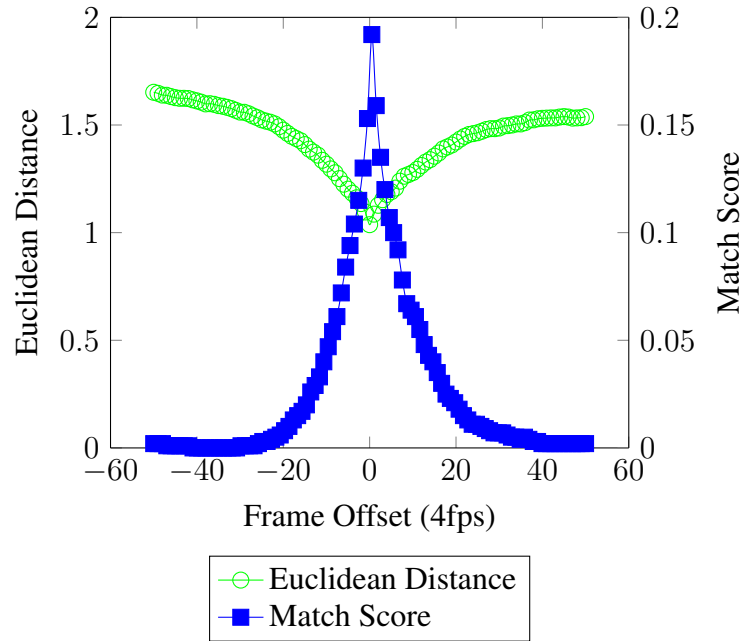


Figure 7.5: Pre-Trained neural network ResNet-18, distance and match score for single match

Figure 7.5 shows that the Euclidean distance (green circles) is minimised, as expected, when the reference and unknown clips are fully aligned (frame offset equal to zero). The alignment point is the inverse peak with the Euclidean distance increasing rapidly as the frame offset moves away from zero. As the frame offset moves further away from zero, in both positive and negative directions, the change in Euclidean distance declines rapidly. The decline in Euclidean distance shows that the feature vectors are very similar at the alignment point. A match score (blue squares) is computed from these frame offset values so that a positive threshold can be used to determine whether the video clips have matched. The match score is computed as:

$$D_{ru} = \frac{1}{N} \sum_{t=1}^N \left(\sqrt{\sum_{j=1}^n (f_r(i, t) - f_u(i, t), t)^2} \right) \quad (7.1)$$

$$M_s = \frac{1}{D_{ru}}$$

Where D_{ru} is the Euclidean distance between the reference $f_r(i, t)$ and unknown $f_u(i, t)$ feature vectors for N frames. n is the number of elements in the feature vectors.

The match score shows a sharp peak at the alignment point so that a match score threshold can be established.

7.3.3 Multiple Frames per Vector

In Section 7.3.2 in this chapter, matching techniques that used a single video frame per neural network feature vector were described.

In this section, techniques that use multiple frames per vector will be explored. Two separate techniques are proposed. Firstly, for each frame location of the unknown, three consecutive frames (with bilinear interpolation to fill each 224×224 plane) were loaded into the separate colour planes of the $224 \times 224 \times 3$ image used for input into the ResNet-18 neural network. The resultant vectors from the ResNet-18 network were then compared as before. Secondly, for each frame location of the unknown, consecutive frames were tiled in a 2×3 pattern (2 images horizontally and 3 images vertically) across the 224 plane required by the ResNet network. As the source frames are 128×72 the frames were cropped horizontally to fit (8 pixels were removed from each edge) and grey pixels were added to the bottom of the frame to fill the 224×224 plane. These tiled images were then used to create feature vectors and compared as before.

As initially described in the image transformation Section 7.2 these input image restructuring steps are well beyond the scope of the original training images (ImageNet dataset [82]) for these pre-trained models. However, since data augmentation techniques such as cropping, rotating, and flipping were applied to the training images to avoid over-fitting [88]. The transformations applied to the original training set for these pre-trained

models provide some hope that a representative feature vector will be produced for these multi-frames per image inputs.

7.4 Results

The match performance using various distortions on the sport database are shown below. The different frame transformation algorithms are compared using a combination of standard and complex simulated distortions with a number of neural networks.

7.4.1 Simulated Distortions

As discussed in the methodology Section 3.3.2, simulated datasets, where the unknowns are directly derived from the reference videos by a simple distortion operation, tend to produce unrealistically good match statistics. However, real-world databases that contain both high-quality definitive reference videos and distorted clips of these references are not available. Simulated unknowns can provide an upper-bound of the quality of matching that might be available from real-world data (i.e. the highest potential value for a true positive rate for a specific false positive rate). If poor match performance is obtained with simulated unknowns we can be reasonably certain that real-world data will produce even poorer results. Hence, simulated unknowns provide good evidence for situations where a particular match algorithm will be weak.

For this test, only the sports dataset was used (263 original sports clips of 5 minutes duration distorted in 32 ways, 8,416 unknowns) as the time taken to compute a set of video signatures exceeded 48 hours as each video signature took over tens of seconds to compute. The unknowns were created from these reference sports clips by distorting them (using ffmpeg) in the ways described in Tables 3.1 and 3.2.

Figure 7.6 shows the matching performance of the ResNet-18 feature vectors using a range of values for the match score threshold. Each video clip comparison used 30 seconds of the unknown feature vectors at 4 fps (120 feature vectors from the middle of the clip) against all the feature vectors in the reference database. The project objective is a true positive rate of at least 90% and a false positive rate of not more than 1% as described in the objectives Section 1.4. The match score threshold is determined by selecting a value

that corresponds to fewer than 1% false positives.

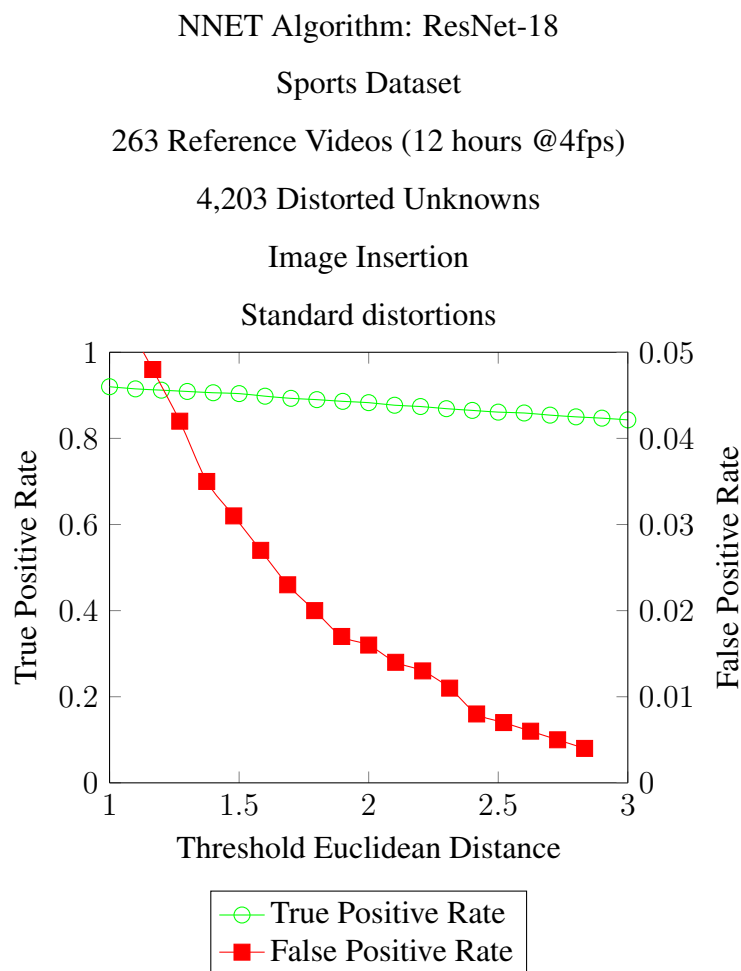


Figure 7.6: Pre-Trained neural network ResNet-18, match score threshold vs euclidean distance

Figure 7.6 shows that a false positive rate of 1% can be achieved (with this limited dataset) using a threshold Euclidean distance of 2.2. This corresponds to a true positive rate of 87% which is slightly below the 90% target. Unfortunately, this is likely to be a significant over estimate of actual performance on real-world data due to the limited size of the reference database and the ease of matching simulated distorted video clips. However, as the image insertion technique depicted in Figure 7.3 was used to transform the frames and other neural networks are available it may be possible to improve these results.

7.4.2 ResNet-18

Figure 7.7 shows the results for the ResNet-18 network using the match threshold score of 2.2 determined in Section 7.4.1. The reference database consists of 263 video clips of 5 minutes duration. Each of the reference video clips were distorted in all the ways described in Figure 7.7 and these distorted clips were used as unknowns to be matched against the reference database. Each point in the graph summarizes the 263 lookups of the distorted unknowns.

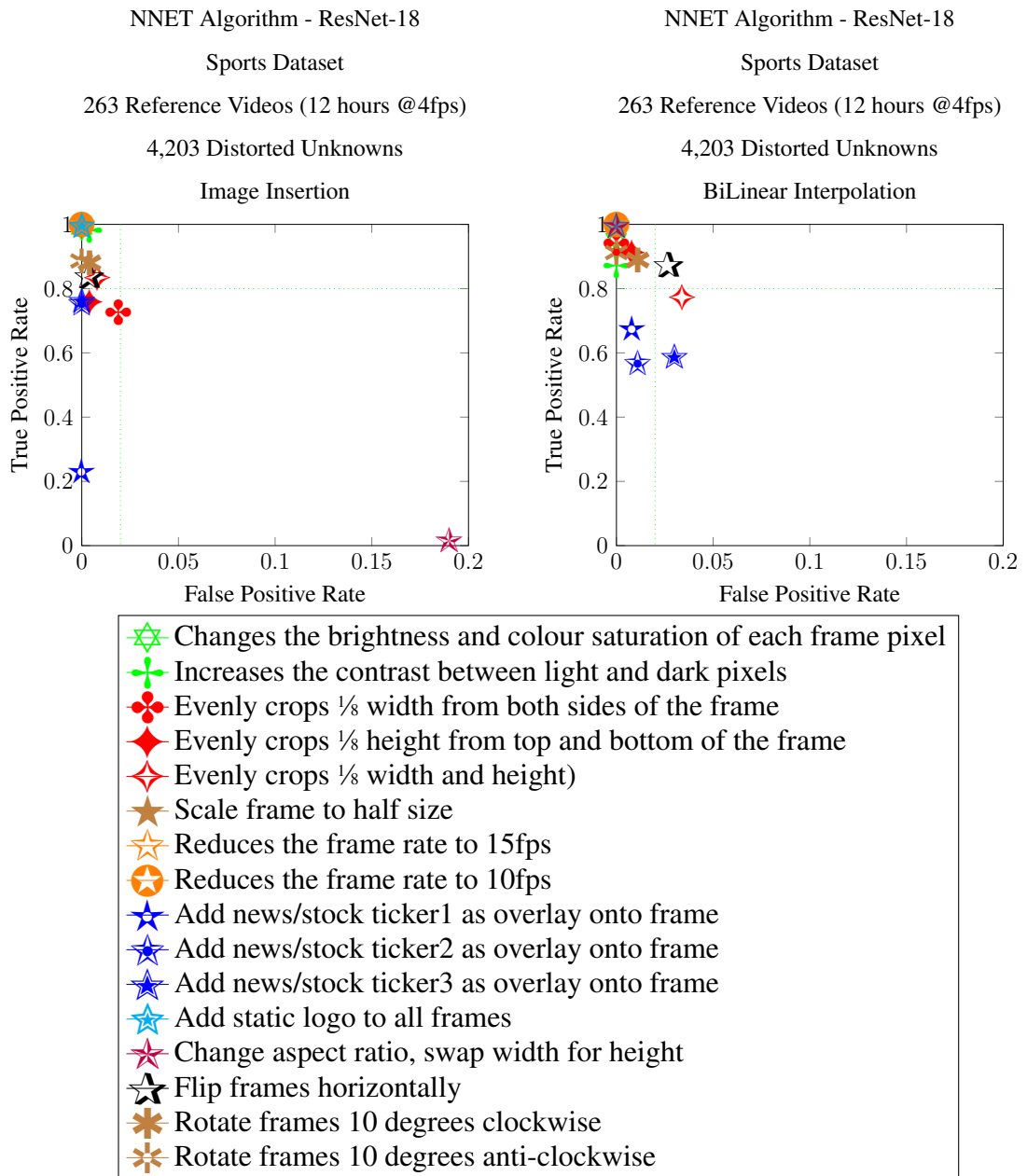


Figure 7.7: Pre-Trained neural network ResNet-18, simulated unknowns, standard distortions match performance

The standard distortions shown in Figure 7.7 shows that most of these distortions produce a true positive rate above 70%. The left hand graph uses the image insertion technique and produces two outliers, the outliers distortions are an overlaid news ticker and the aspect ratio change. The overlaid stock ticker distortion might be expected to produce significantly different feature vectors for each of the frames as the content is different in part of the frame. The very low true positive rate for the aspect ratio change distortion is somewhat unexpected. However, the image insertion technique does not attempt to stretch the frames to fill the $224 \times 224 \times 3$ area that the neural network is operating on. The bilinear interpolation graph has no outliers but the match performance of the all the three distortions with overlaid news/stock tickers is below 70% with the other distortions similar to the Image Insertion results.

Figure 7.8 shows the matching results for complex distortions with the same ResNet-18 neural network.

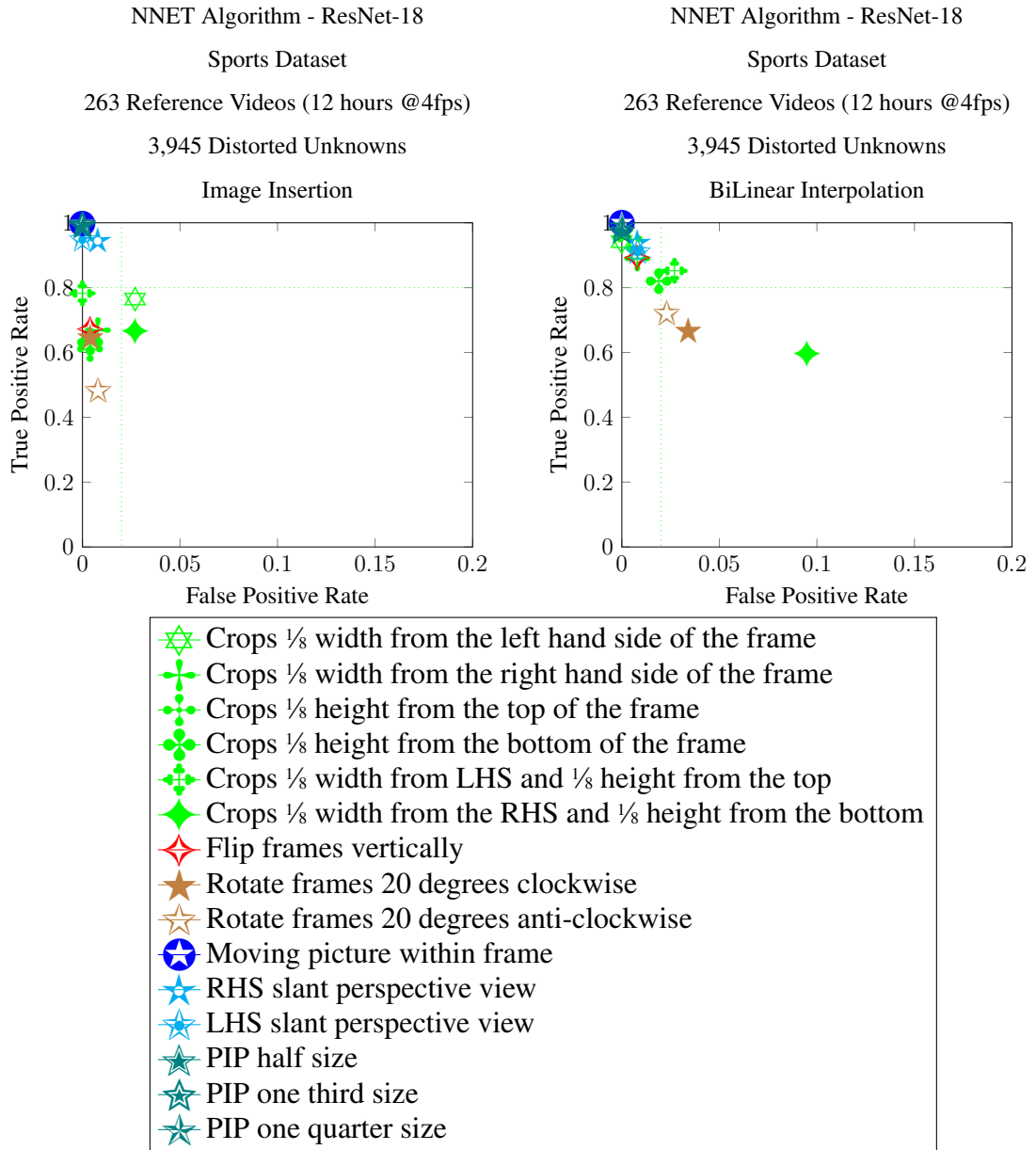


Figure 7.8: Pre-Trained neural network ResNet-18, simulated unknowns, complex distortions match performance

The results for complex distortions shown in Figure 7.8 show a greater distribution of matching performance with the unevenly cropped videos having reduced precision than their evenly cropped counterparts, particularly for those tested with the Image Insertion technique. Also, larger rotations and vertical flips cause poorer matching performance than smaller rotations and horizontal flips respectively. The tests with the bilinear interpolation are grouped closer to higher true positive values than the tests with the Image Insertion method.

7.4.3 ResNet-50

This section explores the use of the pre-trained PyTorch ResNet-50 neural network.

Figure 7.9 shows the results for the ResNet-50 network using the same reference database with standard distortions as for the ResNet-18 results as shown in Figure 7.7. The results in the left-hand graph use the image insertion technique while the graph on the right uses bilinear interpolation.

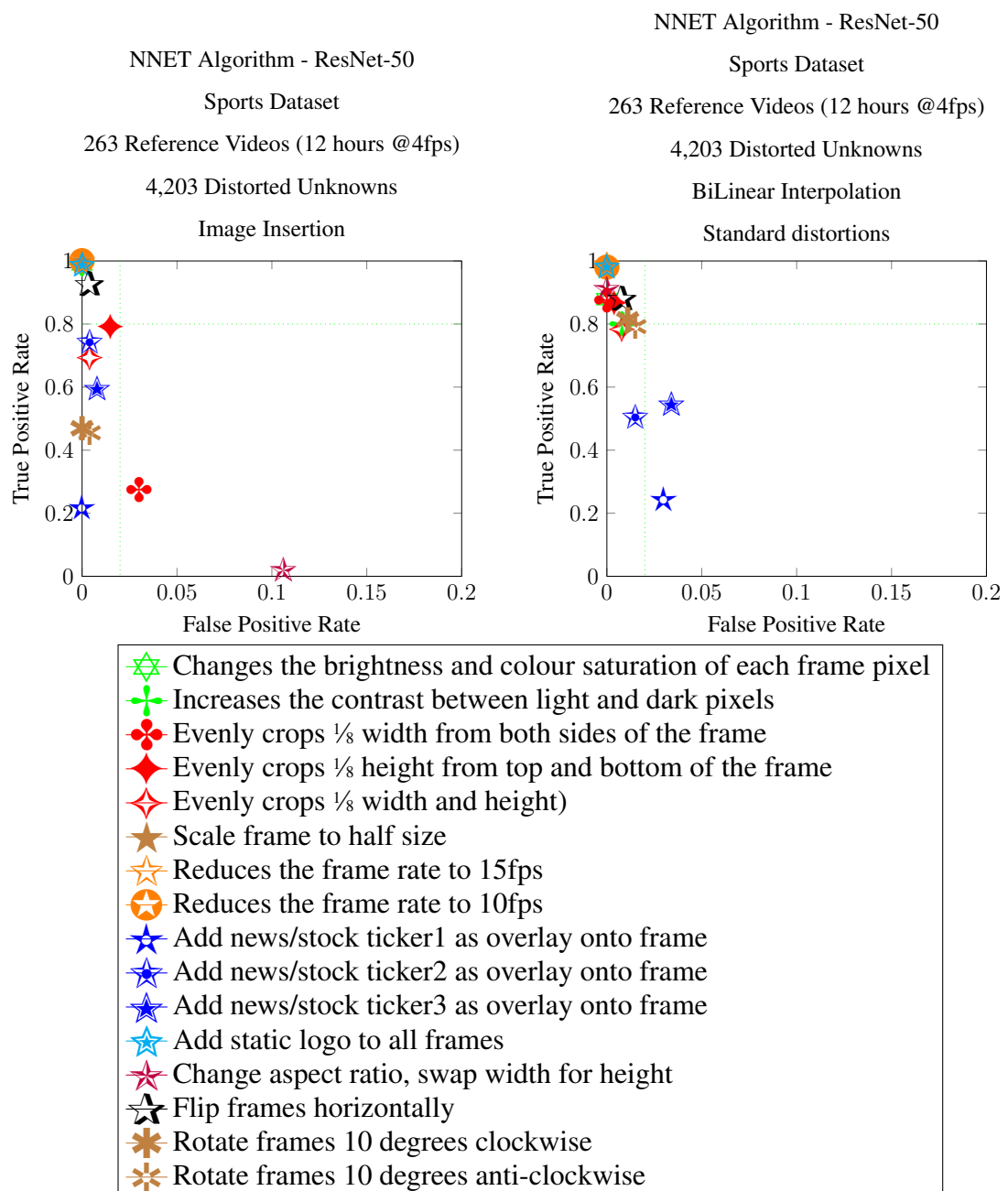


Figure 7.9: Pre-Trained neural network ResNet-50, simulated unknowns, standard distortions match performance

As with the ResNet-18 trials, virtually all the results using the bilinear interpolation (right hand graph) have a higher true positive rate than those for the image insertion technique. The two exceptions are the distortions that include news/stock ticker - however since all the other distortions are grouped into the top left of the graph, the bilinear interpolation is clearly preferred. For these standard distortions the results appear to be similar to those for ResNet-18 and since the ResNet-18 model is significantly smaller, requires less computation and produces a smaller vector - the ResNet-18 model is the preferred model.

Figure 7.10 shows the matching results for complex distortions with the same ResNet-50 neural network.

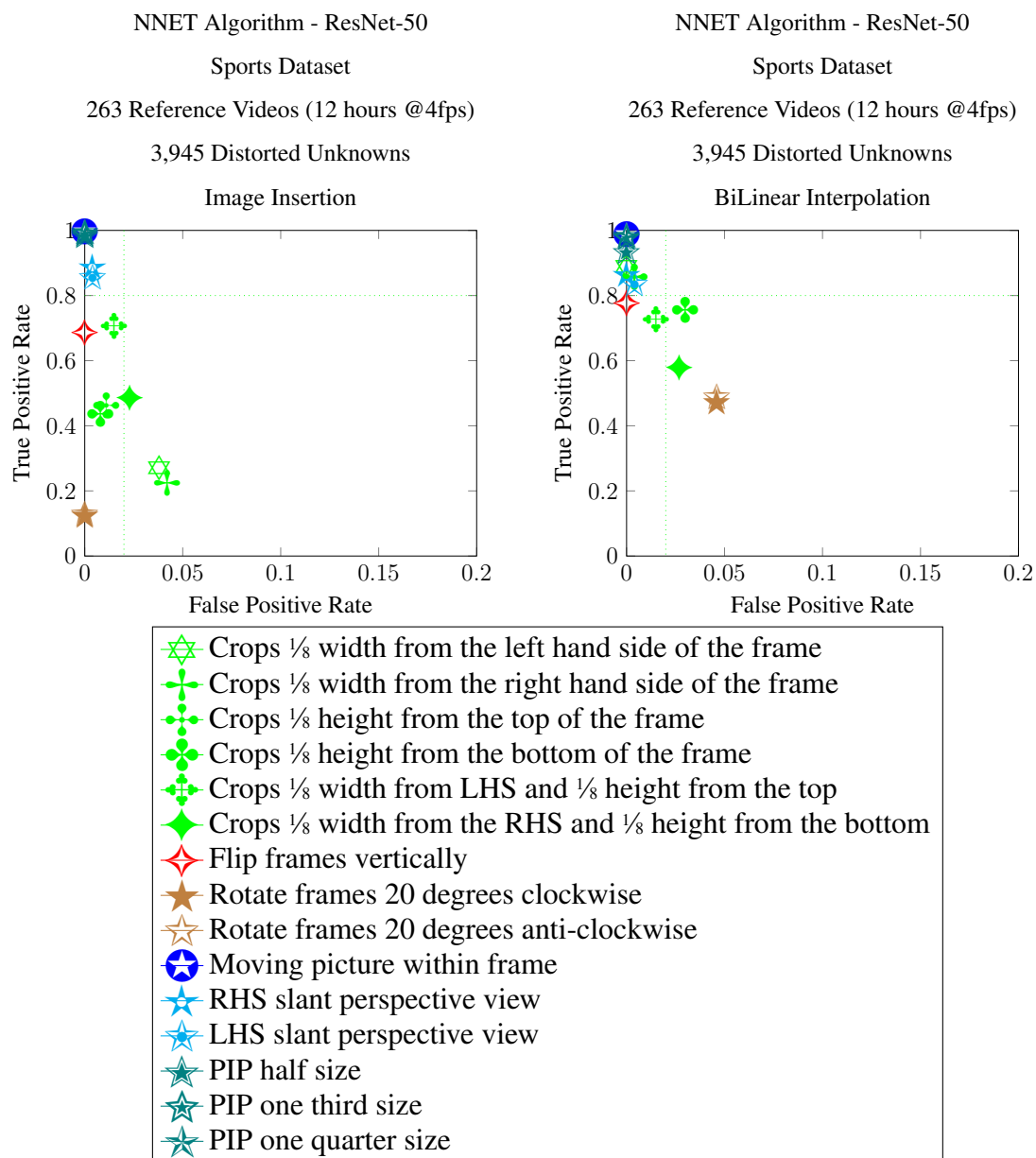


Figure 7.10: Pre-Trained neural network ResNet-50, simulated unknowns, complex distortions match performance

The results for complex distortions shown in Figure 7.10 show a greater distribution of matching performance with the unevenly cropped videos having reduced precision than their evenly cropped counterparts, particularly for those tested with the Image Insertion technique. Also, larger rotations and vertical flips cause poorer matching performance than smaller rotations and horizontal flips respectively. The tests with the bilinear interpolation are grouped closer to higher true positive values than the tests with the Image Insertion method.

7.4.4 Using a Variety of Pre-Trained Networks

Using the same methodology as for the ResNet-18 and ResNet-50 networks presented earlier in this chapter, this section analyses a wider range of pre-trained networks. The Figures 7.11 show the overall true positive rate for all distortions on each neural network against the output feature vector length provided by the network and separately the number of parameters in the neural network. The same sports database is used for each network test and the summarised results for ResNet-18 and ResNet-50 are included in the graph. Both bilinear and image insertion methods are shown, although bilinear always produces better results than image insertion. All distortion types (both standard and complex) are included in the true positive rate with the match threshold set at a point where 1% false positives are produced.

Both the output feature vector length and the number of network parameters are important values for the performance of the video match system. The output feature vector is directly used by the comparison process to determine the match score between frames - thus for performance reasons lower values of this parameter are preferred. The number of parameters that the neural network uses is also a performance factor in the matching process and once again a smaller value is preferred.

The results are shown in Table 7.2 and in Figure 7.11 in graphical form.

Model	True Positive Rate		
	Image Insertion	BiLinear	BiCubic
ResNet-10	-	0.94	-
ResNet-18	0.80	0.88	0.85
ResNet-34	-	0.90	-
ResNet-50	0.69	0.82	-
DenseNet-121	0.35	0.74	-
EfficientNet	0.42	-	-
MobilenNet	0.47	-	-
SqueezeNet	-	0.81	-

Table 7.2: Pre-Trained neural network models, summary results for sports dataset

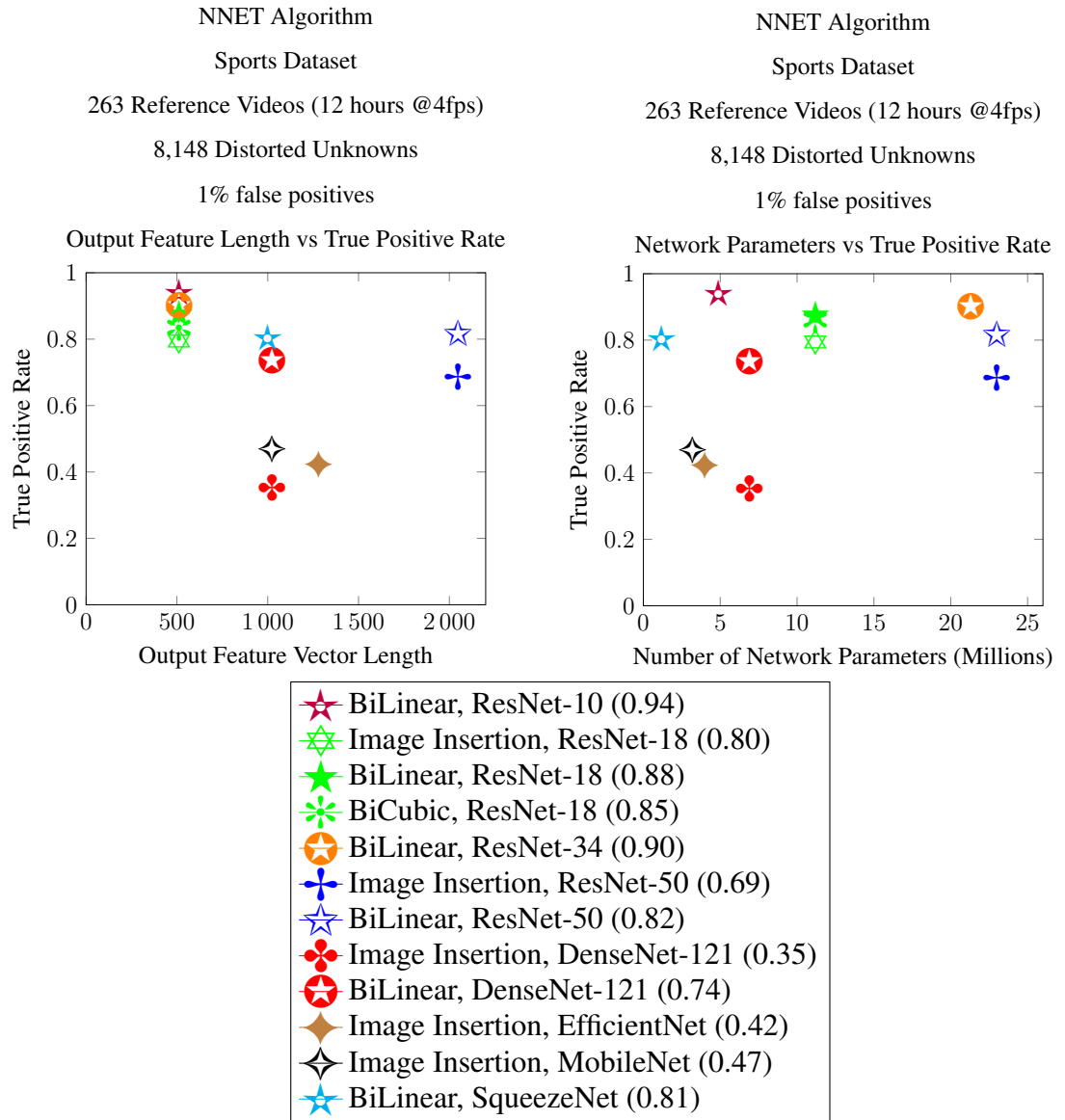


Figure 7.11: Pre-Trained neural network models, summary results for sports dataset

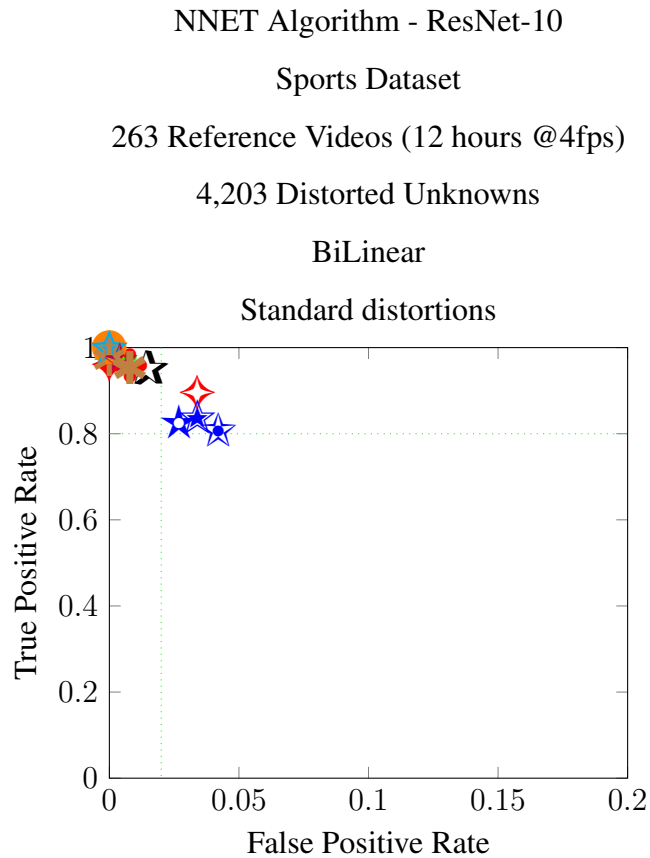
Figure 7.11 shows that the pre-trained ResNet-10 network provides the best overall true positive rate while using the lowest output feature vector length and close to the smallest number of parameters. The ResNet-10 network is the preferred network for video matching. This result appears slightly surprising as one of the smallest networks produces the best results. This may be because the larger networks capture more of the detail of the distortions and hence produce a significantly different feature vector from the frames in the reference video. Where as smaller networks capture less of the detail and therefore provide better generalisation than the larger networks. This issue appears to be

particularly acute with the overlaid ticker distortions.

Closer analysis of the results for SqueezeNet shown in Figures A.11 and A.12 and for DenseNet shown in Figures A.5 and A.6 show that the news ticker and 20° rotation distortions have a significant impact on their overall true positive score. However, removing these outliers would not challenge the conclusion that ResNet-10 produces the best overall true positive rate for this dataset.

It should also be noted that the target feature length per frame (aka GPU memory per second of video) is less than 500 bytes, as described in the overall study objectives described in Section 1.4. An output feature vector of 512 floating point numbers (32 bits) is 2,048 bytes and at 4 fps this equates to 8,192 bytes per second of video. This is 16 times the target value meaning that vectors at this size will not fit into GPU memory. Methods to reduce the feature size per second of video include; significantly reducing the frame rate for the reference feature vectors; finding a neural network with a smaller output feature vector; or training a network with a smaller output feature vector. However, the size of the feature vector is a significant issue when using pre-trained neural networks for this study.

The detailed graphs for the preferred pre-trained neural network (ResNet-10) are shown in Figures 7.12 and 7.13.



- ✱ Changes the brightness and colour saturation of each frame pixel (0.00, 0.99)
- ✱ Increases the contrast between light and dark pixels (0.00, 0.98)
- ✱ Evenly crops $\frac{1}{8}$ width from both sides of the frame (0.01, 0.96)
- ✱ Evenly crops $\frac{1}{8}$ height from top and bottom of the frame (0.00, 0.96)
- ✱ Evenly crops $\frac{1}{8}$ width and height (0.03, 0.90)
- ★ Scale frame to half size (0.00, 1.00)
- ★ Reduces the frame rate to 15fps (0.00, 1.00)
- ★ Reduces the frame rate to 10fps (0.00, 1.00)
- ★ Add news/stock ticker1 as overlay onto frame (0.03, 0.83)
- ★ Add news/stock ticker2 as overlay onto frame (0.04, 0.81)
- ★ Add news/stock ticker3 as overlay onto frame (0.03, 0.84)
- ★ Add static logo to all frames (0.00, 1.0)
- ✱ Change aspect ratio, swap width for height (0.00, 0.97)
- ★ Flip frames horizontally (0.02, 0.95)
- ✱ Rotate frames 10 degrees clockwise (0.01, 0.95)
- ✱ Rotate frames 10 degrees anti-clockwise (0.00, 0.97)

Figure 7.12: Pre-Trained neural network ResNet-10, simulated unknowns, standard distortions match performance

The ResNet-10 results using standard distortions in Figure 7.12 shows the relatively tight grouping of all the distortion types into the top right corner of the graph. The poorest

results are for the news tickers that show a true positive rate around 80% and a false positive rate of around 3% to 4

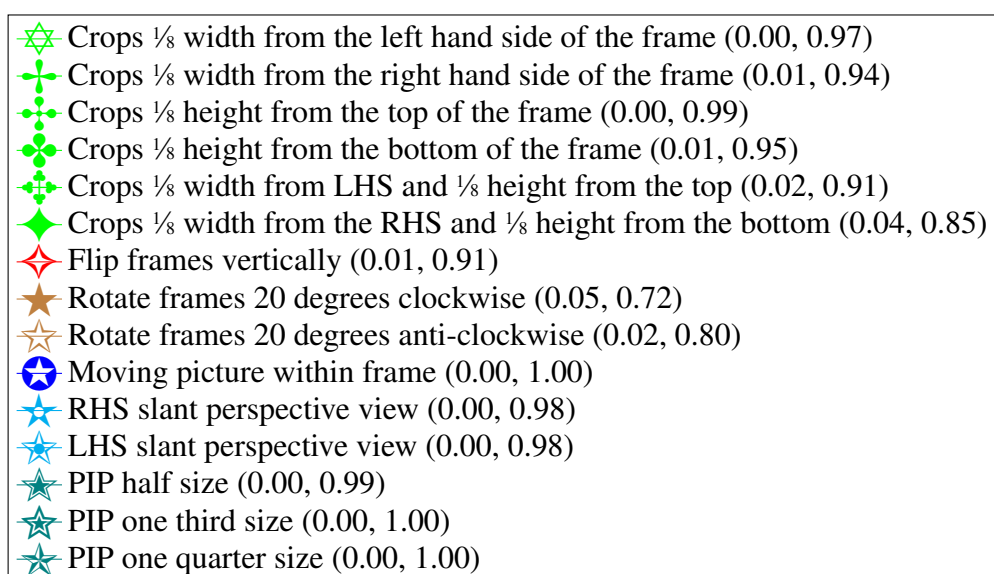
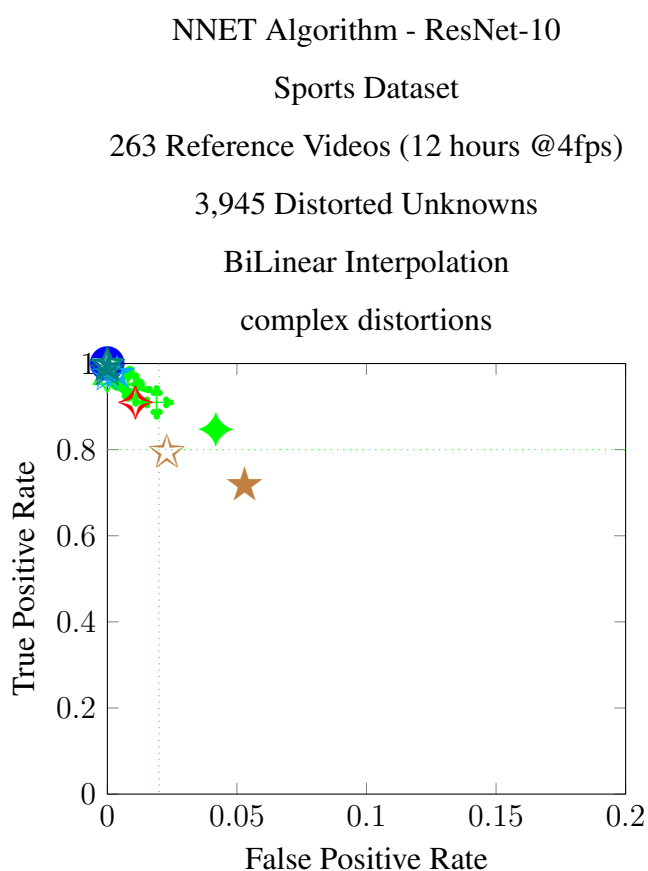


Figure 7.13: Pre-Trained neural network ResNet-10, simulated unknowns, complex distortions match performance

The ResNet-10 results using complex distortions in Figure 7.13 shows that the 20° rotations are poor performers along with the more severe cropping distortions.

7.4.5 Multiple Frames per Vector

The multiple frames per vector techniques insert multiple video frames into one image which is then used to generate a feature vector with the pre-trained neural network. For this experiment ResNet-10 was used with the sports database generating the results shown in Table 7.3.

Pre-trained Network	Technique	True Positive Rate		
		Normal Distortions	Complex Distortions	All Distortions
ResNet-10	Color Planes (Consecutive Frames @4fps)	0.701	0.536	0.613
ResNet-10	Color Planes (Consecutive Frames @1fps)	0.701	0.539	0.615
ResNet-10	Tiled (2×3)	0.800	0.805	0.803

Table 7.3: Multiple frames per vector results

These results show that these multiple frame per vector techniques do not improve the match performance over the standard bilinear interpolation of a single frame. Therefore since the standard bilinear interpolation is a simpler method that is the preferred technique.

7.4.6 Matching with the Feature Film Dataset

So far in this chapter the sports dataset has been used to test the quality of matching with predefined neural networks. The sports database uses simulated unknowns and does not contain any noise items (i.e. items that are not derived from one of the sports videos). In this section the feature film dataset will use the same pre-defined neural networks to determine whether the trends obtained from the sports dataset are reproduced with a more realistic dataset.

Figure 7.14 shows the results of using the feature film dataset with the same set of pre-defined neural networks as for the sports dataset in Figures 7.12 and 7.13 above. No noise items were added to the dataset in this initial test. With no noise items a high matching accuracy is expected as there is a significantly lower probability that feature vectors will match the wrong item. The feature film dataset is described in detail in Appendix D.

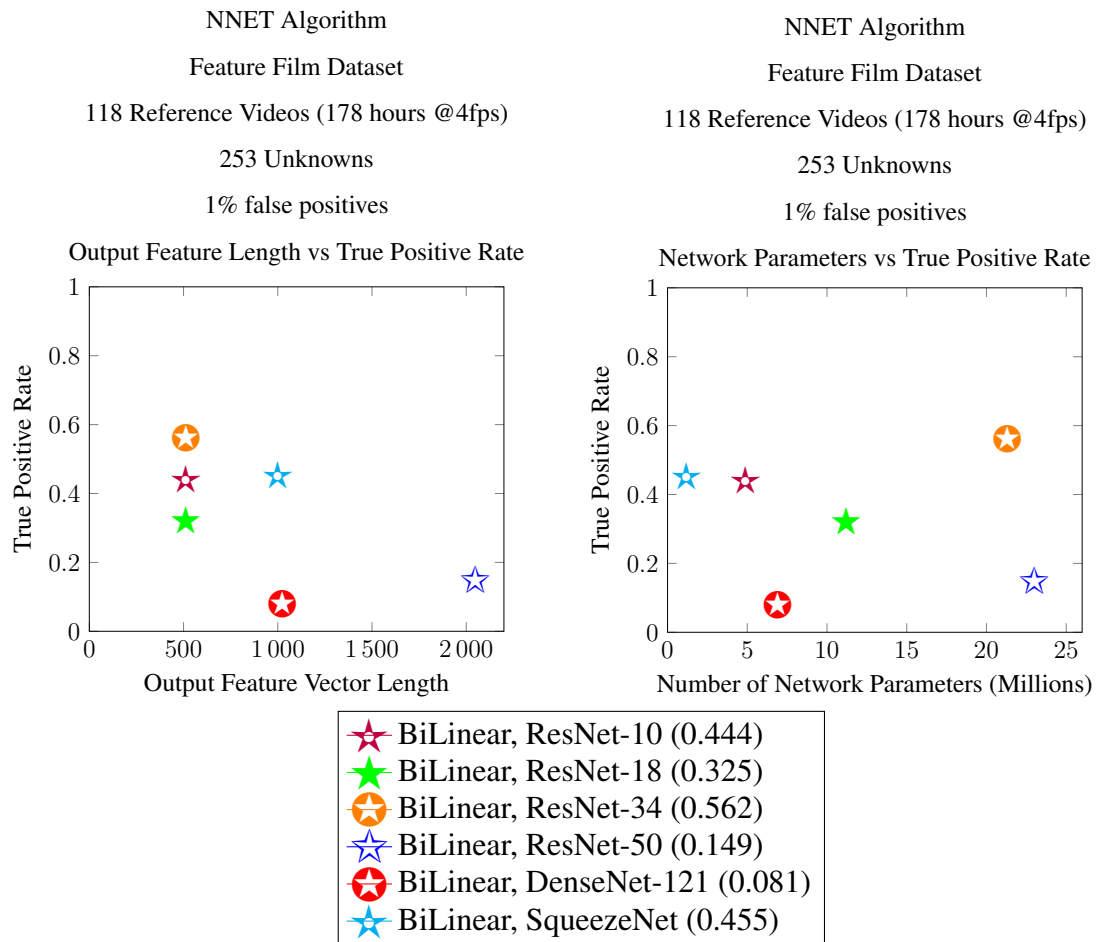


Figure 7.14: Pre-Trained neural networks, feature film dataset matching performance evaluation

Figure 7.14 shows the real-world feature film dataset produces a significantly lower proportion of true positives compared with the simulated sports dataset above. ResNet-34 produces the best result around 10% better than the second place network SqueezeNet.

The low match rate (without noise items) coupled with the size of the feature vector means that the use of pre-defined neural networks is inconsistent with the objectives of this study.

For completeness, the same experiment using the reference videos at 1fps rather than

4fps is shown in Figure 7.15. As expected the true positive rate decreases further, confirming the lack of accuracy obtained using these pre-trained neural networks.

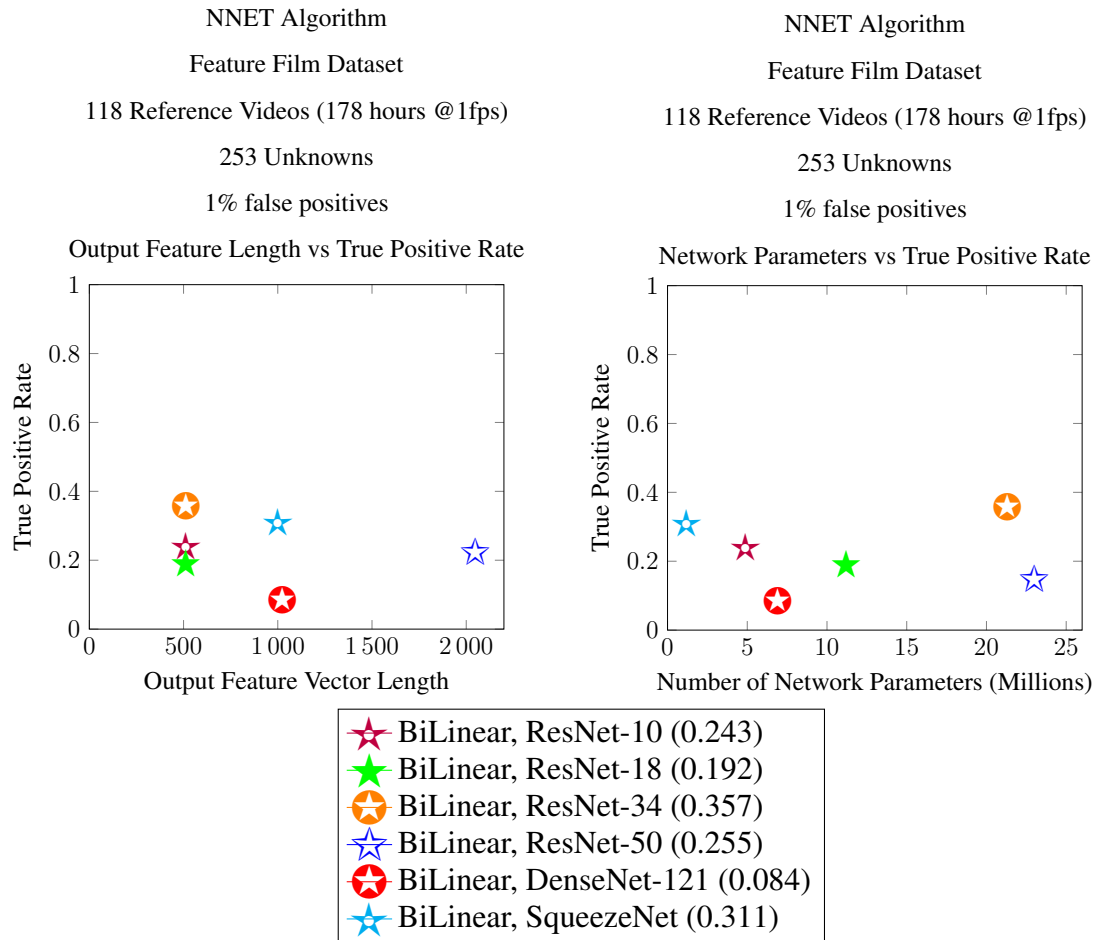


Figure 7.15: Pre-trained neural networks, simulated unknowns, matching performance at 1fps evaluation

7.5 Summary

In this chapter, video frame matching using a number of pre-trained neural networks was explored. Using a synthetically created sports-based video database it was found that the ResNet-18 network could achieve a match rate of 94.3% on a closed set of distorted unknown videos. However, the feature vector used for matching was relatively large (8k bytes per second of video) far exceeding the goal of less than 500 bytes per second of video described in Table 1.4.

A further experiment was performed with the feature film database, used in this study, which contains videos from two separate sources. In this test the ResNet-34 network produced the best results with a match rate of 56.2%. This result is considerably below the requirement of this study to achieve a 90% match rate.

Further matching techniques were explored using multiple frames per vector but the match performance of these was not better than for the single frame per vector methods.

Overall pre-trained neural networks did not achieve the match accuracy, the bytes per frame target or the performance characteristics to meet the goals of this study.

Training Convolutional Neural Networks with Triplet Loss

Convolutional neural networks such as the ResNet family [87] (ResNet-10, ResNet-18, ResNet34 etc.) can be trained using sets of triples or triplets (an anchor, a positive and a negative image) [89]. These sets of triplets are used to train a neural network through a process called triplet loss. During training, the triplet loss algorithm attempts to minimise the distance between the anchor and positive image while maximising the distance between the anchor and negative image.

In this chapter the use of a triplet loss training method will be used to train the classification layer of certain pre-trained neural networks. The benefit of this approach is that the original training of the network is supplemented by training with comparatively little data [90]. Once the neural network is trained, feature vectors will be extracted for each frame in the reference video database along with the unknown video clips. These feature vectors will be compared to obtain match candidates for the unknown video clip which will be further analysed to determine whether a match has been obtained.

8.1 Reference Video Dataset

The reference video signatures held by Audible Magic contain a degraded form of the original video along with descriptive and technical metadata. These degraded videos contain no sound, have a frame size of 128×72 pixels, use 8 bit greyscale pixels at a frame rate of 4 frame per second. One hundred and twenty of these reference video signatures were selected that contained feature films that were released in the last 10 years. This selection of reference video signatures forms the reference dataset for the experiments in this chapter.

The videos to be matched against this reference dataset are film clips distributed by the content owners, these clips are generally of between 2 and 10 minutes duration and are called the unknowns. 253 film clips (unknowns) were obtained with at least one clip from each reference film.

While the unknown clips are excerpts of the original full-length film, the visual content is generally different. These differences between the unknown and reference videos include cropping and letter-boxing (to adjust the clip to be viewed on a computer device), frame rate differences, colour model and balance differences, overlays on the unknowns (logos to identify the source) and trailer video clips of about 30 seconds that describe the distributor of the film and other features. At least one of the unknowns contains a scene of over 20 seconds duration not included in the reference video held by Audible Magic. To achieve successful matching between the unknowns and references a matching algorithm that is robust to these types of distortions is required.

8.1.1 Blank Frames

Blank frames (frames of a uniform colour often just black) must be ignored for matching purposes as they could appear in any video, will match with other blank frames and are likely to cause false positives. The same blank frame detector is used for all the algorithms in this study. Frame sequences that include blank frames will not be used in the feature vector comparisons.

8.2 Creating Triples

To train and test the match algorithm, a set of match triples were constructed from the unknown and reference datasets. These match triples consist of an anchor image, which is a frame taken from the reference video, a positive match image, which is the corresponding frame taken from the unknown video clip that matches the reference, and a negative match image, taken from the unknown video clip one or more frames later (in time) than the positive frame. This negative match image is a hard negative as it is likely to be very similar to the positive image. Hard negatives are recommended for training [91] as the objective is to create a strong discriminator function between the anchor and the positive and negative images.

8.2.1 Alignment

In the construction of the match triples some difficulty was experienced in precisely aligning (in time) the reference and unknown video clips. Since the reference and unknown have been independently down-sampled to 4 fps, exact alignment of reference frames with the unknown frames was not possible. The difference in frame content is particularly acute for videos with fast changing action sequences. Two separate methods were used to minimise this issue.

In the first method, the Euclidean Distance was calculated between the greyscale pixels in the reference and unknown frames around the alignment point averaged for all frames in the video clip. The alignment point between the reference and unknown frames was adjusted until the average Euclidean distance was at a minimum. Only if the Euclidean distance for the aligned frames was below a certain threshold were the reference and unknown frames used as a basis for a match triple. The upside of this approach is that the anchor image (reference) and positive image (unknown) are very similar; however, the downside is that the approach may artificially ignore distortions which lead to a larger Euclidean distance and therefore potential triples are not used as part of the training set - making the training set less comprehensive.

In the second method, only the unknown is used to construct the triple. The anchor image was taken as a frame from the unknown, with the positive image a distorted version

of the anchor (randomly cropping, adding noise). This approach sidesteps the issue of alignment but relies on the distortion process to faithfully replicate the types of distortion found between reference and unknown to create a comprehensive training set.

The results of these different approaches to constructing triples for training and testing are shown in the results section of this chapter.

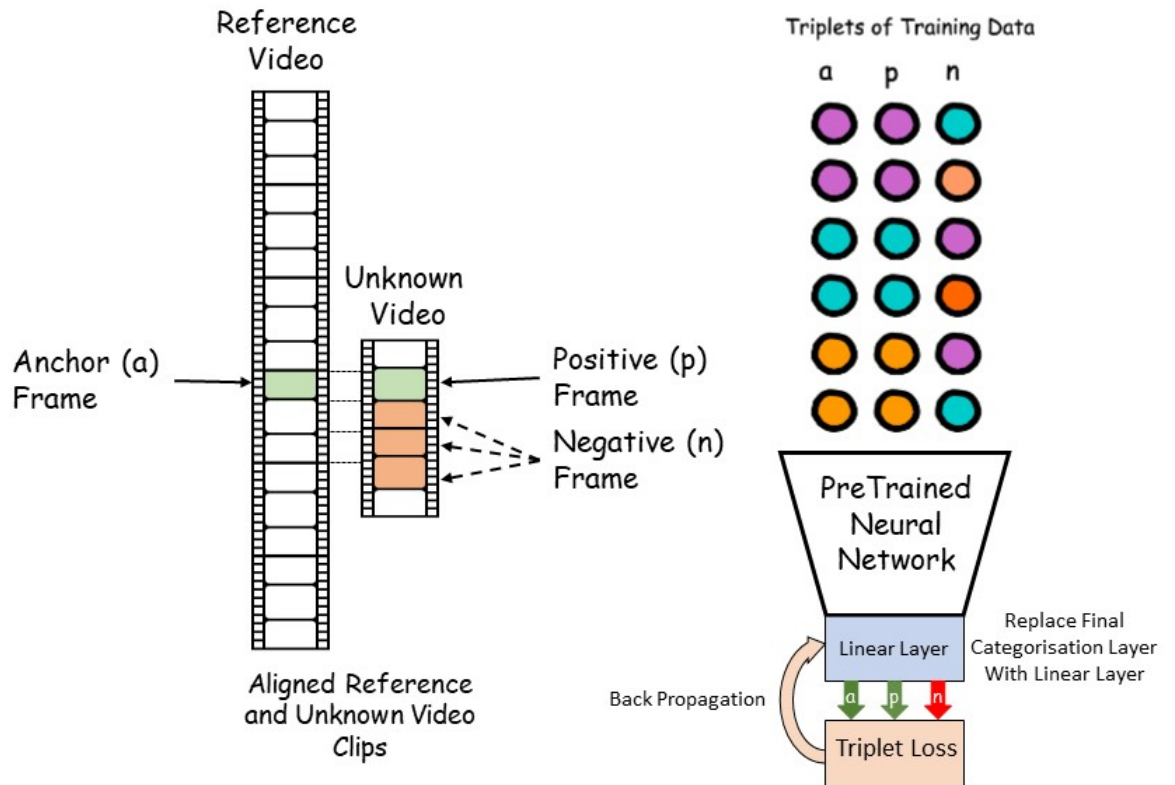


Figure 8.1: Neural network training and triplet construction

Figure 8.1 shows how the triplets are constructed and used in the training process. An anchor frame is selected from the reference video with the corresponding frame in the unknown used as the positive frame. A negative frame is selected from a different location in the unknown video. This combination of anchor, positive and negative frames (triplet) is used as the basis for training the final layer of the pre-trained neural network.

8.2.2 Training

The ResNet neural network architecture was used to obtain representative feature vectors for both the reference and unknown video frames. In all cases, a pre-trained ResNet neural network was used as the basis for training. The final classification layer of this pre-trained network was replaced with a fully connected linear layer that produced a 512 floating point feature vector. Some experimentation was performed on the size of this output feature vector. A larger output feature vector requires more computational resources at match time than a smaller vector but a larger vector presumably also provides more accuracy for the match process than a smaller vector. However, in matching a reference frame to a distorted frame from an unknown video this accuracy may not be beneficial as the training process might model the distortions. Therefore it is uncertain whether a larger or smaller vector will produce the best results. The choice of feature vector size is therefore explored in the results section of this study.

The training triplets were randomly feed into the ResNet network using the triplet loss algorithm with backpropagation to train the weights of the network.

A number of networks were trained with the training data set that consisted of 10,000 triples. The parameters that were explored in this training process were:

1. Neural Network Model: The type of pre-trained neural network used (default: ResNet-18)
2. Size Of Training Set: The number of triples in the training dataset (default: 10,000)
3. Output Vector Length: The number of floating point values in the output feature vector (default: 512)
4. Frame Lag For Negative: The number of frames in the unknown between the positive frame and negative frames (default: 2)
5. Batch Size: Batch size used in training the neural network (default: 128)
6. Triplet Loss Margin: The margin added to the triplet loss equation (default: 1.0)
7. Match Pair Score Threshold: Minimum value for the total pixel Euclidean distance between reference and unknown frames to be accepted for training (default: 150.0)

8. Epochs: Number of training epochs (default: 50)
9. Learning Rate: Learning rate of the neural network (default: 0.0001)
10. Norm Frame Width, Norm Frame Height: The normalised size (in pixels) of video frames (default: 128×72)
11. Normalisation: The normalised mean and standard deviation of the pixels in each frame (default: 0.0, 1.0)

8.3 Matching

The reference feature vectors were stored in an indexed database for high performance lookup. The FAISS [52] scalable search package was used for this process.

Feature vectors from 30 second excerpts of the unknown video clips were matched against the reference database to obtain candidate matches. Each frame of the unknown video was looked up in the reference feature database, the output of this process is a set of reference frames with the smallest Euclidean distance between the reference and unknown feature vectors - these reference and unknown frame pairs form the basis of candidate matches.

The candidate matches provide an alignment point from which 30 seconds of the reference and unknown feature vectors are compared. The average Euclidean distance between the reference and unknown features vectors for the frames in the 30 seconds excerpt are computed. This average Euclidean distance is empirically thresholded to determine if a match has been achieved, i.e. If the average distance is below the empirically determined threshold value - a match is declared.

To obtain matches from this process the following assumptions are made:

1. The reference and unknown video clips contain the same frames (i.e. no frames are inserted or deleted from the unknown video clip).
2. The frames between the reference and unknown remain aligned for the 30 second interval.

3. The content in the frames of the reference and unknown are largely the same (probably incorrect for some types of distortion including cropping, overlays such as tickers or logos)

If these assumptions are violated matching would not be possible by these methods. Frame insertion or deletion precludes the use of matching based on a continuous set of sequential frames that these techniques rely on. Frame alignment is similar to frame insertion as the reference and unknown frames will not align in time and hence the assumed sequential matching between reference and unknown will not occur. Clearly, the frame content must be similar between reference and unknown as the matching techniques depend on frame content. In the vast majority of cases these assumptions are only weakly violated making matching possible.

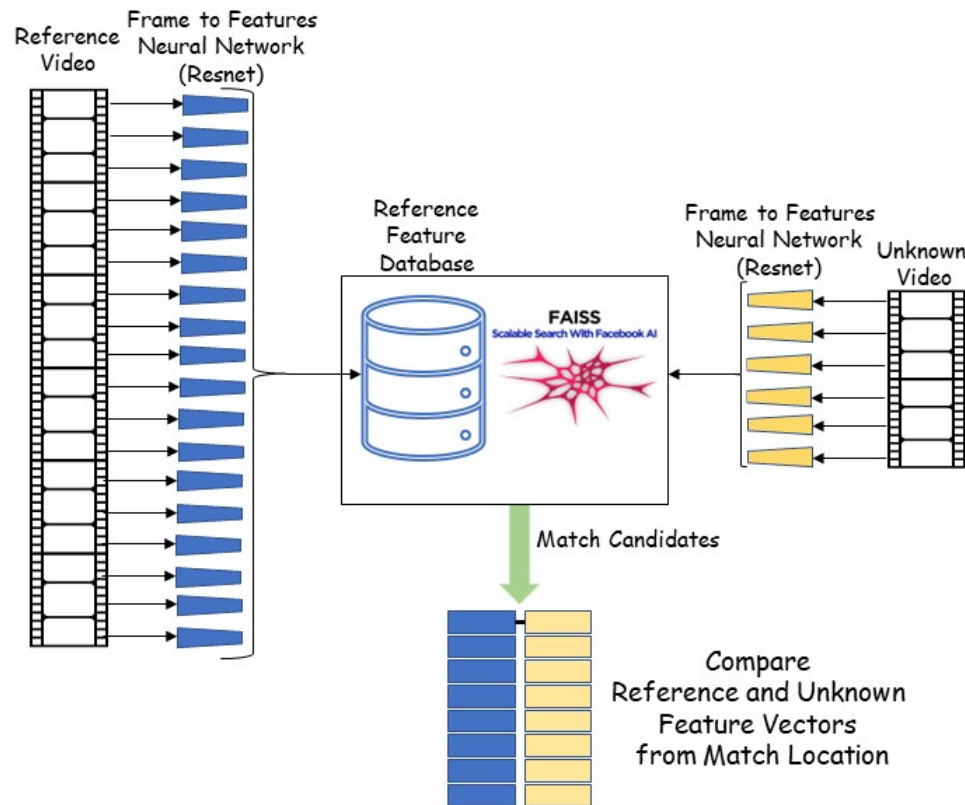


Figure 8.2: Neural network feature computation and matching with reference database using Faiss index

Figure 8.2 shows the matching process used to compare the reference and unknown feature vectors. A feature vector is computed for each reference video frame by passing the frame through the trained ResNet model. These feature vectors are stored in the Reference Feature Database. At match time feature vectors are computed for the unknown video clip. The feature vectors for the unknown are compared with the reference feature database to produce match candidates. Once a match candidate has been identified, a comparison of the reference and unknown feature vectors is made for the match duration.

8.3.1 Matching Statistics

The exact match points between the references and unknowns were computed. These match points were constructed from a combination of metadata and previous match algorithms along with a final visual verification using a purpose-built Windows standalone application. This alignment data was used to determine whether a correct match was made between the reference and unknown videos.

In this analysis, each match transaction can have three possible outcomes. These outcomes are:

1. True Positive: The reference and unknown match close to the alignment point.
2. False Negative: The unknown does not match any references.
3. False Positive: The unknown matches the wrong reference or the correct reference at the wrong time.

Since all unknowns have a corresponding reference in the dataset, true negatives are not possible in the match process.

8.4 Results

The graphs in Appendix B.1a of this chapter show the results of testing the trained ResNet-18 neural network with a range of parameters. In each case a pre-trained ResNet network was first trained with the training set and then tested with a separate test dataset. Figure 8.3 shows a typical result of this experimentation.

Distortion: Cropped

program: resnet18train3.py, frameLagForNegative: 1000

outputVectorLength: 512, checkpointFile: s18_1.pt

batchSize: 64, tripletLossMargin: 1.0, matchPairScoreThreshold: 150.0

epochs: 50, learningRate: 0.0001, newMean: 0.0

newStd: 1.0, useUnknownForAnchor: False

Saved: s18_1.pt loss: 0.032857694 batch size: 64

threshold score: 1.308, true positive Rate: 0.39

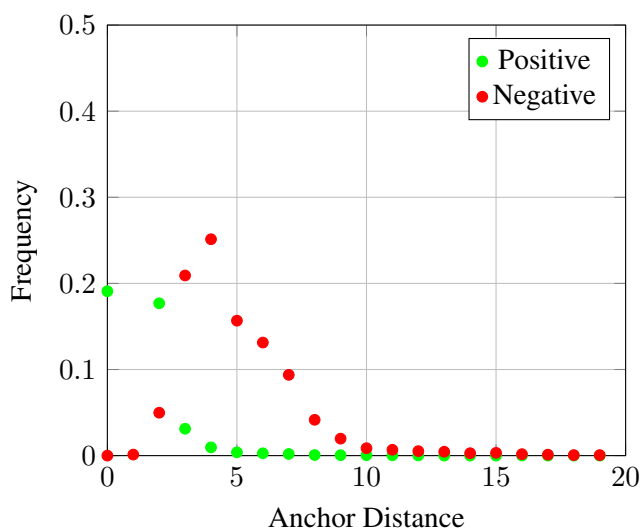


Figure 8.3: ResNet-18 neural network with a trained final layer, match performance using triples with a frame lag to the negative component of 1,000 frames

In all these histograms (example Figure 8.3), the y-axis shows the proportion (frequency) of the test triples that produced the anchor distance shown on the x-axis. The anchor distance is computed between the anchor and positive feature vector (green) and the anchor and the negative feature vector (red). The most desirable outcome is a separation between the positive and negative curves which would imply that the anchor to positive distance is smaller than the anchor to negative distance.

The key above each graph describes the parameter values used to perform the test.

The graphs in Appendix B.1a are grouped into two sets. The first set of Figures (B.1a to B.7b) show the results when the negative frame is 1000 frames away from the positive frame. The second set of Figures B.8a to B.14b show results where the negative frame is 4 frames or fewer from the positive frame.

In the first set of Figures B.1a to B.7b the highest true positive rate is 0.92 produced by Figure 8.4. This graph shows a fairly good separation for the anchor distance between the positive and negative items. The parameters used for this test are a batch size of 128, triplet loss margin of 1.0, a match pair score threshold of 150.0, using 50 epochs of training with a learning rate of 0.0001.

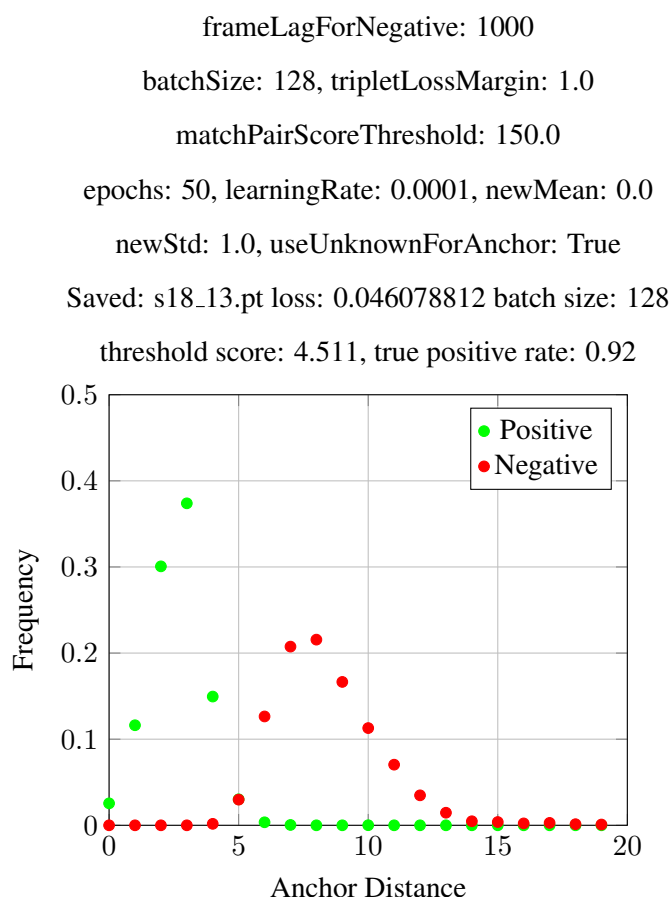


Figure 8.4: ResNet-18 neural network with a trained final layer, match performance using triples, example 13

In the second set of Figures B.8a to B.14b where the negative frame was 4 or fewer frames from the positive frame very low true positive rates were obtained. The positive and negative distance curves are almost superimposed on each other meaning that the neural network has produced a very similar vector for both the positive and negative frames.

As a result of these experiments, the parameter set that consistently produced the best discrimination between the positive and negative images is shown in Table 8.1. A ResNet-

50 model was also used in these tests but the results were poorer than those that used the ResNet-18 model.

Parameter	Value
model	ResNet-18
outputVectorLength	512
frameLagForNegative	1000
batchSize	128
tripletLossMargin	1.0
matchPairScoreThreshold	150.0
epochs	50
learningRate	0.0001
normFrameWidth, normFrameHeight	128×72
normMean, normStd	0.0, 1.0

Table 8.1: ResNet-18 trained neural network, best discriminator model parameters

8.4.1 Video Matching Experiments

The results described so far in this chapter have focused on matching individual frames from reference and unknown videos. The best trained model described in Table 8.1 was then implemented in the production AudibleMagic video matching code using the C++ library that PyTorch provides for this purpose.

The results for the ResNet-18 and ResNet-50 neural network are shown in Figures 8.5 and 8.6 respectively. These graphs show the true and false positive histograms for the trained networks. As seen in these graphs, the ResNet-18 network produced more discrimination between the positives and the negatives than the ResNet-50 network. However, with the ResNet-18 network an anchor to positive distance of 0.02 corresponds to a false positive rate of 2.3% and a true positive rate of 30% considerably lower than the target of 90% described in the objectives of this study in Section 1.4.

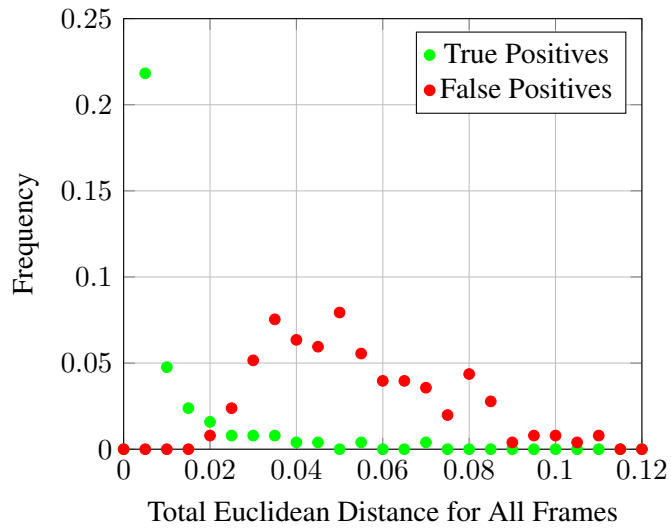


Figure 8.5: Video matching using the ResNet-18 neural network

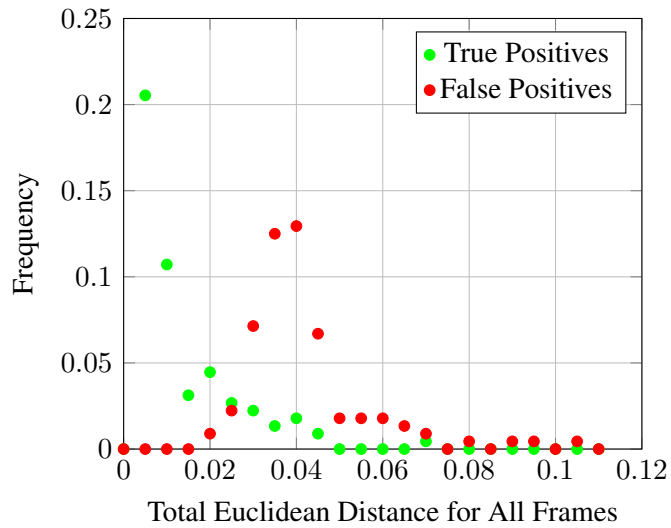


Figure 8.6: ResNet-50 trained neural network, match performance

A further set of experiments was performed reducing the size of the reference and unknown video frame sizes to 32×18 shown in Figure 8.7 and 16×9 shown in Figure 8.8. The training was performed on a ResNet-18 neural network and then implemented in the production software. The result for the frame size of 32×18 showed a better separation between true positives and false positives with 70% of true positives having a smaller total euclidean distance than 4% of the false positives. The graph utilising the

frame size of 16×9 did not sufficiently discriminate between the positives and negative items.

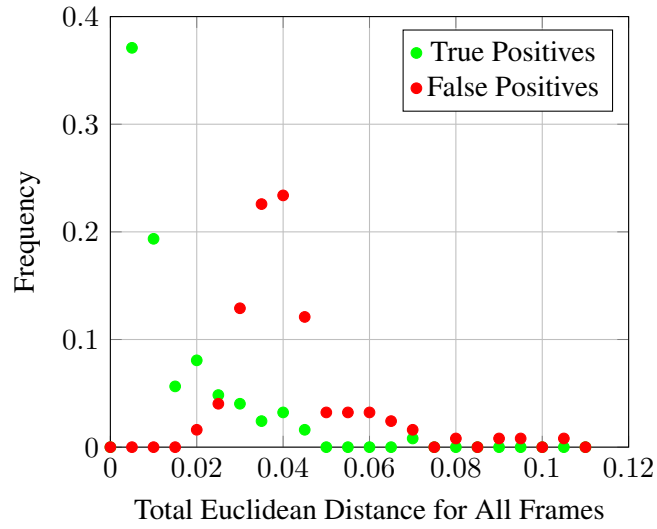


Figure 8.7: ResNet-18 trained network using frame size of 32×18 , match performance

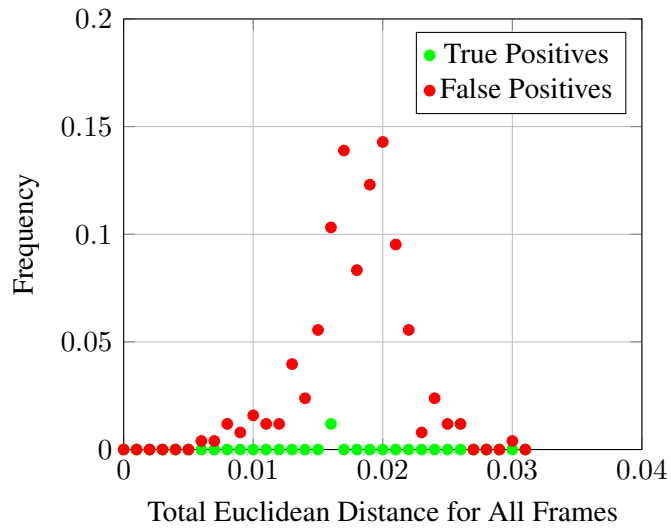


Figure 8.8: ResNet-18 trained network using frame size of 16×9 , match performance

8.5 Summary

A pre-trained ResNet-18 neural network was trained with video frames (data triplets) obtained from Audible Magic reference collection and corresponding feature film clips from the internet. The training parameters for the neural network were varied to obtain the optimal neural network for discriminating between positive and negative video frames.

The optimal network was then implemented in Audible Magic production software to match reference and unknown videos segments. The minimum acceptable standard for this video match test is to obtain less than a false positive rate of less than 2% with a true positive rate greater than 80% as described in the objectives of this study in Section 1.4. The trained ResNet-18 network only yielded a true positives rate of 30% for a false positive rate of 2.3%.

However, when the frame size was reduced from 128×72 to 32×18 the true positive rate increased to 70% for a false positive rate of 4%. Unfortunately, these measures are significantly below the minimum standard required for the matching process in this study described in Section 1.4.

CHAPTER 9

Hybrid Approaches

As described in the preceding chapters each algorithm presented in this study has strengths and weaknesses. However, none of the algorithms alone has achieved the overall goals of this study as originally set out in Table 1.4.

The GRID algorithm in Section 4.10 is too sensitive to the assumption that the bounding box of the reference and unknown frames are the same. Spatial points of interest in Section 5.4 are not resilient to small amounts of noise. The PIXL algorithm in Section 6.8 is sensitive to frame sequence and timing changes within the match region and generated too many false positives. Pre-trained neural networks in Section 7.5 did not produce a high enough true positive rate even for a small synthetically created dataset. The trained ResNet-18 neural network in Section 8.5 also did not have a sufficiently high true positive rate and its feature vector is too large to accommodate in GPU memory.

A potential method to achieve these objectives is to combine the most successful of these algorithms. The two algorithms that have produced the best results on a real-world dataset are the GRID algorithm in Section 4.10 and the PIXL algorithm in Section 6.8. The GRID algorithm achieved a 70% true positive rate along with a false positive rate of 1.2% on the feature film database, while the PIXL algorithm achieved a 70% true positive rate with a false positive rate of 3.0% with the same database. These tests included 20,000

noise videos that helped create a real-world experiment where a large number of videos are not expected to match the unknown.

Since the match score for each algorithm is normalised by its match threshold score (above this match score a match is declared) - each algorithm can separately compute a normalised match score (by dividing the raw score by the match threshold score) and the highest score from either algorithm can be used as the overall match score - above one will be a match.

9.1 Combined Algorithm with Feature Film Dataset

Since the PIXL algorithm has a higher false positive rate (3.0%) than the GRID algorithm - the various parameters of the PIXL algorithm can be adjusted to reduce this false positive score so that when combined with the GRID algorithm a higher overall true positive rate is achieved while maintaining a low overall false positive score. The simplest approach to reducing the number of false positives from the PIXL algorithm is to increase the match score threshold score. The match score threshold is then used to normalise the match score for each algorithm, as follows

$$MS_{norm} = \frac{MS_{raw}}{MST_{algorithm}} \quad (9.1)$$

where MS_{norm} and MS_{raw} are the normalised and raw match scores for a specific algorithm, $MST_{algorithm}$ is the match score threshold for the same algorithm. If either of the normalised match scores is above 1.0 a match is declared.

Figure 9.1 shows the effect of increasing the PIXL match threshold score on the combined match rate.

Combined GRID and PIXL Algorithm

Feature Film Database

Using 20,000 production reference video signatures (noise videos)

Representing 7,750 hours of video

PIXL match score threshold versus combined match rate

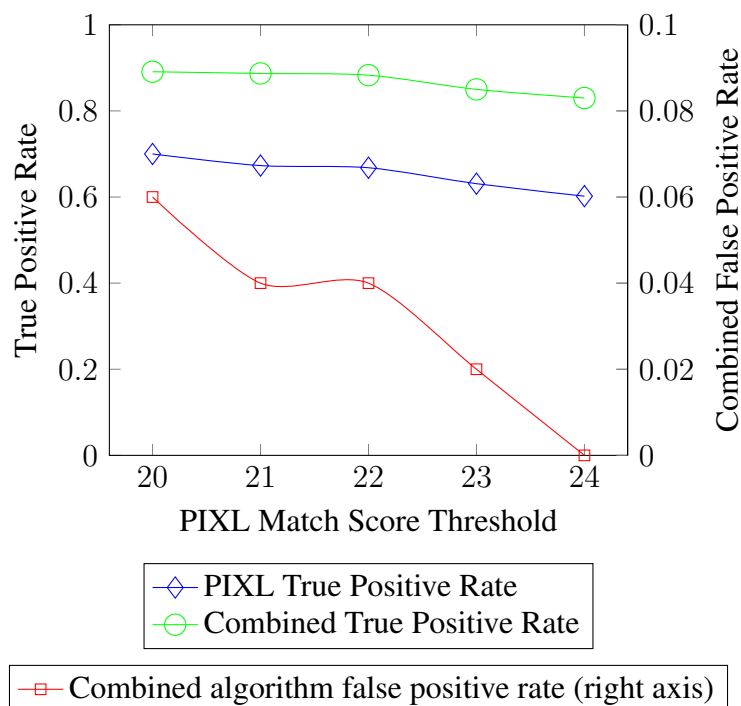


Figure 9.1: PIXL match score threshold versus combined match rate

The PIXL match score threshold shown in Figure 9.1 shows that increasing the match threshold score of the PIXL algorithm from 20.0 to 24.0 reduces the false positive rate for the combined algorithm. This simultaneously reduces the combined algorithm true positive rate from 0.89 to 0.83. A true positive score above 0.8 with a false positive rate below 0.02 put this combined algorithm into the range of the minimum acceptable standard originally defined in Table 1.4 at the beginning of this study.

9.2 Combined Algorithm with Sports Dataset

Using a PIXL match score threshold value of 23.0 with the default GRID match score threshold of 21.0, a combined algorithm match (PIXL & GRID) was performed with the sports database (including the same 20,000 noise videos). This experiment produced the

match statistics shown in Figures 9.2 and 9.3.

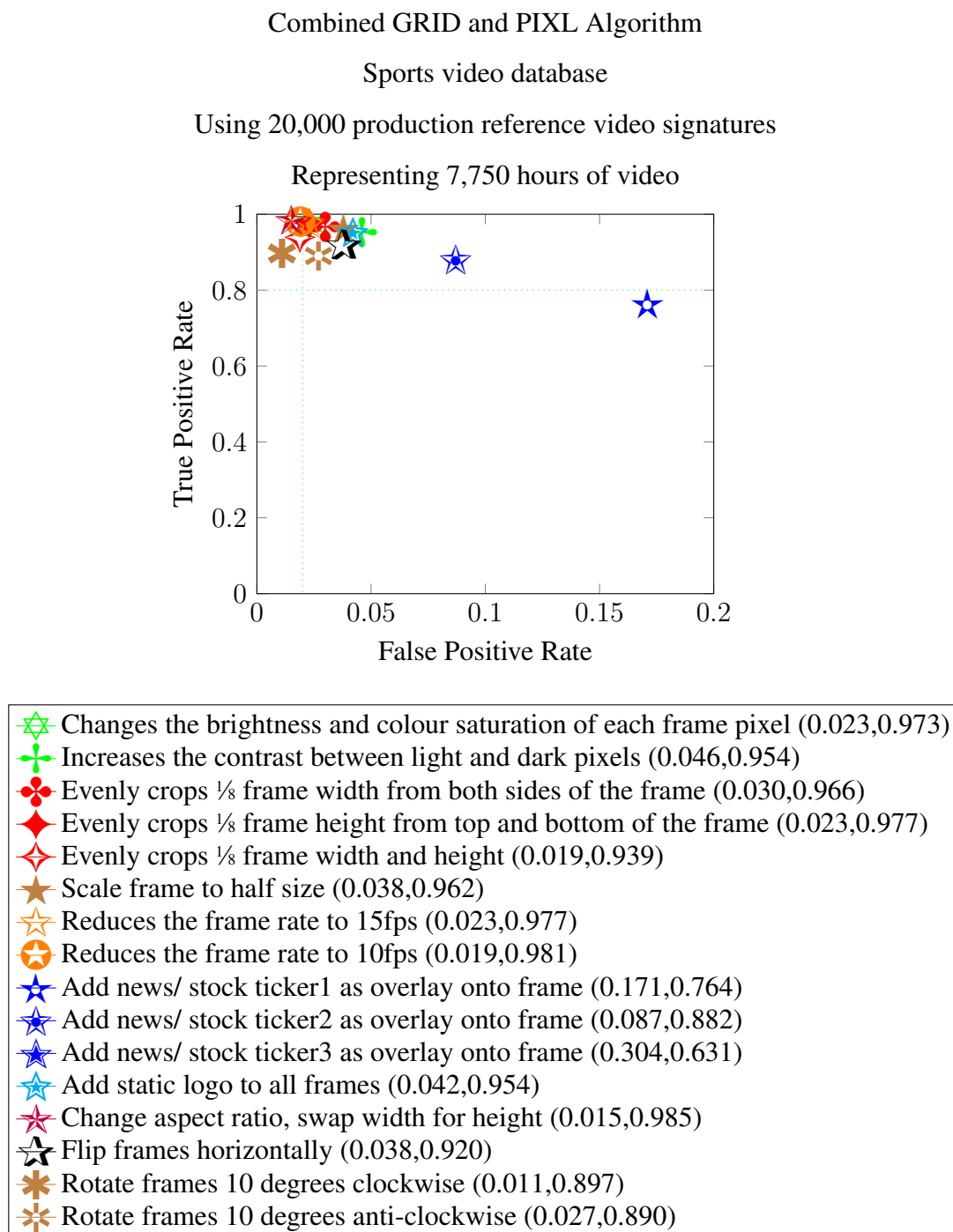
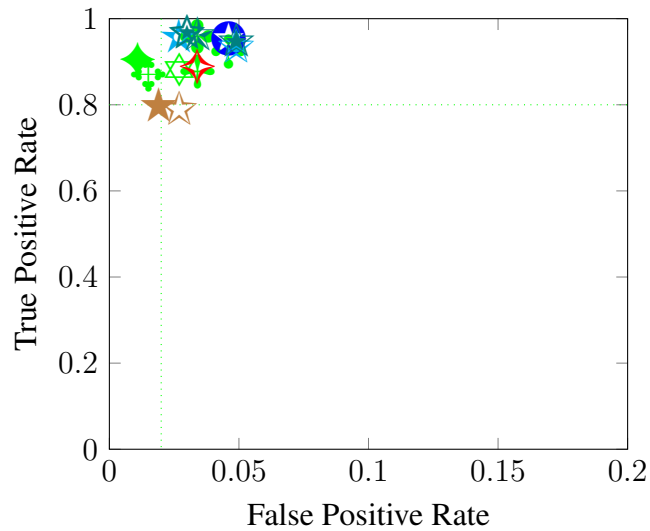


Figure 9.2: Combined algorithm, sports database, standard distortions

Overall, the standard distortions shown in Figure 9.2 shows a true positive rate of 0.97 with a false positive rate of 0.055 while the complex distortions shown in Figure 9.3 shows 0.94 and 0.032 respectively. This experiment has produced more false positives than ob-

Combined GRID and PIXL Algorithm
Sports video database
Using 20,000 production reference video signatures
Representing 7,750 hours of video



- ✧ Crops $\frac{1}{8}$ width from the left hand side of the frame(0.027, 0.882)
- ✧ Crops $\frac{1}{8}$ width from the right hand side of the frame(0.034, 0.878)
- ✧ Crops $\frac{1}{8}$ height from the top of the frame(0.046, 0.924)
- ✧ Crops $\frac{1}{8}$ height from the bottom of the frame(0.034, 0.958)
- ✧ Crops $\frac{1}{8}$ width from LHS and $\frac{1}{8}$ height from the top(0.015, 0.871)
- ✧ Crops $\frac{1}{8}$ width from the RHS and $\frac{1}{8}$ height from the bottom(0.011, 0.905)
- ✧ Flip frames vertically(0.034, 0.890)
- ★ Rotate frames 20 degrees clockwise(0.019, 0.802)
- ★ Rotate frames 20 degrees anti-clockwise(0.027, 0.791)
- ★ Moving picture within frame(0.046, 0.954)
- ★ RHS slant perspective view(0.027, 0.962)
- ★ LHS slant perspective view(0.049, 0.939)
- ★ PIP half size(0.049, 0.951)
- ★ PIP one third size(0.030, 0.970)
- ★ PIP one quarter size(0.034, 0.966)

Figure 9.3: Combined GRID and PIXL algorithms, sports database, complex distortions

tained from the feature film database - this might be explained by the repetitive nature of sporting events and the possibility of these repetitive sequences matching other unrelated sequences in the noise videos. Also, the sports database was synthetically created and has a disproportionate number of unusual distortions compared with the distribution of distortions that might be found in a real-world database. However, as noted in Section 3.3.2, synthetically produced distortions are generally easier to match than real-world distortions. Both the GRID and PIXL algorithms are susceptible to matching repeat sequences in repetitive sporting events such as snooker where the camera position and view is quite often static for long periods of time. These static sequences are likely to match within and across videos of the same sport.

Since the minimal acceptable standard for false positive rate is 0.02 - increasing the match score threshold on the PIXL and GRID algorithms was used to obtain a lower false positive rate. Figure 9.4 shows a variety of combinations for GRID and PIXL match threshold scores.

Combined GRID and PIXL Algorithm
Sports video database
Using 20,000 reference video signatures (noise videos)
Representing 7,750 hours of video
Comparing variations in match score thresholds

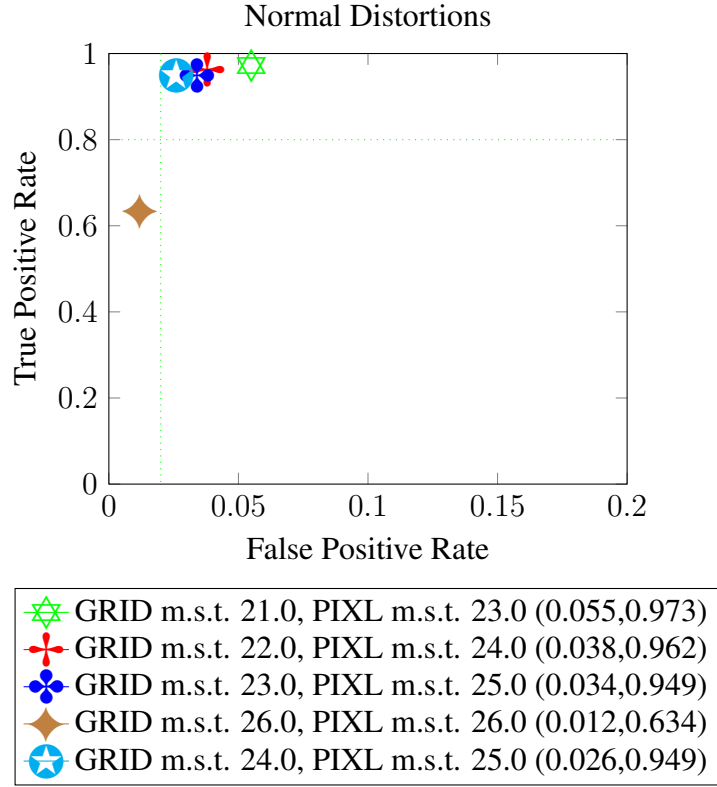


Figure 9.4: Combined GRID and PIXL algorithms with various match score thresholds

Based on this experiment on the sports database, shown in Figure 9.4 using a match score threshold of 24.0 and 25.0 for the GRID and PIXL algorithm respectively produces a true positive rate of 94.9% and a false positive rate of 2.6%. While the false positive score is slightly higher than the overall objective it is probably close enough to build an initial video matching service.

9.3 Combined Algorithms with VCBD Database

The publicly available video database VCDB [41] has been used by several authors to evaluate the performance of their video matching algorithms [1–3], and [42]. Unfortu-

nately, this database does not contain definitive reference videos; rather, it consists of 28 sets of distorted videos. No reference videos are provided for these sets. The approach proposed in [41] is to treat each set of videos as a collection of pairs to be matched. Therefore, the task with the VCDB database is to match distorted videos with each other. This is a significantly different objective from that of this study. In this study, a definitive reference is provided, and the task is to match distorted video clips to this reference. Since many of the videos in VCDB have different frame sequences and exhibit distortions beyond the scope of this study, using VCDB as a basis to assess the results of this study presents additional challenges.

In addition to the miss-match in objectives between VCDB and this study, the distribution of videos in each set is wide, see Figure 9.5. In the smallest set containing 3 videos, only 3 unique comparison pairs are possible, whereas in the largest set with 43 videos, 903 unique pairs can be compared, calculated as $\frac{43!}{2!(43-2)!} = 903$. As a result, using VCDB inherently skews the overall performance towards video sets with a higher number of videos.

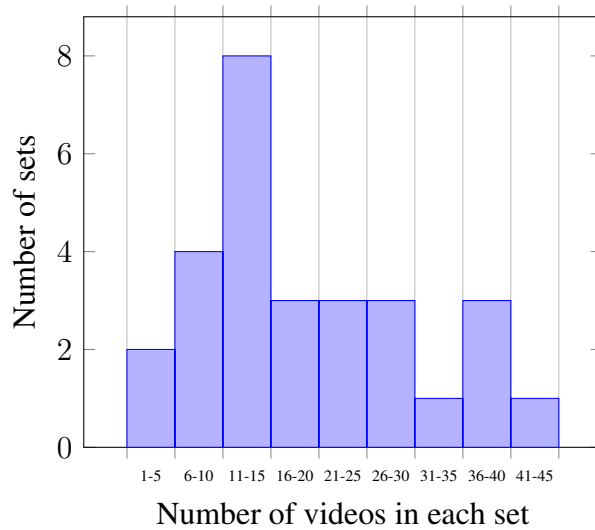


Figure 9.5: VCDB database, distribution of the number of vidoes in each set

In addition to the issues with the VCDB database itself, some of the studies [2, 3] that use VCDB do not include noise videos i.e. videos that should not match the contents of VCDB meaning that the probability of obtaining false positives is much lower than in a real-world database.

Bearing in mind all the limitations with the VCDB database described in this section and the limitations of the previous studies for comparison purposes, an experiment was performed using the GRID and PIXL algorithms defined in this study. This experiment used all the VCDB videos, of 30 seconds duration or longer, as well as, 20,000 noise videos for the references. Since all the algorithms in this study use a minimum duration of 30 seconds only the match pairs from this database that have a match duration of 30 seconds or longer were chosen - this reduces the number of match pairs from 9,236 to 3,765.

Each matched pair of videos (3,765) was tested by first removing one of the pair from the reference set and using the other as the unknown to perform the lookup and then using the other item from the pair in the same way.

The results of this test for the GRID and PIXL algorithms with different match score thresholds are shown in Figure 9.6.

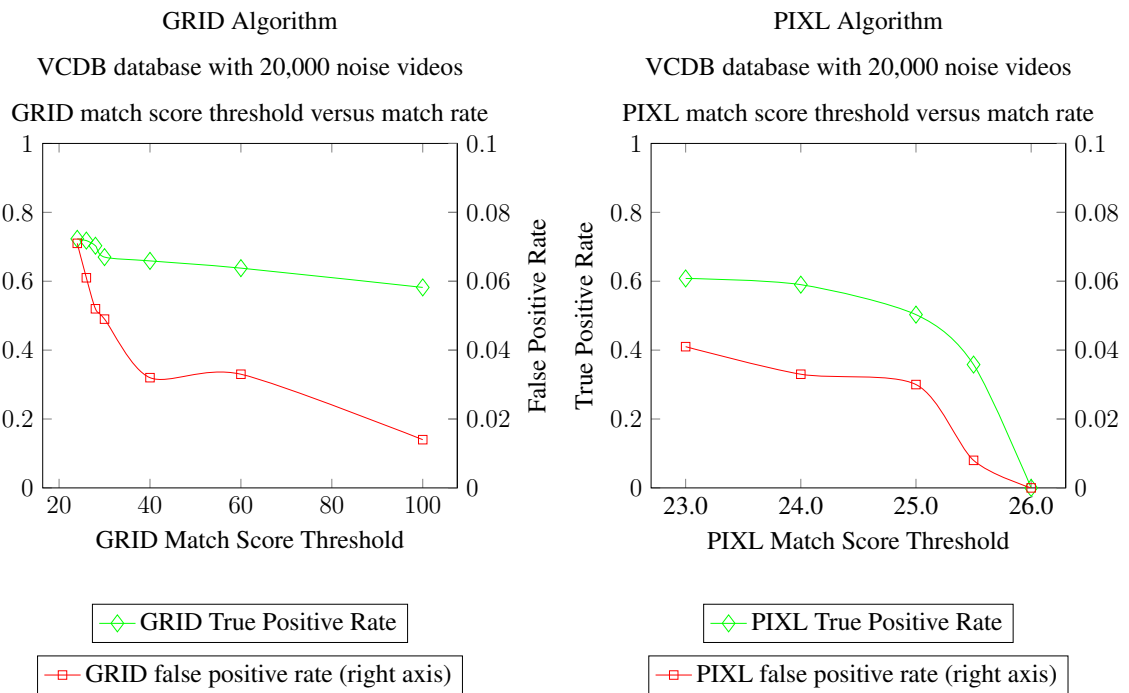


Figure 9.6: GRID and PIXL match score thresholds on VCDB database

The results in Figure 9.6 show that a high match score threshold is required to drive down the false positive rate for both algorithms. This is probably because definitive references are not provided for the video sets in VCDB meaning that distorted videos are matching other VCDB and/or noise videos. For the GRID algorithm a match score thresh-

old of 100 is required to reduce the false positive rate to 2% which is the minimum acceptable standard for this study as described in Section 1.4. At this match score threshold the GRID algorithm yields a true positive rate of only 58%.

The PIXL algorithm performs less well with this database, probably because of the variety of distortions being compared in the match process. A match score threshold of 25.5 yields a false positive rate of 0.8% with a true positive rate of 36%, however the curves are highly non-linear around this match score threshold and drop to zero at a match score threshold of 26.0. This suggests that these results would be difficult to generalise to other similar databases.

Combining the GRID and PIXL algorithms using match score thresholds of 100 and 25.5 respectively yields a true positive rate (TPR) of 73% and a false positive rate (FPR) of 2.1%. This translates into a precision score of $precision = \frac{TPR}{TPR+FPR} = \frac{0.73}{0.73+0.021} = 0.97$. Since recall is calculated as $recall = \frac{TPR}{TPR+FNR}$ and the false negative rate is $FNR = 1 - TPR - FPR - TNR$ but the true negative rate (TNR) is zero as we are only testing items that exist in the database, therefore $FNR = 1 - 0.73 - 0.021 = 0.25$, therefore $recall = \frac{0.73}{0.73+0.249} = 0.75$ with the precision value being of far more importance in this study. Using the harmonic mean of precision and recall $F1 = 2 \left(\frac{precision*recall}{precision+recall} \right)$ yields an F1 score of 84%. Therefore the match statistics for the VCDB dataset with the combined GRID and PIXL algorithms are shown in Table 9.1.

Match Statistic	Value
True Positive Rate	0.73
False Positive Rate	0.021
Precision	0.97
Recall	0.75
F_1	0.84

Table 9.1: Match statistics for VCDB dataset using combined GRID and PIXL algorithms

9.3.1 VCDB Comparison to Other Studies

With all the caveats described in this section around the VCDB database, VCDB is one of the very few publicly available databases that other studies have used to assess video match performance. The study [2] produced a comparison of existing methods using VCDB which is reproduced in Table 9.2.

Method	Reference	F_1 score
CNN	[92]	0.6503
LAMV	[58]	0.6740
	[46]	0.6440
	[93]	0.8025
No-transformer	[2]	0.8613
Transformer	[2]	0.8764
Combined GRID & PIXL		0.84

Table 9.2: [2] Comparison with existing methods on VCDB dataset

While the F1 score from this study (0.84) is slightly below the F1 score from [2] using the Transformer network (0.8764) it should be noted that [2] did not use any noise videos. However, as stated in this section, this study only used video matched pairs that are longer than 30 seconds which probably leads to an over estimate of the accuracy of the combined GRID and PIXL matching algorithm.

The study [1] compared the precision and recall performance of a number of approaches to video matching and plotted them in a graph that is reproduced here as Figure 9.7.

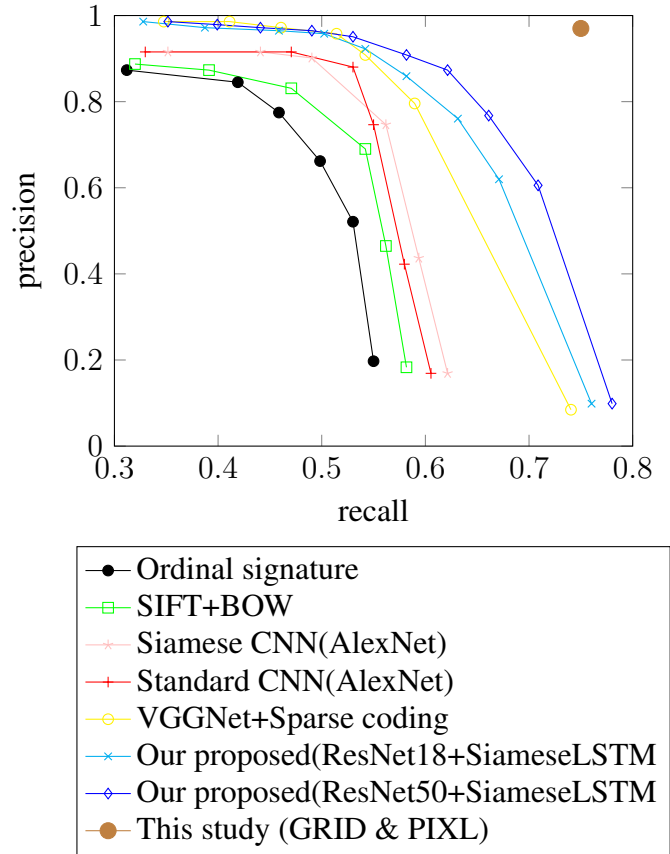


Figure 9.7: [1] Precision-recall curves for different methods on VCDB dataset.

The result from the combined GRID and PIXL experiment are included in the Graph 9.7 as a single point. [1] used 100,000 noise videos to make the reference database more realistic but does not specify how many hours of video was used so a comparison between their study and this study is difficult.

The study [3] produced a comparison of existing methods using the VCDB dataset with no noise videos, which is reproduced in Table 9.3. As before, the results from this study compare favourably with the results from the studies presented in [3].

The study [42] used the same Table 9.3 as [2] but also includes the line that starts with VSAL which is the best result from [42].

Method	Reference	SP	SR	F_1 score
ATN	[92]	0.7050	0.5220	0.5956
CNN	[92]	-	-	0.6503
SNN	[92]	-	-	0.6317
CNN+SNN	[92]	-	-	0.6454
Th+CC+ORB	[94]	0.5052	0.9294	0.6546
LAMV	[58]	-	-	0.6870
CNN+SC (1 fps)	[95]	-	-	0.6995
CNN+SC (all frames)	[95]	-	-	0.7038
BTA	[96]	0.7600	0.7500	0.7549
Our method	[3]	0.8829	0.7355	0.8025
VSAL	[42]	0.8971	0.8462	0.8709
This study		0.97	0.75	0.84

Table 9.3: [3] Comparison of performance of our proposed method at segment-level in the VCDB core dataset.

Again, the results from this study compare favourably with the other studies but is probably not a fair comparison for all the reasons previously stated.

9.4 Throughput

As described in the objectives of this study in Section 1.7, the throughput objective is that the time taken for an average lookup transaction should be less than a second. This transaction rate should be achieved against a reference database of between 10,000 to 50,000 hours of video. The hardware available for this transaction rate is a standard Linux-based server with 8 GPU cards at a cost of approximately \$25,000.

For this throughput analysis the feature film dataset (described here in Appendix D) was bulked out with noise videos to create a reference dataset of 61,000 hours of reference video.

The GRID algorithm uses the FAISS library [52] for the performance critical vector matching step while the PIXL algorithm uses a CUDA-based [97] self built application

to perform the matching. Unfortunately these approaches require separate GPU hardware devices to operate successfully. This limitation means that the reference database has to be loaded onto the GPUs dedicated to the GRID algorithm, as well as, onto the GPUs dedicated to the PIXL algorithm. This effectively means that a single server with 8 GPUs (the target hardware for this study) using the combined GRID and PIXL algorithms has to dedicate 4 GPUs to the GRID algorithm and 4 GPUs to the PIXL algorithm. Therefore the server has only half the effective capacity using the combined GRID and PIXL algorithms of the same hardware that is just using the GRID algorithm.

The GRID and PIXL algorithms are configured for their optimal parameter values described in Table 4.15 and Table 6.13 respectively. The result of the combined performance test is shown in Graph 9.8.

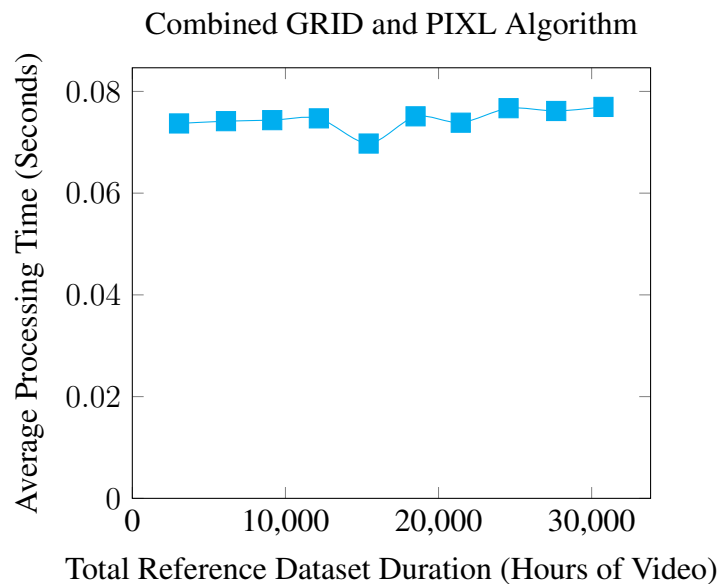


Figure 9.8: Combined GRID and PIXL algorithms average processing time

The Graph 9.8 shows the average processing time for 2,000 match transactions, the throughput is dominated by the GRID algorithm as the throughput of the PIXL algorithm is much higher at these reference database sizes.

The graph shows that the throughput rate for the combined GRID and PIXL algorithm with a 30,000 hour reference video database is $\frac{1}{0.077} = 13$ or 13 transactions per second. This easily meets the objectives of this study of at least one transaction per second.

The throughput statistics from the other studies analysed in this chapter are presented

in a number of different ways as shown in Table 9.4.

Study	Noise Videos	Throughput
[2]	None	Search time per frame 1.48 mS
[42]	None	No performance data provided
[1]	100,000 clips	42 secs for 1 min unknown video
[3]	None	8 secs for 1 min unknown video

Table 9.4: Combined GRID and PIXL results on different datasets

The Table 9.4 shows that even the studies that did not use noise videos produce a throughput of significantly less than one transaction per second on an unknown of one minute of video. However, each study has used a different hardware platform so direct comparisons are questionable. Overall this study has produced a throughput consistent with the requirements of a commercial video matching service.

9.5 Summary

Combining the GRID and PIXL algorithms provides a better match performance on the datasets analysed in this study than either algorithm on its own. These results are summarised in Table 9.5

Dataset	True Positive Rate	False Positive Rate
Feature Film Dataset	0.83	0.020
Sports Dataset	0.95	0.026
VCDB Dataset	0.73	0.021

Table 9.5: Combined GRID and PIXL results on different datasets

The results in Table 9.5 show that the combined algorithms produce results which depend on the underlying dataset. Tuning the algorithms to a particular dataset clearly improves the match performance. This implies that higher precision results will be obtained if video datasets can be kept separate by domain. In general, Audible Magic customers are

focused on a specific genre of video content so that the configuration for a video matching service could be tuned to a specific customer requirement. In addition to segmenting the references, if the genre of the unknown was known or could be determined (perhaps by a neural network) a lookup could be performed against a reference database that contained only the specific genre of content that the unknown contained. This would improve the match accuracy as well as reduce the computational load.

The combined GRID and PIXL algorithm produced a significantly better true positive match rate than the the state of the art studies analysed here on the VCDB dataset. The F_1 scores were comparable particularly when noise videos were included in the analysis.

The throughput rate of the combined GRID and PIXL algorithm easily outperformed any of the comparative studies.

CHAPTER 10

Conclusion

Global revenues from original video content are growing at an unprecedented rate with the Worldwide market for just video streaming projected to reach US\$ 108.5Bn in 2024 [98]. Content owners are looking for services to protect this content from unlicensed use. The sponsor of this study, Audible Magic Inc., provides automatic content recognition services to detect use of this protected content.

The objective of this study was to create and validate video matching algorithms that meet the quality, performance, scale and stability criteria required to create a commercial video matching service. The minimum acceptable standard for these criteria are a video database size of at least 10,000 hours, a true positive rate of 80% or better, a false positive rate of less than 2% with a transaction rate of better than 1 transaction a second. The specific detailed objectives are described in Table 1.4. The most demanding objectives for the video matching algorithms are the match accuracy, database size and transaction rate.

The literature review in Chapter 2 found few studies that used a reference video database of more than 1,000 hours of video while this study requires a minimum reference database size of 10,000 hours of video. Similarly, no studies were found where a transaction rate of more than one transaction per second was achieved against a database

larger than 200 hours of video.

A common methodology for analysing video matching algorithms was described in Chapter 3 along with a description of the datasets used in this study. Unfortunately, very few large publicly available datasets for partial video copy detection are available making comparisons with other studies difficult. The primary method used to assess the quality of the matching algorithms developed in this study was to compare them with each other. Once a high-performing matching algorithm had identified an attempt was made to compare it with other matching algorithms described in the literature in Chapter 9.

Chapter 4 describes the GRID algorithm which uses a simple box filter on each frame to construct a feature vector. This algorithm met the throughput objective but fell short of the match accuracy goals particularly for cropped videos. Chapter 5 considered spatial points of interest within each frame but found that even a small amount of noise made this approach computationally impractical. Chapter 6 analysed a temporal points of interest algorithm (PIXL) which is resilient to distortions at frame boundaries. The PIXL algorithm meet the throughput objective but also fell short on the match accuracy objective.

In Chapter 7 a number of pre-trained convolutional neural networks were used to compute a frame-level feature vector as the basis for matching a sequence of video frames. Unfortunately the size of the feature vectors and the accuracy of the match process on a real-world database fell well short of the objectives of this study. Chapter 8 analysed the benefits of training a layer of certain pre-trained neural networks, the training improved the match rate but still fell well short of the match accuracy required.

Finally in Chapter 9 the GRID and PIXL algorithms were combined to form a hybrid method which met the minimum objectives of this study. Using the VCDB video database [41] this hybrid method was compared with the state of the art algorithms from the literature ([1, 3, 42, 80]) and was found to out-perform these studies both in throughput and match accuracy.

Table 10.1 summaries the performance of the algorithms analysed in this study.

Chapter	4	5	6	7	8	9
Algorithm	GRID	SPOI	PIXL	P-CNN	T-CNN	Combined
Database size (hours)	D	-	D	-	-	M
Unknown video clip duration (seconds)	M	-	M	-	-	M
Throughput rate (transactions per second)	D	-	D	-	-	D
GPU memory per second of video (bytes)	D	-	D	-	-	M
True positive rate	-	-	-	-	-	M
False positive rate	M	-	M			M

Table 10.1: Overall video matching results by algorithm

SPOI is spatial points of interest from Chapter 5, P-CNN is pre-trained convolutional neural networks from Chapter 7, T-CNN is trained convolutional neural networks from Chapter 8 and Combined is the combined GRID and PIXL algorithm from Chapter 9.

In the Table 10.1, D means the algorithm met the desired standard for the objective, M means the algorithm met the minimum acceptable standard for the objective, blank entries mean that the algorithm did not meet the minimum acceptable standard.

Only the Combined GRID and PIXL algorithm from Chapter 9 met the minimum acceptable standard for all the objectives while the pure GRID and PIXL algorithms only missed the true positive rate objective by 10%.

10.1 Areas for Future Research - Existing Algorithms

Areas for future research into high-performance video clip matching can be considered in two specific areas. Firstly, improvements to the matching algorithms used in this study and secondly further matching algorithms not used in this study.

For the algorithms used in this study the GRID algorithm, from Chapter 4, has limited potential due to its inherent weakness in matching cropped videos. Methods to extend the algorithms effectiveness by using a registration point (centre of illumination) had limited

success. The method is fast, simple and capable of rejecting false positives quickly so can form the basis of a hybrid algorithm for matching videos.

The spatial points of interest algorithm discussed in Chapter 5 also appears to have limited future potential. The quantity of information required to match a 30 to 40 second video appears to be prohibitive if each frame is considered in isolation.

The temporal points of interest algorithm (PIXL) from Chapter 6 has significantly more potential for research as it does not appear in the literature and performs reasonably well. The method chosen in this study used a bitmap to record the illumination changes from pixel to pixel. A normalised feature vector rather than a bitmap could be used to improve the match accuracy. This approach is likely to increase the computational load - however the technique has significant head room as the method described in this study supported 42 transactions per second.

The convolutional neural networks explored in Chapters 7 and 8 produced disappointing results. Since a significant amount of research is currently being undertaken in machine learning models [99–101] it seems likely that improved video matching algorithms will be produced over time. Neural network models that are trained on video sequences rather than individual frames may also produce superior results. However, large publicly available real-world video databases are still required to train, test and compare different approaches.

The most successful method in this study, described in Chapter 9, combined two non-machine learning algorithms but required twice the GPU hardware to accomplish this task than each individual algorithm. As GPU hardware becomes more capable and increases memory capacity it seems likely that these limitations will be overcome.

10.2 Areas for Future Research - Other Algorithms

Some initial experimentation was performed using two dimensional frequency spectra of each frame and constructing a short feature vector from the low frequency components. This approach proved unsuccessful probably because cropped video frames do not contain all the information that the corresponding reference video frame contains.

A more promising area of research could involve considering three-dimensional points

of interest that integrate spatial and temporal characteristics. These points of interest could be identified by detecting peaks in frame intensity or frequency spectra as they vary over time. To avoid boundary distortions such as cropping, points of interest should be selected near the center of the frames. Algorithms of this type could be potentially more robust to boundary distortions and noise than those explored in this study.

Bibliography

- [1] Yaocong Hu and Xiaobo Lu. Learning spatial-temporal features for video copy detection by the combination of cnn and rnn. *Journal of Visual Communication and Image Representation*, 55:21–29, 2018. (document), 2.1.1, 2.2.2, 4.5.2, 9.3, 9.3.1, 9.7, 9.3.1, 9.4, 10
- [2] Weijun Tan, Hongwei Guo, and Rushuai Liu. A fast partial video copy detection using knn and global feature database, 2021. (document), 2.2.2, 9.3, 9.3.1, 9.2, 9.3.1, 9.3.1, 9.4
- [3] Zobeida J. G-Zavaleta and Claudia F-Urbe. Partial-copy detection of non-simulated videos using learning at decision level. *Multimedia Tools and Applications*, 78:2427 – 2446, 2018. (document), 2.1.1, 2.1.2, 2.2.2, 3.3.2, 4.5.2, 9.3, 9.3, 9.3.1, 9.3, 9.4, 10
- [4] Tubefilter. More than 500 hours of content are now being uploaded to youtube every minute, 2024. Accessed: 2024-11-11. 1
- [5] Statista. Data volume of global internet video to tv traffic from 2016 to 2021, 2024. Accessed: 2024-11-11. 1
- [6] Hootsuite. 27 inspiring youtube stats to know in 2024, 2024. Accessed: 2024-11-11. 1
- [7] Biteable. Video marketing statistics: The state of video marketing in 2021, 2024. Accessed: 2024-11-11. 1
- [8] Investopedia. How exactly do movies make money?, 2021. Accessed: 2024-11-11. 1.1
- [9] Copyrightuser.org. Is it always necessary to ask permission to use another’s film?, 2021. Accessed: 2024-11-11. 1.1
- [10] RightsDirect. What is copyright?, 2021. Accessed: 2024-11-11. 1.1

- [11] The UK Copyright Service. Uk copyright law: An introduction, 2021. Accessed: 2024-11-11. 1.1
- [12] Deloitte. Digital media: the subscription prescription, 2020. Accessed: 2024-11-11. 1.1
- [13] Stefan Miller. Video piracy’s impact and the technology to stop it, 2024. Accessed: 2024-11-11.
- [14] VideoMaker. Why copyright will be the biggest issue for youtube in 2019 (updated), 2019. Accessed: 2024-11-11. 1.1
- [15] Fast Company. Youtube is using ai to police copyright—to the tune of \$2 billion in payouts, 2016. Accessed: 2024-11-11. 1.1
- [16] US Copyright Office. Copyright in derivative works and compilations, 2020. Accessed: 2024-11-11. 1.1
- [17] “Influencer Marketing Hub“. The ultimate tiktok video size guide for 2024, 2024. Accessed: 2024-11-11. 1.1
- [18] Shon Patil. Fundamentals of digital watermarking. Msc, University of Houston, 2014. 1.4
- [19] Fabrice Bellard. ffmpeg: A complete, cross-platform solution to record, convert and stream audio and video., 2024. Accessed: 2024-11-11. 1.5
- [20] Splasheo. Splasheo: Scale your short-form video content, without scaling your team, 2024. Accessed: 2024-11-11.
- [21] Wistia. Wistia: Turn your videos into marketing machines, 2024. Accessed: 2024-11-11.
- [22] Vidyad. Vidyad: Wow your buyers and win more deals, 2024. Accessed: 2024-11-11. 1.5
- [23] Tao Wu, Runyu He, Gangshan Wu, and Limin Wang. Sportshhi: A dataset for human-human interaction detection in sports videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18537–18546, June 2024. 1.6
- [24] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, Nov 2004. 1.6, 3.2.2, 5, 5.3
- [25] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 110:346–359, June 2008. 1.6, 5
- [26] Anlong Ming and Huadong Ma. A blob detector in color images. In *Proceedings of the 6th ACM International Conference on Image and Video Retrieval, CIVR ’07*, pages 364–370, New York, NY, USA, 2007. ACM. 1.6

- [27] Roberto Brunelli. *Template Matching Techniques in Computer Vision: Theory and Practice*. John Wiley & Sons, 2009. 1.6
- [28] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. Brief: Binary robust independent elementary features. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Computer Vision – ECCV 2010*, pages 778–792, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. 1.6, 5
- [29] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary R Bradski. Orb: An efficient alternative to sift or surf. In *ORB: An efficient alternative to SIFT or SURF*. Citeseer, 2011. 1.6, 5
- [30] Ebrahim Karami, Siva Prasad, and Mohamed Shehata. Image Matching Using SIFT, SURF, BRIEF and ORB: Performance Comparison for Distorted Images. *arXiv e-prints*, page arXiv:1710.02726, Oct 2017. 1.6, 5
- [31] Stephen Follows. Stephen follows, film data and education, 2023. Accessed: 2024-11-11. 1.7
- [32] Athanasios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis2. Deep learning for computer vision: A brief review. *Computational Intelligence and Neuroscience*, 2018. 2
- [33] Neena Aloysius and M. Geetha. A review on deep convolutional neural networks. In *2017 International Conference on Communication and Signal Processing (ICCSP)*, pages 0588–0592, 2017.
- [34] Azeddine Elhassouny and Florentin Smarandache. Trends in deep convolutional neural networks architectures: a review. In *2019 International Conference of Computer Science and Renewable Energies (ICCSRE)*, pages 1–8, 2019. 2
- [35] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017. 2
- [36] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015. Accessed: 2024-11-11. 2
- [37] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 2, 7, 7.1
- [38] Kaiyu Yang, Klint Qinami, Li Fei-Fei, Jia Deng, and Olga Russakovsky. Towards fairer datasets: Filtering and balancing the distribution of the people subtree in the imagenet hierarchy. In *Conference on Fairness, Accountability, and Transparency*, 2020. 2
- [39] Mingqi Gao, Feng Zheng, James J. Q. Yu, Caifeng Shan, Guiguang Ding, and Jungong Han. Deep learning for video object segmentation: a review. *Artificial Intelligence Review*, 56(1):457–531, 2023. 2

- [40] Vijeta Sharma, Manjari Gupta, Ajai Kumar, and Deepti Mishra. Video processing using deep learning techniques: A systematic literature review. *IEEE Access*, 9:139489–139507, 2021. 2
- [41] Yu-Gang Jiang, Yudong Jiang, and Jiajun Wang. Vcdb: A large-scale database for partial copy detection in videos. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 357–371, Cham, 2014. Springer International Publishing. 2.1.1, 2.2.2, 9.3, 10
- [42] Zhen Han, Xiangteng He, Mingqian Tang, and Yiliang Lv. Video similarity and alignment learning on partial video copy detection, 2021. 2.1.1, 2.2.2, 4.5.2, 4.6.9, 4.6.9, 9.3, 9.3.1, 9.4, 10
- [43] Van-Hao Le, Mathieu Delalandre, and Donatello Conte. A large-scale tv dataset for partial video copy detection. In Stan Sclaroff, Cosimo Distanto, Marco Leo, Giovanni M. Farinella, and Federico Tombari, editors, *Image Analysis and Processing – ICIAP 2022*, pages 388–399, Cham, 2022. Springer International Publishing. 2.1.2, 3.3.2
- [44] Matthijs Douze, Hervé Jégou, Cordelia Schmid, and Patrick Pérez. Compact video description for copy detection with precise temporal alignment. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Computer Vision – ECCV 2010*, pages 522–535, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. 2.2.1, 6
- [45] NIST. Trecvid, 2024. Accessed: 2024-11-11. 2.2.1, 3.3.2
- [46] Dongsu Liu, Chenhui Huo, and Hao Yan. Research of commodity recommendation workflow based on lsh algorithm. *Multimedia Tools and Applications*, 78(4):4327–4345, Feb 2019. 2.2.1, 3.2.1, 9.3.1
- [47] Xin Feng, Youni Jiang, Xuejiao Yang, Ming Du, and Xin Li. Computer vision algorithms and hardware implementations: A survey. *Integration*, 69:309–320, 2019. 2.2.2, 6
- [48] Laith Alzubaidi, Jinglan Zhang, Amjad J. Humaidi, Ayad Al-Dujaili, Ye Duan, Omran Al-Shamma, J. Santamaría, Mohammed A. Fadhel, Muthana Al-Amidie, and Laith Farhan. Review of deep learning: concepts, cnn architectures, challenges, applications, future directions. *Journal of Big Data*, 8(53), 2021. 2.2.2, 2.3
- [49] Jie Lin, Ling-Yu Duan, Shiqi Wang, Yan Bai, Yihang Lou, Vijay Chandrasekhar, Tiejun Huang, Alex Kot, and Wen Gao. Hnip: Compact deep invariant representations for video matching, localization, and retrieval. *IEEE Transactions on Multimedia*, 19(9):1968–1983, 2017. 2.2.2
- [50] X. Zhang, Y. Xie, and Luan X. et al. Video copy detection based on deep cnn features and graph-based sequence matching. *Wireless Personal Communications*, page 401–416, 2018. 2.2.2

- [51] Van-Hao LE, Mathieu Delalandre, and Donatello Conte. Real-time detection of partial video copy on tv workstation. In *2021 International Conference on Content-Based Multimedia Indexing (CBMI)*, pages 1–4, 2021. 2.2.2
- [52] Facebook Engineering. Faiss: A library for efficient similarity search, 2021. Accessed: 2024-11-11. 2.2.2, 4.3, 4.3.3, 4.8, 7, 7.3, 8.3, 9.4
- [53] Ling Shen, Richang Hong, and Yanbin Hao. Advance on large scale near-duplicate video retrieval. *Frontiers of Computer Science*, 14:2095–2236, 2020. 2.3
- [54] Antonio Orvieto, Samuel L Smith, Albert Gu, Anushan Fernando, Caglar Gulcehre, Razvan Pascanu, and Soham De. Resurrecting recurrent neural networks for long sequences, 2023. 2.3
- [55] Shahroz Tariq, Sangyup Lee, and Simon S. Woo. A convolutional lstm based residual network for deepfake video detection, 2020. 2.3
- [56] Mengyue Li, Yuchun Guo, and Yishuai Chen. Cnn-based commercial detection in tv broadcasting. In *Proceedings of the 2017 VI International Conference on Network, Communication and Computing, ICNCC 2017*, page 48–53, New York, NY, USA, 2017. Association for Computing Machinery. 2.4
- [57] Van-Hao LE, Mathieu Delalandre, and Donatello Conte. Real-time detection of partial video copy on tv workstation. In *2021 International Conference on Content-Based Multimedia Indexing (CBMI)*, pages 1–4, 2021. 2.4
- [58] Lorenzo Baraldi, Matthijs Douze, Rita Cucchiara, and Herve Jegou. Lamv: Learning to align and match videos with kernelized temporal layers. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7804–7813, 2018. 2.5, 9.3.1, 9.3.1
- [59] Omid Jafari, Preeti Maurya, Parth Nagarkar, Khandker Mushfiqul Islam, and Chidambaram Crushev. A survey on locality sensitive hashing algorithms and their applications, 2021. 3.2.1
- [60] Yeguang Li, Liang Hu, Ke Xia, and Jie Luo. Fast distributed video deduplication via locality-sensitive hashing with similarity ranking. *EURASIP Journal on Image and Video Processing*, 2019(1):51, Mar 2019. 3.2.1
- [61] Ryan Weber Martin Loncaric, Bowei Liu. Convolutional hashing for automated scene matching, Feb 2018. 3.2.1
- [62] Y. Qiao, C. Cappelle, Y. Ruichek, and T. Yang. Convnet and lsh-based visual localization using localized sequence matching. *Sensors*, 19, May 2019. 3.2.1
- [63] Roger Weber, Hans-Jorg Schek, and Stephen Blott. A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In *Proceedings of the 24rd International Conference on Very Large Data Bases, VLDB '98*, pages 194–205, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc. 3.2.2

- [64] Newton Spolaôr, Huei Diana Lee, Weber Shoity Resende Takaki, Leandro Augusto Ensina, Claudio Saddy Rodrigues Coy, and Feng Chung Wu. A systematic review on content-based video retrieval. *Engineering Applications of Artificial Intelligence*, 90:103557, 2020. 3.2.2
- [65] X. Bai, H. Yang, J. Zhou, P. Ren, and J. Cheng. Data-dependent hashing based on p-stable distribution. *IEEE Transactions on Image Processing*, 23(12):5033–5046, Dec 2014. 3.2.3
- [66] Xiao Bai, Haichuan Yang, Jun Zhou, Peng Ren, and Jian Cheng. Data-dependent hashing based on p-stable distribution. *IEEE Transactions on Image Processing*, 23(12):5033–5046, 2014. 3.2.3
- [67] Hongtao Xie, Zhineng Chen, Yizhi Liu, Jianlong Tan, and Li Guo. Data-dependent locality sensitive hashing. In *Advances in Multimedia Information Processing – PCM 2014*, pages 284–293, Cham, 2014. Springer International Publishing. 3.2.3
- [68] Jingdong Wang, Heng Tao Shen, Jingkuan Song, and Jianqiu Ji. Hashing for similarity search: A survey, 2014. 3.2.3, 3.2.6
- [69] Piotr Indyk Mayur Datar, Nicole Immorlica and Vahab S. Mirrokni. Locality-sensitive hashing scheme based on p-stable. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2004. 3.2.4, 3.2.4, 3.2.6
- [70] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC 1998, pages 604–613, New York, NY, USA, 1998. ACM. 3.2.4
- [71] Alphonse Mariyagnanaseelan. Locality-sensitive hashing scheme based on p-stable distributions, 2006. Accessed: 2024-11-11. 3.2.6
- [72] Open Media. 4k video downloader, 2024. Accessed: 2024-11-11. 3.3.2
- [73] Li Chen and F. W. M. Stentiford. Video sequence matching based on temporal ordinal measurement. *Pattern Recogn. Lett.*, 29(13):1824–1831, oct 2008. 4.1
- [74] C.J. Solomon and T.P. Breckon. *Fundamentals of Digital Image Processing: A Practical Approach with Examples in Matlab*. Wiley-Blackwell, 2010. 4.4.2
- [75] Jiayi Ma, Xingyu Jiang, Aoxiang Fan, Junjun Jiang, and Junchi Yan. Image matching from handcrafted to deep features: A survey. *International Journal of Computer Vision*, 129:23–79, 01 2021. 5
- [76] Jianbo Shi et al. Good features to track. In *1994 Proceedings of IEEE conference on computer vision and pattern recognition*, pages 593–600. IEEE, 1994. 5.3, 5.3, 5.3, 5.4
- [77] Shaharyar Ahmed Khan Tareen and Zahra Saleem. A comparative analysis of sift, surf, kaze, akaze, orb, and brisk. In *2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, pages 1–10, 2018. 5.4

- [78] R. Mohan. Video sequence matching. In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP '98 (Cat. No.98CH36181)*, volume 6, pages 3697–3700 vol.6, 1998. 6
- [79] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks, 2018. 7, 7.1
- [80] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks, 2020. 7, 7.1, 10
- [81] PyTorch Foundation. Pytorch machine learning library based on the torch library, 2024. Accessed: 2024-11-11. 7, 7.1, 7.1.1
- [82] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 7, 7.2, 7.3.3
- [83] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks, 2019. 7.1
- [84] Forrest N. Iandola, Song Han, Matthew W. Moskewicz, Khalid Ashraf, William J. Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and 0.5mb model size, 2016. 7.1
- [85] F. Chollet. *Deep Learning with Python, Second Edition*. Manning, 2021. 7.1.1
- [86] PyTorch Foundation. Pytorch resnet-50, 2024. Accessed: 2024-11-11. 7.1.2
- [87] Sifeng He, Xudong Yang, Chen Jiang, Gang Liang, Wei Zhang, Tan Pan, Qing Wang, Furong Xu, Chunguang Li, Jingxiong Liu, Hui Xu, Kaiming Huang, Yuan Cheng, Feng Qian, Xiaobo Zhang, and Lei Yang. A large-scale comprehensive dataset and copy-overlap aware evaluation protocol for segment-level video copy detection, 2022. 7.1.2, 8
- [88] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. 7.2, 7.3.3
- [89] Vassileios Balntas, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. Learning local feature descriptors with triplets and shallow convolutional neural networks. In *Bmvc*, volume 1:2, page 3, 2016. 8
- [90] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010. 8
- [91] Hong Xuan, Abby Stylianou, Xiaotong Liu, and Robert Pless. Hard negative examples are hard, but useful, 2021. Accessed: 2024-11-11. 8.2

- [92] Yu-Gang Jiang and Jiajun Wang. Partial copy detection in videos: A benchmark and an evaluation of popular methods. *IEEE Transactions on Big Data*, 2(1):32–42, 2016. 9.3.1, 9.3.1
- [93] Z Jezabel G-Zavaleta and Claudia F-Urbe. Partial-copy detection of non-simulated videos using learning at decision level. *Multimedia Tools and Applications*, 78(2):2427–2446, 2019. 9.3.1
- [94] Zobeida Jezabel G-Zavaleta and Claudia F-Urbe. Towards a video passive content fingerprinting method for partial-copy detection robust against non-simulated attacks. *PLOS ONE*, 11(11):1–19, 11 2016. 9.3.1
- [95] Ling Wang, Yu Bao, Haojie Li, Xin Fan, and Zhongxuan Luo. Compact cnn based video representation for efficient video copy detection. In *MultiMedia Modeling*, pages 576–587. Springer International Publishing, 2017. 9.3.1
- [96] Yue Zhang and Xinxiang Zhang. Effective real-scenario video copy detection. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 3951–3956. IEEE, 2016. 9.3.1
- [97] Nicholas Wilt. *The CUDA Handbook, A Comprehensive Guide to GPU Programming*. Addison-Wesley Professional, 2013. 9.4
- [98] Statista R. Video streaming worldwide, 2024. Accessed: 2024-11-11. 10
- [99] Ed Pizzi, Giorgos Kordopatis-Zilos, Hiral Patel, Gheorghe Postelnicu, Sugosh Nagavara Ravindra, Akshay Gupta, Symeon Papadopoulos, Giorgos Tolias, and Matthijs Douze. The 2023 video similarity dataset and challenge. *Computer Vision and Image Understanding*, 243:103997, 2024. 10.1
- [100] Giorgos Kordopatis-Zilos, Giorgos Tolias, Christos Tzelepis, Ioannis Kompatsiaris, Ioannis Patras, and Symeon Papadopoulos. Self-supervised video similarity learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 4756–4766, June 2023.
- [101] Zhenhua Liu, Feipeng Ma, Tianyi Wang, and Fengyun Rao. A similarity alignment model for video copy segment matching, 2023. 10.1

APPENDIX A

Auxilliary Results for Pre-Trained Neural Network Algorithm

The following graphs show the full video matching results for the pre-trained neural networks which was summarised in chapter 4.

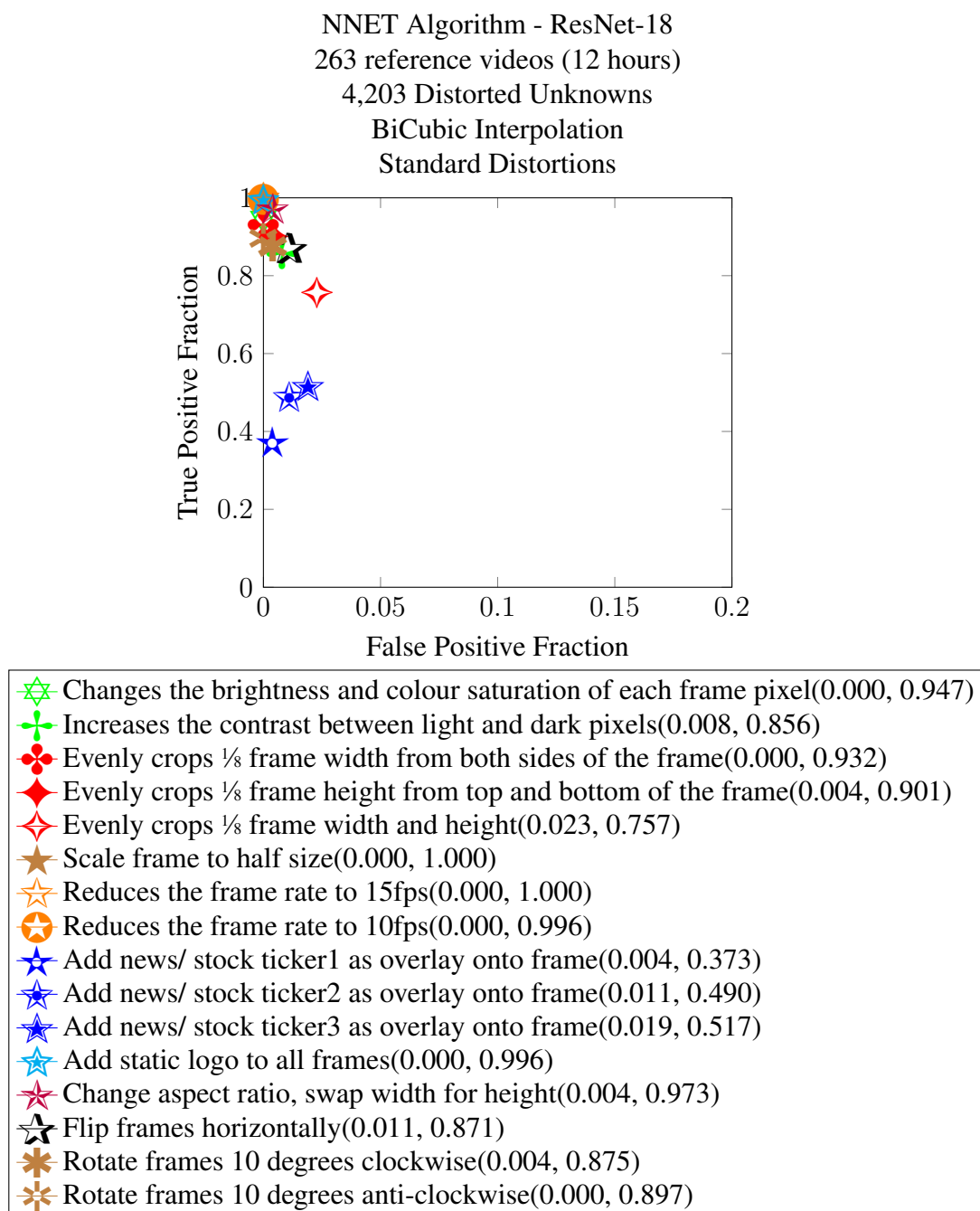


Figure A.1: ResNet-18, bi-cubic interpolation, standard distortions, match performance

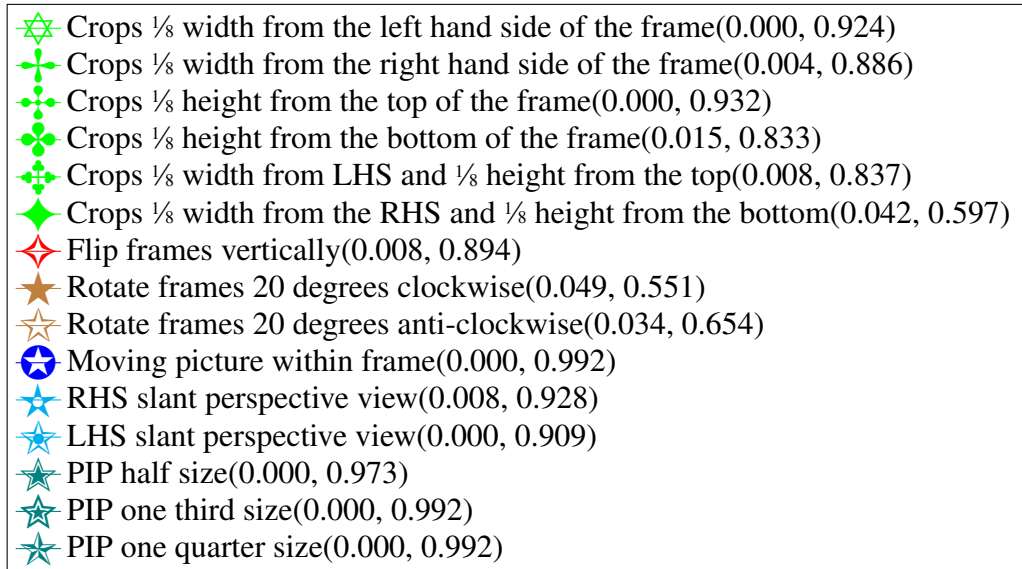
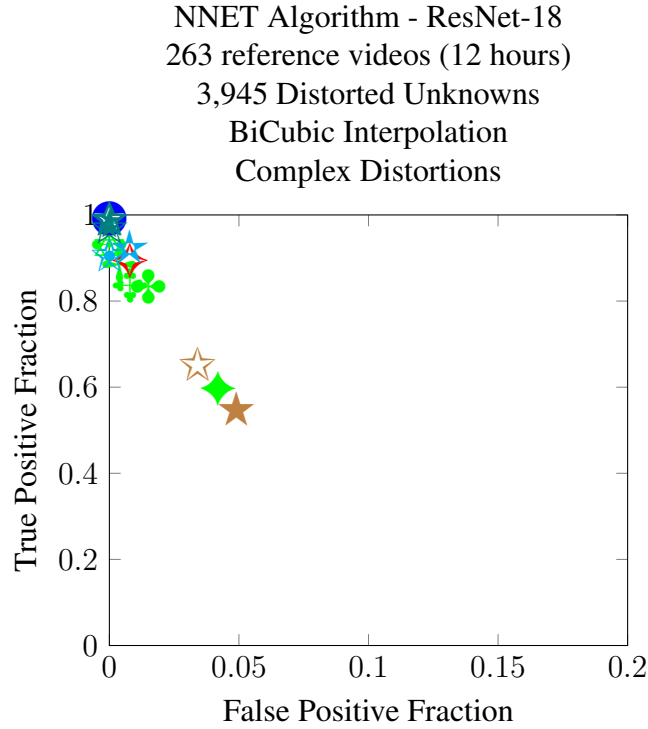


Figure A.2: ResNet-18, bi-cubic interpolation, complex distortions, match performance

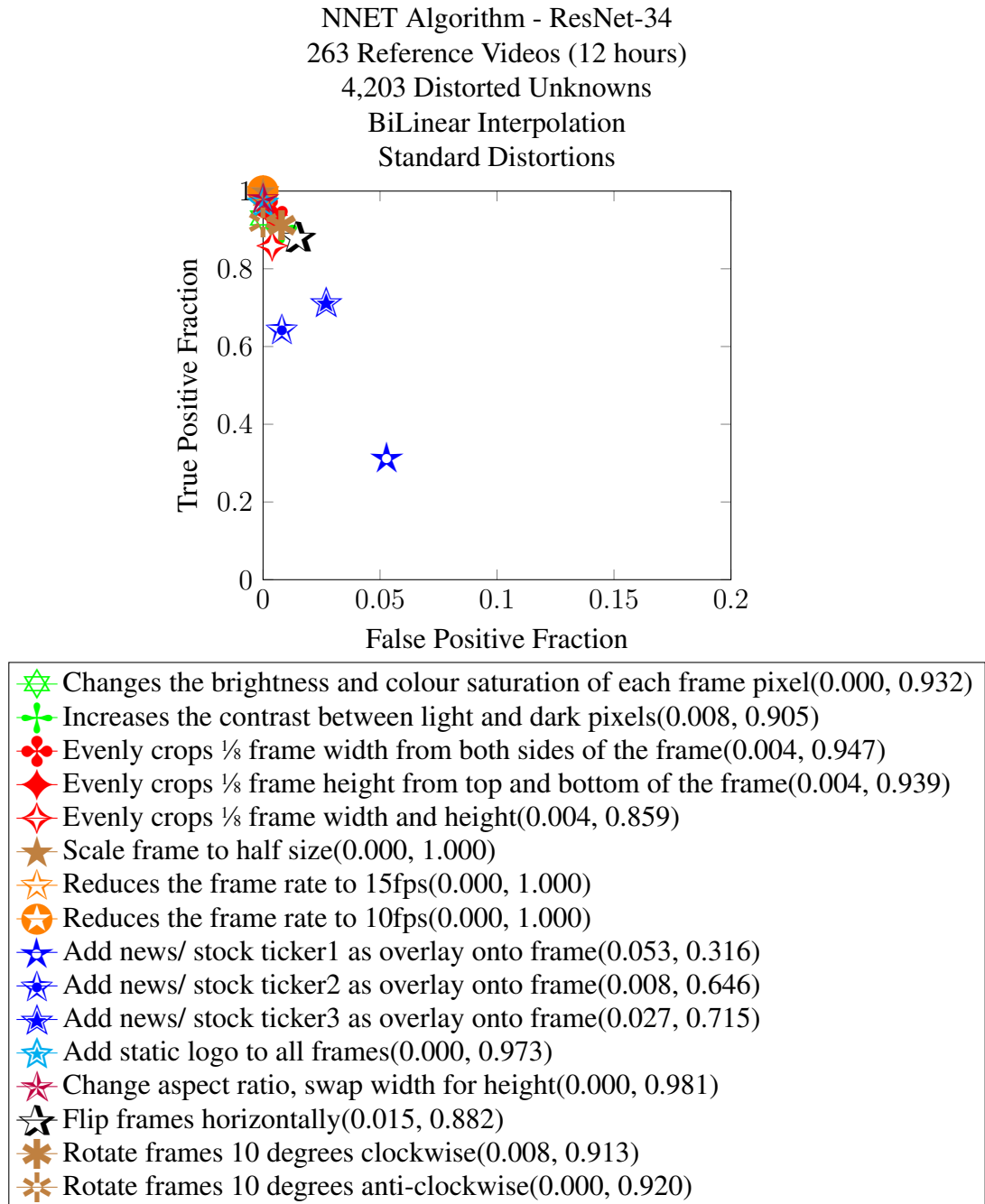


Figure A.3: ResNet-34, bi-linear interpolation, standard distortions, match performance

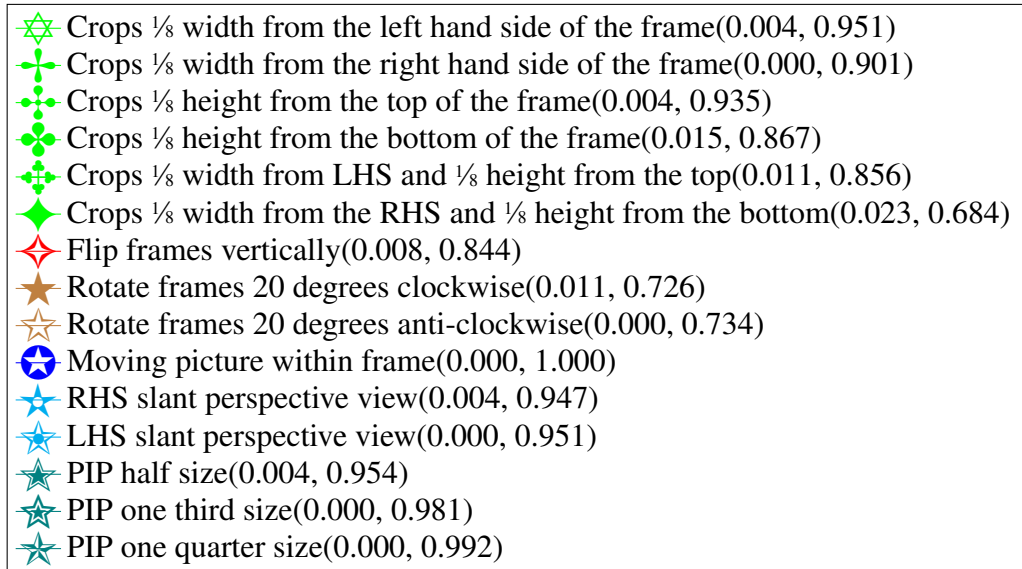
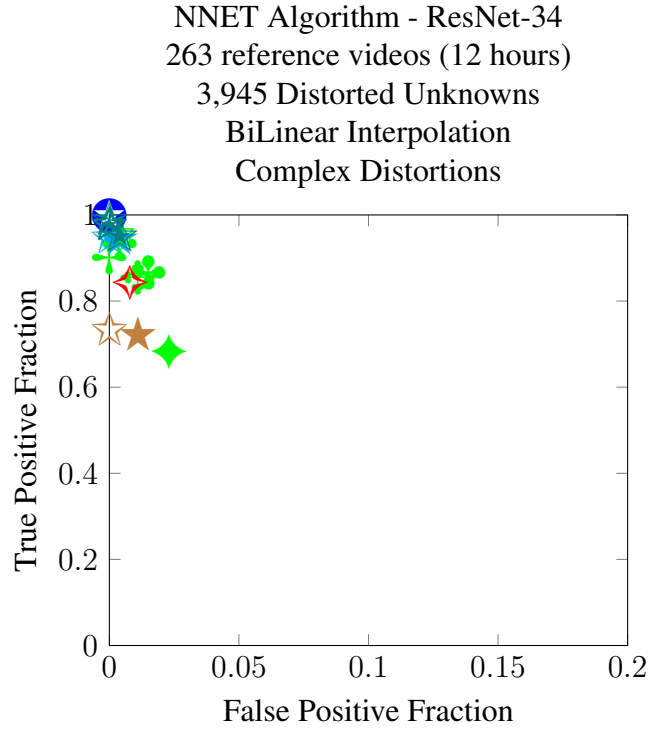
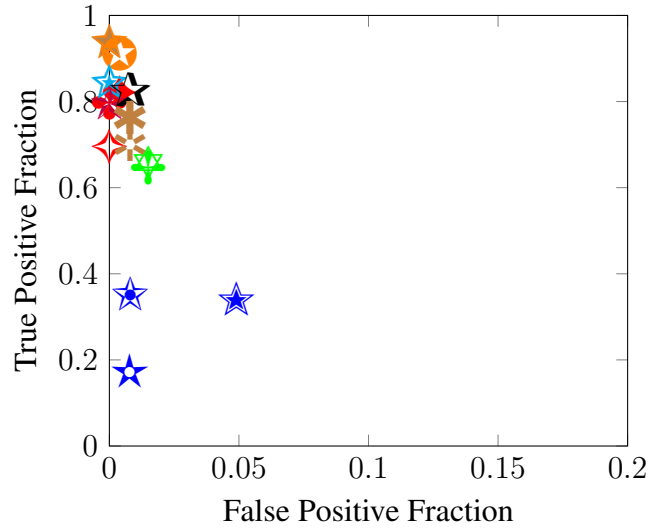


Figure A.4: ResNet-34, bi-linear interpolation, complex distortions, match performance

NNET Algorithm - DenseNet-121
 263 reference videos (12 hours)
 4,203 Distorted Unknowns
 BiLinear Interpolation
 Standard Distortions



- ✧ Changes the brightness and colour saturation of each frame pixel(0.015, 0.658)
- ✧ Increases the contrast between light and dark pixels(0.015, 0.646)
- ✧ Evenly crops 1/8 frame width from both sides of the frame(0.000, 0.798)
- ✧ Evenly crops 1/8 frame height from top and bottom of the frame(0.004, 0.821)
- ✧ Evenly crops 1/8 frame width and height(0.000, 0.696)
- ✧ Scale frame to half size(0.000, 0.939)
- ✧ Reduces the frame rate to 15fps(0.000, 0.939)
- ✧ Reduces the frame rate to 10fps(0.004, 0.913)
- ✧ Add news/ stock ticker1 as overlay onto frame(0.008, 0.175)
- ✧ Add news/ stock ticker2 as overlay onto frame(0.008, 0.354)
- ✧ Add news/ stock ticker3 as overlay onto frame(0.049, 0.342)
- ✧ Add static logo to all frames(0.000, 0.848)
- ✧ Change aspect ratio, swap width for height(0.000, 0.798)
- ✧ Flip frames horizontally(0.008, 0.829)
- ✧ Rotate frames 10 degrees clockwise(0.008, 0.764)
- ✧ Rotate frames 10 degrees anti-clockwise(0.008, 0.700)

Figure A.5: DenseNet-121, bi-linear interpolation, standard distortions, match performance

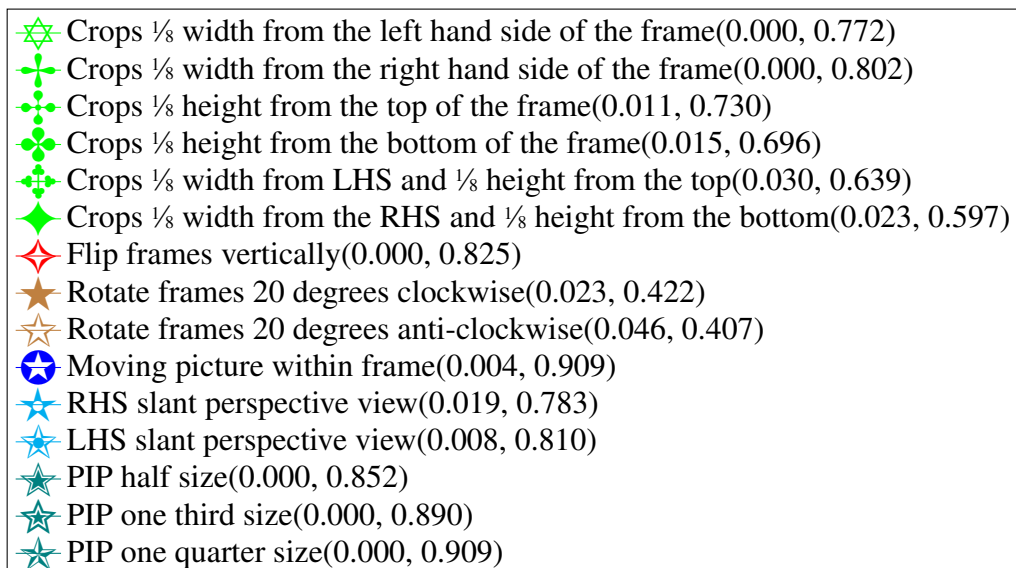
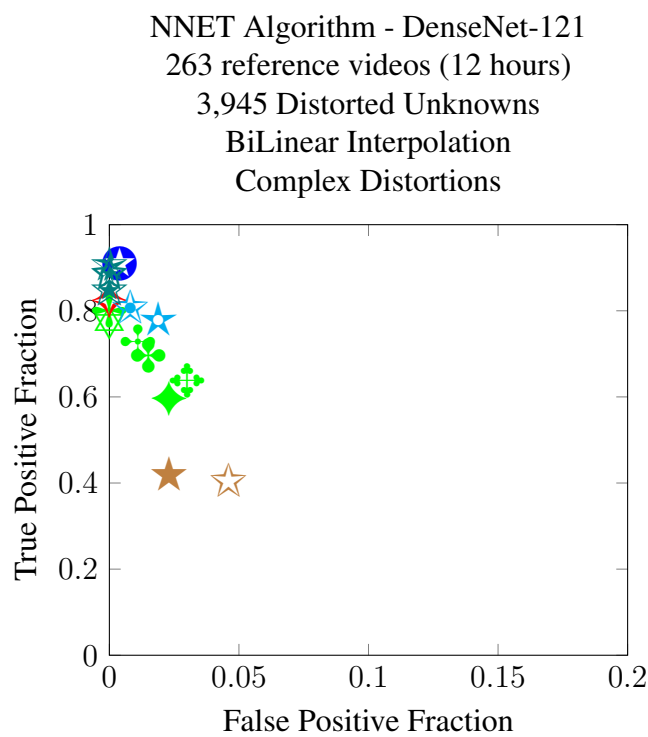
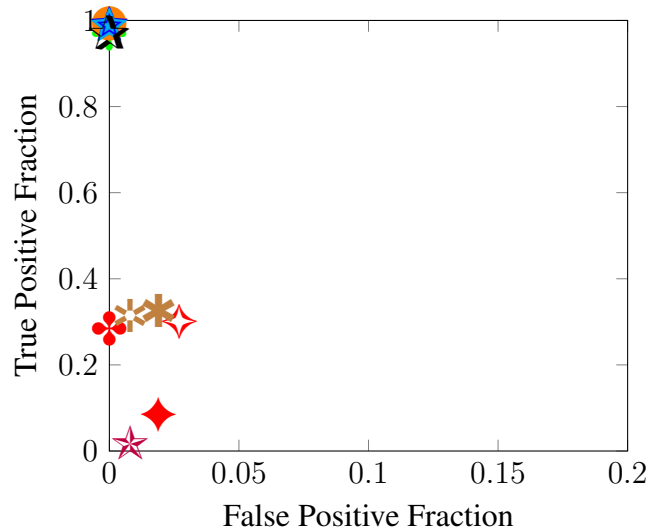


Figure A.6: DenseNet-121, bilinear interpolation, complex distortions, match performance

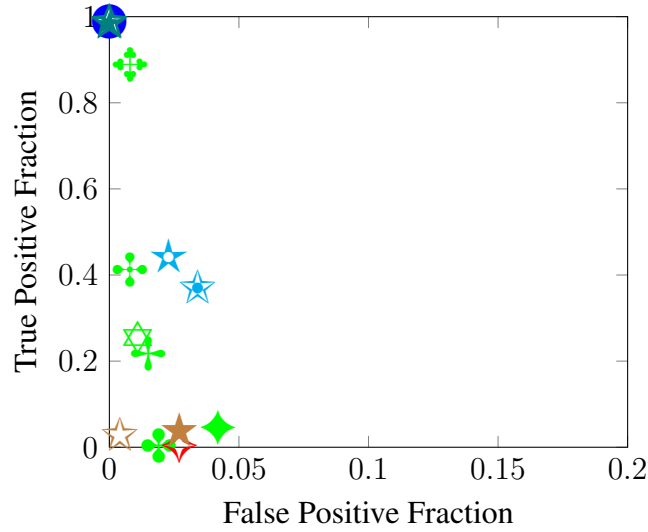
NNET Algorithm - ResNet-10 (Colour Planes @4fps)
 263 reference videos (12 hours)
 4,203 Distorted Unknowns
 BiLinear Interpolation
 Standard Distortions



- ✧ Changes the brightness and colour saturation of each frame pixel(0.000, 0.989)
- ✧ Increases the contrast between light and dark pixels(0.000, 0.970)
- ✧ Evenly crops 1/8 frame width from both sides of the frame(0.000, 0.285)
- ✧ Evenly crops 1/8 frame height from top and bottom of the frame(0.019, 0.084)
- ✧ Evenly crops 1/8 frame width and height(0.027, 0.300)
- ✧ Scale frame to half size(0.000, 0.992)
- ✧ Reduces the frame rate to 15fps(0.000, 0.992)
- ✧ Reduces the frame rate to 10fps(0.000, 0.992)
- ✧ Add news/ stock ticker1 as overlay onto frame(0.000, 0.992)
- ✧ Add news/ stock ticker2 as overlay onto frame(0.000, 0.992)
- ✧ Add news/ stock ticker3 as overlay onto frame(0.000, 0.992)
- ✧ Add static logo to all frames(0.000, 0.992)
- ✧ Change aspect ratio, swap width for height(0.008, 0.019)
- ✧ Flip frames horizontally(0.000, 0.973)
- ✧ Rotate frames 10 degrees clockwise(0.019, 0.327)
- ✧ Rotate frames 10 degrees anti-clockwise(0.008, 0.316)

Figure A.7: ResNet-10, colour planes @4fps, bi-linear interpolation, standard distortions, match performance

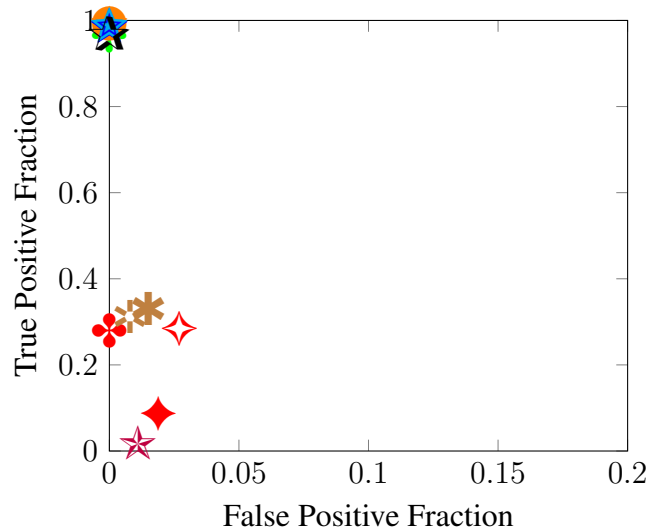
NNET Algorithm - ResNet-10 (Colour Planes @4fps)
 263 reference videos (12 hours)
 3,945 Distorted Unknowns
 BiLinear Interpolation
 Complex Distortions



- ✱ Crops 1/8 width from the left hand side of the frame(0.011, 0.255)
- ✱ Crops 1/8 width from the right hand side of the frame(0.015, 0.217)
- ✱ Crops 1/8 height from the top of the frame(0.008, 0.414)
- ✱ Crops 1/8 height from the bottom of the frame(0.019, 0.004)
- ✱ Crops 1/8 width from LHS and 1/8 height from the top(0.008, 0.890)
- ✱ Crops 1/8 width from the RHS and 1/8 height from the bottom(0.042, 0.046)
- ✱ Flip frames vertically(0.027, 0.004)
- ✱ Rotate frames 20 degrees clockwise(0.027, 0.042)
- ✱ Rotate frames 20 degrees anti-clockwise(0.004, 0.030)
- ✱ Moving picture within frame(0.000, 0.989)
- ✱ RHS slant perspective view(0.023, 0.445)
- ✱ LHS slant perspective view(0.034, 0.373)
- ✱ PIP half size(0.000, 0.989)
- ✱ PIP one third size(0.000, 0.992)
- ✱ PIP one quarter size(0.000, 0.992)

Figure A.8: ResNet-10, colour planes @4fps, bi-linear interpolation, complex distortions, match performance

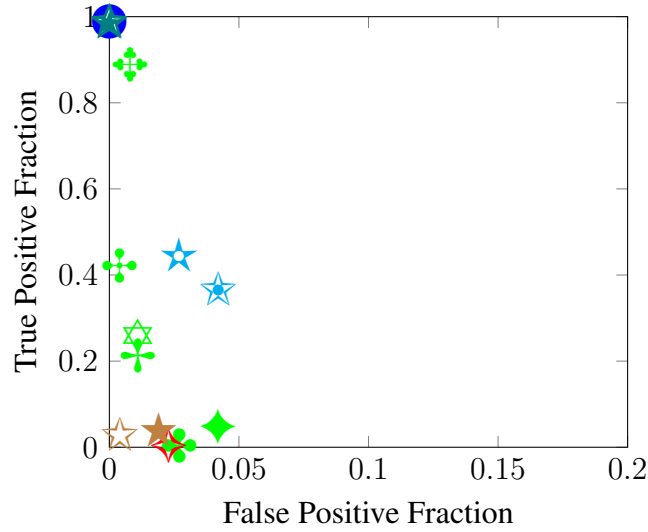
NNET Algorithm - ResNet-10 (Colour Planes @1fps)
 263 reference videos (12 hours)
 4,203 Distorted Unknowns
 BiLinear Interpolation
 Standard Distortions



- ✧ Changes the brightness and colour saturation of each frame pixel(0.000, 0.989)
- ✧ Increases the contrast between light and dark pixels(0.000, 0.966)
- ✧ Evenly crops 1/8 frame width from both sides of the frame(0.000, 0.281)
- ✧ Evenly crops 1/8 frame height from top and bottom of the frame(0.019, 0.087)
- ✧ Evenly crops 1/8 frame width and height(0.027, 0.285)
- ✧ Scale frame to half size(0.000, 0.992)
- ✧ Reduces the frame rate to 15fps(0.000, 0.992)
- ✧ Reduces the frame rate to 10fps(0.000, 0.992)
- ✧ Add news/ stock ticker1 as overlay onto frame(0.000, 0.989)
- ✧ Add news/ stock ticker2 as overlay onto frame(0.000, 0.989)
- ✧ Add news/ stock ticker3 as overlay onto frame(0.000, 0.989)
- ✧ Add static logo to all frames(0.000, 0.992)
- ✧ Change aspect ratio, swap width for height(0.011, 0.019)
- ✧ Flip frames horizontally(0.000, 0.970)
- ✧ Rotate frames 10 degrees clockwise(0.015, 0.331)
- ✧ Rotate frames 10 degrees anti-clockwise(0.008, 0.312)

Figure A.9: ResNet-10, colour planes @1fps, bi-linear interpolation, standard distortions, match performance

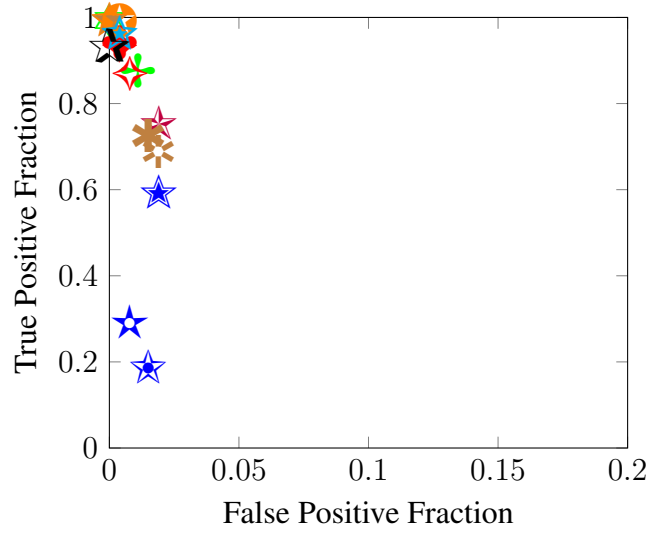
NNET Algorithm - ResNet-10 (Colour Planes @1fps)
 263 reference videos (12 hours)
 3,945 Distorted Unknowns
 BiLinear Interpolation
 Complex Distortions



- ✱ Crops 1/8 width from the left hand side of the frame(0.011, 0.259)
- ✱ Crops 1/8 width from the right hand side of the frame(0.011, 0.213)
- ✱ Crops 1/8 height from the top of the frame(0.004, 0.422)
- ✱ Crops 1/8 height from the bottom of the frame(0.027, 0.004)
- ✱ Crops 1/8 width from LHS and 1/8 height from the top(0.008, 0.890)
- ✱ Crops 1/8 width from the RHS and 1/8 height from the bottom(0.042, 0.049)
- ✱ Flip frames vertically(0.023, 0.004)
- ✱ Rotate frames 20 degrees clockwise(0.019, 0.042)
- ✱ Rotate frames 20 degrees anti-clockwise(0.004, 0.030)
- ✱ Moving picture within frame(0.000, 0.989)
- ✱ RHS slant perspective view(0.027, 0.449)
- ✱ LHS slant perspective view(0.042, 0.369)
- ✱ PIP half size(0.000, 0.989)
- ✱ PIP one third size(0.000, 0.992)
- ✱ PIP one quarter size(0.000, 0.992)

Figure A.10: ResNet-10, colour planes @1fps, bi-linear interpolation, complex distortions, match performance

NNET Algorithm - SqueezeNet
 263 Reference Videos (12 hours)
 4,203 Distorted Unknowns
 BiLinear
 Standard Distortions



- ✧ Changes the brightness and colour saturation of each frame pixel(0.000, 0.992)
- ✧ Increases the contrast between light and dark pixels(0.011, 0.878)
- ✧ Evenly crops 1/8 frame width from both sides of the frame(0.004, 0.943)
- ✧ Evenly crops 1/8 frame height from top and bottom of the frame(0.004, 0.935)
- ✧ Evenly crops 1/8 frame width and height(0.008, 0.871)
- ✧ Scale frame to half size(0.000, 0.996)
- ✧ Reduces the frame rate to 15fps(0.000, 0.996)
- ✧ Reduces the frame rate to 10fps(0.004, 0.992)
- ✧ Add news/ stock ticker1 as overlay onto frame(0.008, 0.293)
- ✧ Add news/ stock ticker2 as overlay onto frame(0.015, 0.190)
- ✧ Add news/ stock ticker3 as overlay onto frame(0.019, 0.597)
- ✧ Add static logo to all frames(0.004, 0.966)
- ✧ Change aspect ratio, swap width for height(0.019, 0.757)
- ✧ Flip frames horizontally(0.000, 0.935)
- ✧ Rotate frames 10 degrees clockwise(0.015, 0.726)
- ✧ Rotate frames 10 degrees anti-clockwise(0.019, 0.688)

Figure A.11: SqueezeNet, bi-linear interpolation, standard distortions, match performance

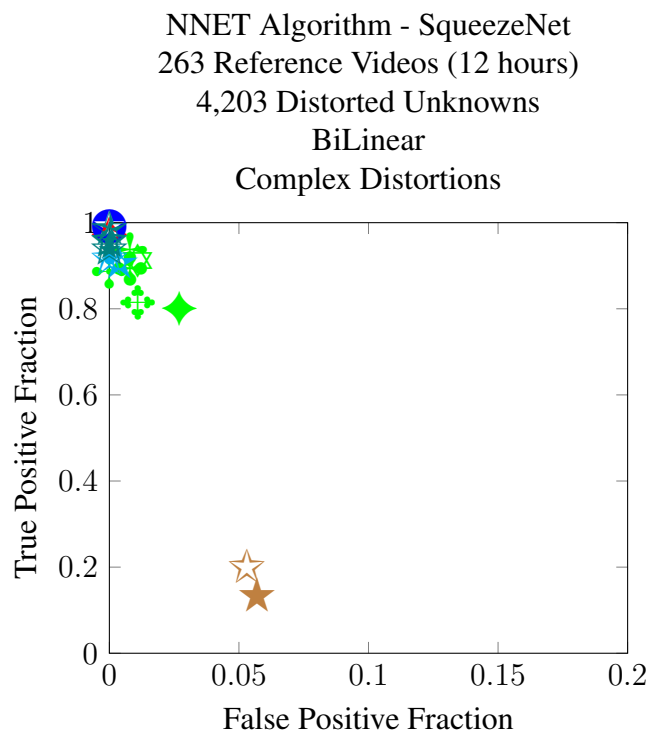
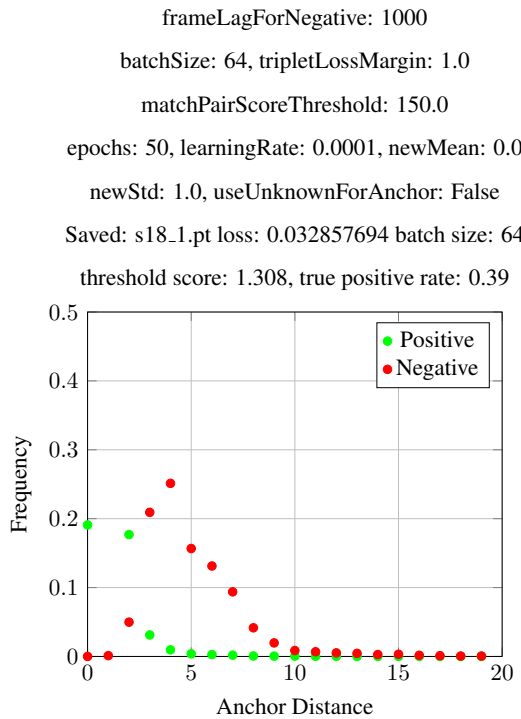


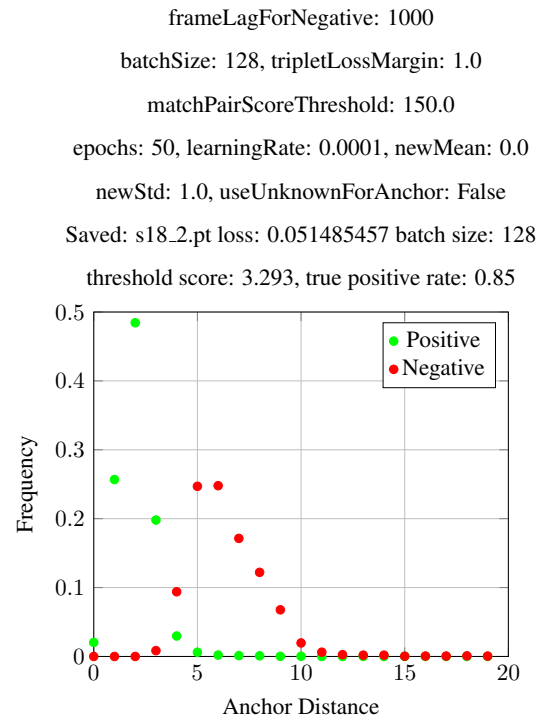
Figure A.12: SqueezeNet, bi-linear interpolation, complex distortions, match performance

APPENDIX B

Auxillary Results for Training Convolutional Neural Networks with Triplet Loss

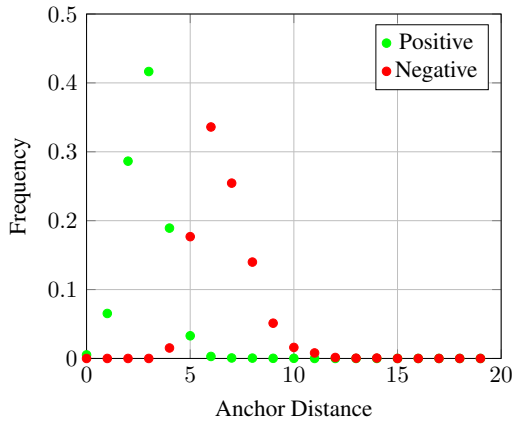


(a) NNET algorithm, a18test_1



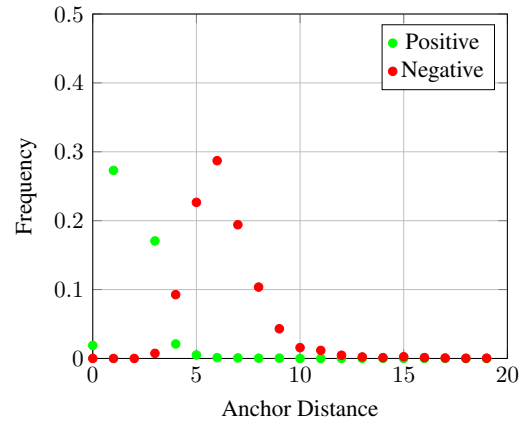
(b) NNET algorithm, a18test_2

frameLagForNegative: 1000
batchSize: 256, tripletLossMargin: 1.0
matchPairScoreThreshold: 150.0
epochs: 50, learningRate: 0.0001, newMean: 0.0
newStd: 1.0, useUnknownForAnchor: False
Saved: s18_3.pt loss: 0.051726893 batch size: 256
threshold score: 4.111, true positive rate: 0.81



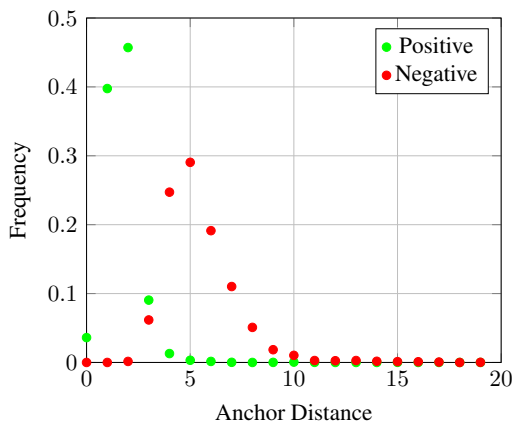
(a) NNET algorithm, a18test_3

frameLagForNegative: 1000
batchSize: 128, tripletLossMargin: 1.0
matchPairScoreThreshold: 150.0
epochs: 50, learningRate: 0.0001, newMean: 0.0
newStd: 1.0, useUnknownForAnchor: False
Saved: s18_4.pt loss: 0.10698559 batch size: 128
threshold score: 3.027, true positive rate: 0.81



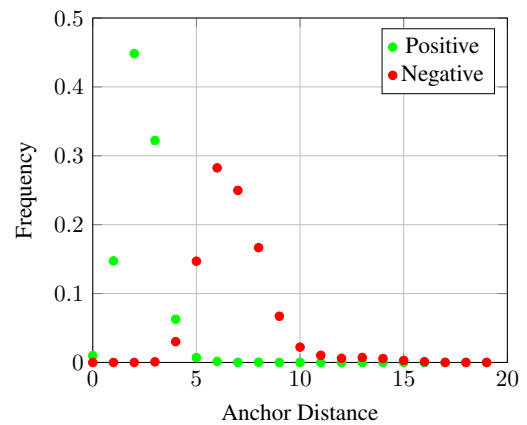
(b) NNET algorithm, a18test_4

frameLagForNegative: 1000
batchSize: 128, tripletLossMargin: 1.0
matchPairScoreThreshold: 150.0
epochs: 50, learningRate: 0.0001, newMean: 0.0
newStd: 1.0, useUnknownForAnchor: False
Saved: s18_5.pt loss: 0.0277526 batch size: 128
threshold score: 2.649, true positive rate: 0.78



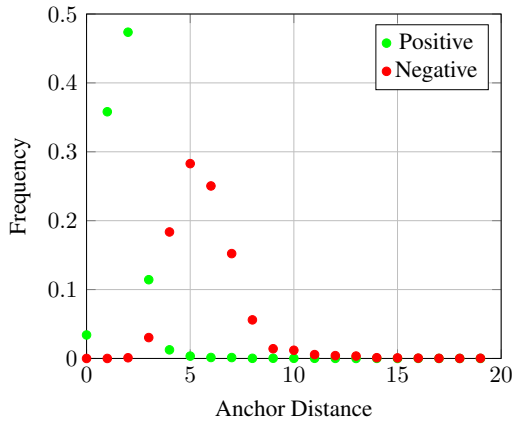
(a) NNET algorithm, a18test_5

frameLagForNegative: 1000
batchSize: 128, tripletLossMargin: 1.0
matchPairScoreThreshold: 150.0
epochs: 50, learningRate: 0.0001, newMean: 0.0
newStd: 1.0, useUnknownForAnchor: False
Saved: s18_6.pt loss: 0.030701553 batch size: 128
threshold score: 3.211, true positive rate: 0.70



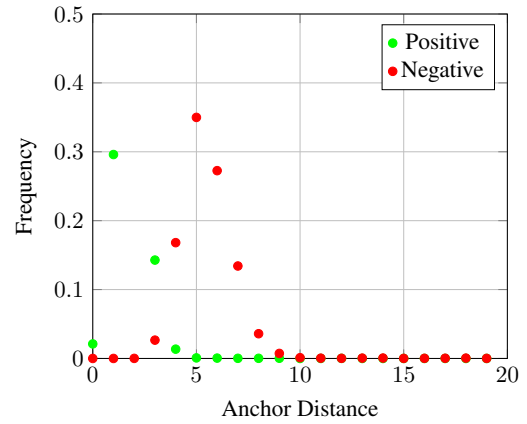
(b) NNET algorithm, a18test_6

frameLagForNegative: 1000
batchSize: 128, tripletLossMargin: 1.0
matchPairScoreThreshold: 100.0
epochs: 50, learningRate: 0.0001, newMean: 0.0
newStd: 1.0, useUnknownForAnchor: False
Saved: s18_7.pt loss: 0.038108148 batch size: 128
threshold score: 2.313, true positive rate: 0.57



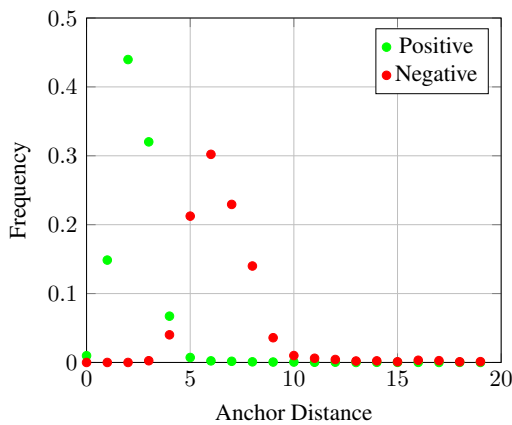
(a) NNET algorithm, a18test_7

frameLagForNegative: 1000
batchSize: 128, tripletLossMargin: 1.0
matchPairScoreThreshold: 200.0
epochs: 50, learningRate: 0.0001, newMean: 0.0
newStd: 1.0, useUnknownForAnchor: False
Saved: s18_8.pt loss: 0.04293019 batch size: 128
threshold score: 2.854, true positive rate: 0.79



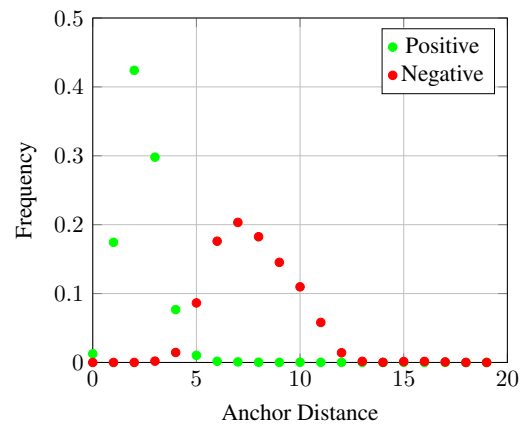
(b) NNET algorithm, a18test_8

frameLagForNegative: 1000
batchSize: 128, tripletLossMargin: 1.0
matchPairScoreThreshold: 150.0
epochs: 50, learningRate: 0.0001, newMean: 0.0
newStd: 1.0, useUnknownForAnchor: False
Saved: s18_9.pt loss: 0.0444863 batch size: 128
threshold score: 3.156, true positive rate: 0.67



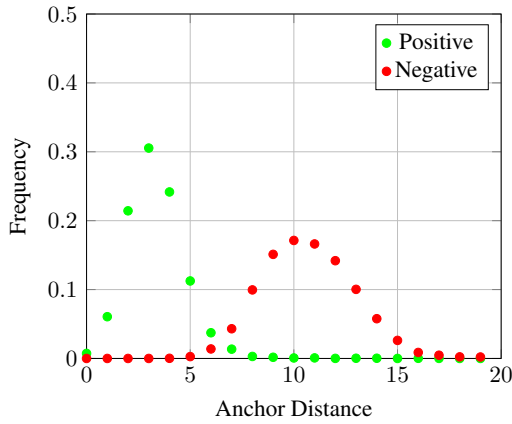
(a) NNET algorithm, a18test_9

frameLagForNegative: 1000
batchSize: 128, tripletLossMargin: 2.0
matchPairScoreThreshold: 150.0
epochs: 50, learningRate: 0.0001, newMean: 0.0
newStd: 1.0, useUnknownForAnchor: False
Saved: s18_10.pt loss: 0.06363927 batch size: 128
threshold score: 3.282, true positive rate: 0.73



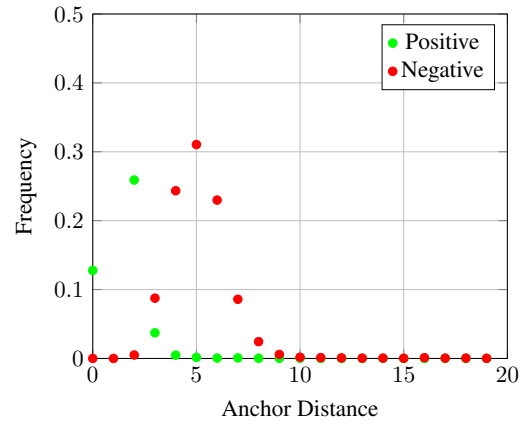
(b) NNET algorithm, a18test_10

frameLagForNegative: 1000
batchSize: 128, tripletLossMargin: 4.0
matchPairScoreThreshold: 150.0
epochs: 50, learningRate: 0.0001, newMean: 0.0
newStd: 1.0, useUnknownForAnchor: False
Saved: s18_11.pt loss: 0.091085374 batch size: 128
threshold score: 4.803, true positive rate: 0.79



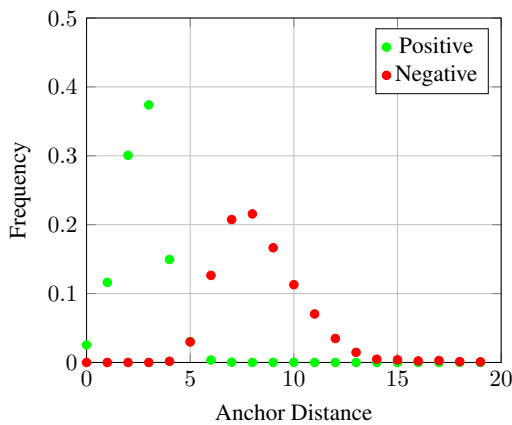
(a) NNET algorithm, a18test_11

frameLagForNegative: 1000
batchSize: 128, tripletLossMargin: 1.0
matchPairScoreThreshold: 150.0
epochs: 100, learningRate: 0.0001, newMean: 0.0
newStd: 1.0, useUnknownForAnchor: False
Saved: s18_12.pt loss: 0.011355658 batch size: 128
threshold score: 2.297, true positive rate: 0.82



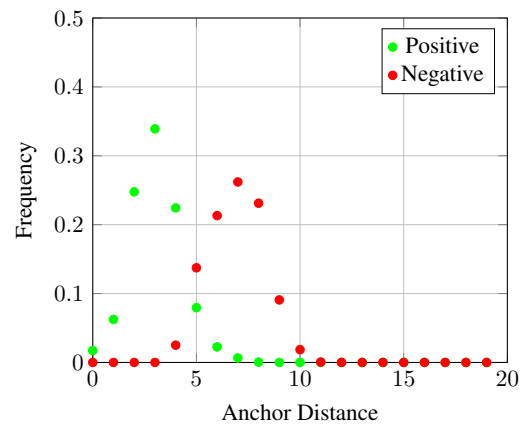
(b) NNET algorithm, a18test_12

frameLagForNegative: 1000
batchSize: 128, tripletLossMargin: 1.0
matchPairScoreThreshold: 150.0
epochs: 50, learningRate: 0.0001, newMean: 0.0
newStd: 1.0, useUnknownForAnchor: True
Saved: s18_13.pt loss: 0.046078812 batch size: 128
threshold score: 4.511, true positive rate: 0.92

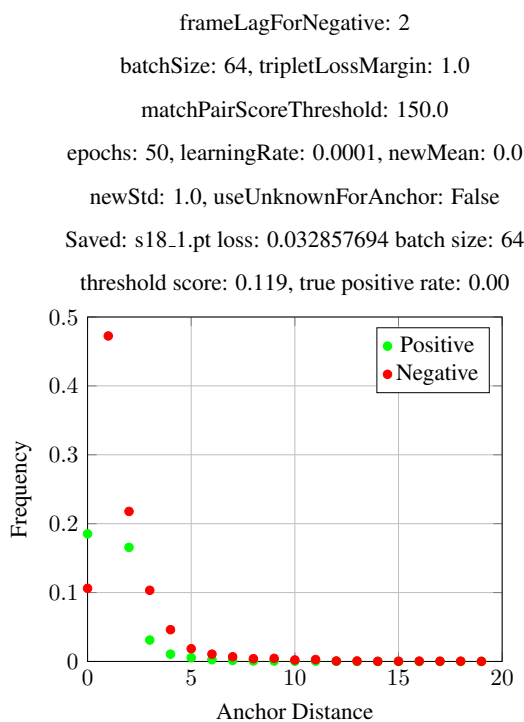


(a) NNET algorithm, a18test_13

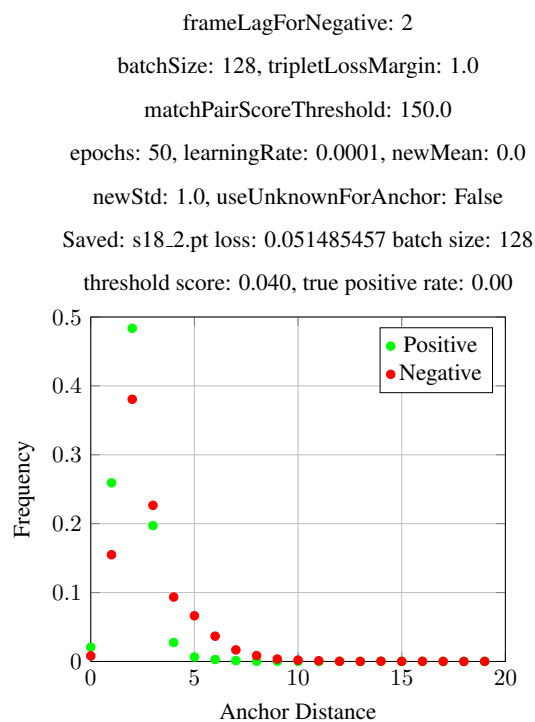
frameLagForNegative: 1000
batchSize: 128, tripletLossMargin: 1.0
matchPairScoreThreshold: 0.456
epochs: 50, learningRate: 0.0001, newMean: 0.0
newStd: 0.224, useUnknownForAnchor: False
Saved: s18_14.pt loss: 0.0009985929 batch size: 128
threshold score: 4.033, true positive rate: 0.67



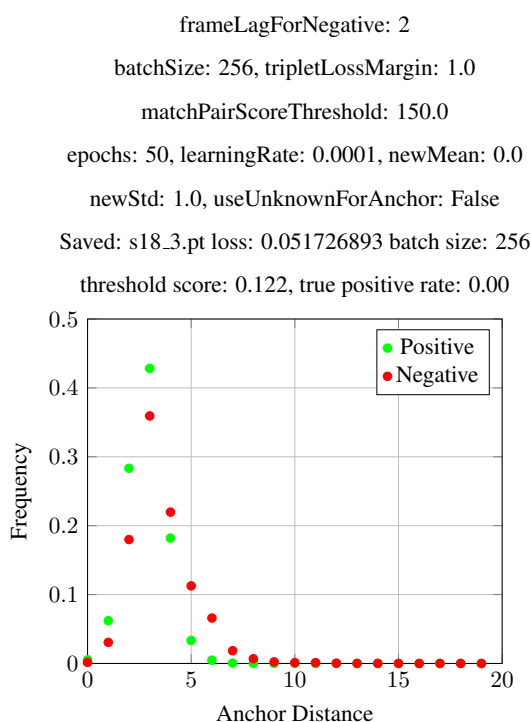
(b) NNET algorithm, a18test_14



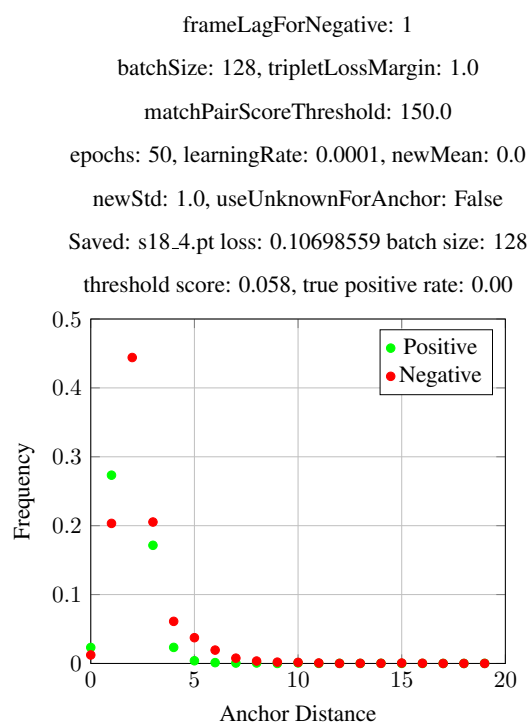
(a) NNET algorithm, b18test_1



(b) NNET algorithm, b18test_2

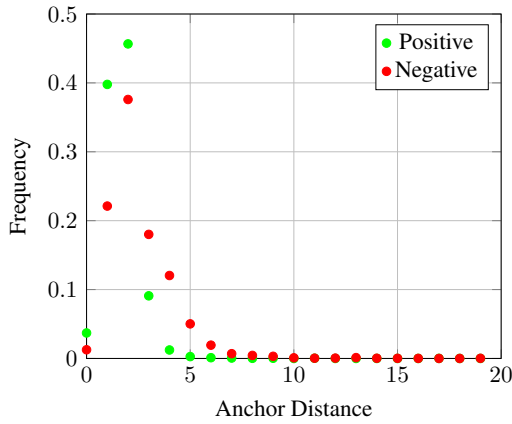


(a) NNET algorithm, b18test_3



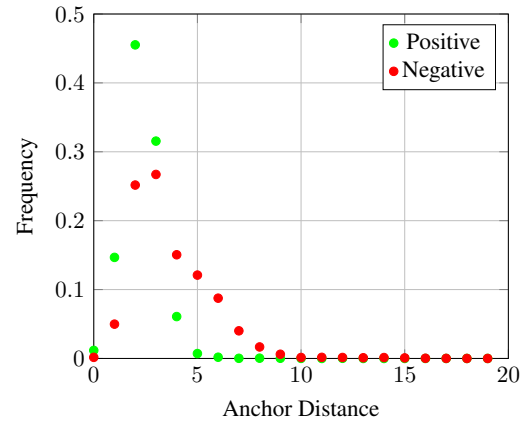
(b) NNET algorithm, b18test_4

frameLagForNegative: 3
 batchSize: 128, tripletLossMargin: 1.0
 matchPairScoreThreshold: 150.0
 epochs: 50, learningRate: 0.0001, newMean: 0.0
 newStd: 1.0, useUnknownForAnchor: False
 Saved: s18_5.pt loss: 0.0277526 batch size: 128
 threshold score: 0.162, true positive rate: 0.00



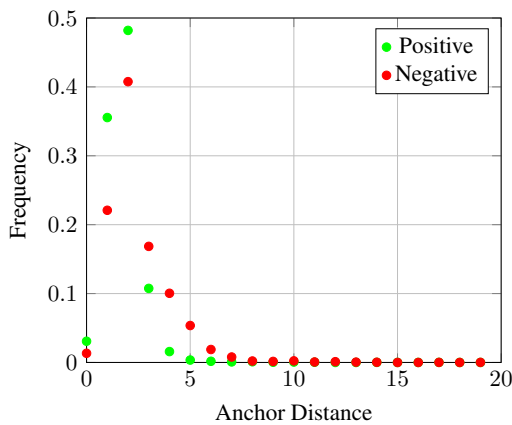
(a) NNET algorithm, b18test_5

frameLagForNegative: 4
 batchSize: 128, tripletLossMargin: 1.0
 matchPairScoreThreshold: 150.0
 epochs: 50, learningRate: 0.0001, newMean: 0.0
 newStd: 1.0, useUnknownForAnchor: False
 Saved: s18_6.pt loss: 0.030701553 batch size: 128
 threshold score: 0.284, true positive rate: 0.00



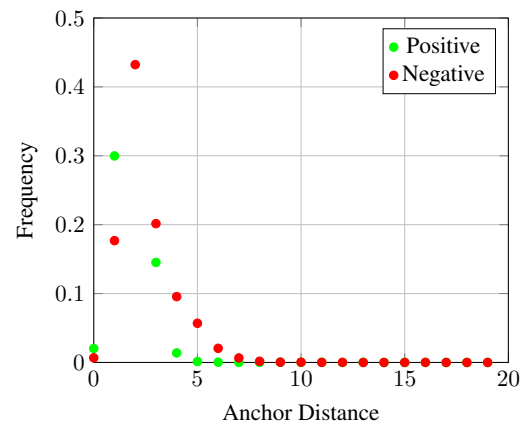
(b) NNET algorithm, b18test_6

frameLagForNegative: 2
 batchSize: 128, tripletLossMargin: 1.0
 matchPairScoreThreshold: 100.0
 epochs: 50, learningRate: 0.0001, newMean: 0.0
 newStd: 1.0, useUnknownForAnchor: False
 Saved: s18_7.pt loss: 0.038108148 batch size: 128
 threshold score: 0.080, true positive rate: 0.00



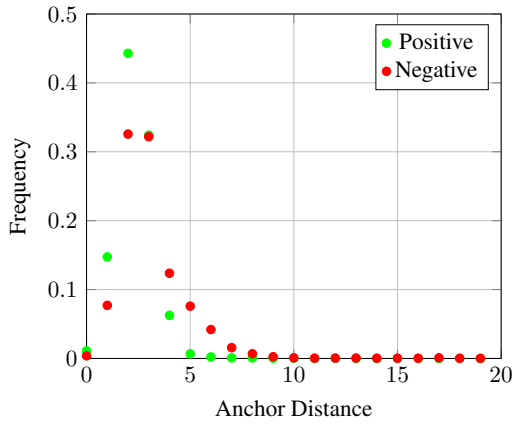
(a) NNET algorithm, b18test_7

frameLagForNegative: 2
 batchSize: 128, tripletLossMargin: 1.0
 matchPairScoreThreshold: 200.0
 epochs: 50, learningRate: 0.0001, newMean: 0.0
 newStd: 1.0, useUnknownForAnchor: False
 Saved: s18_8.pt loss: 0.04293019 batch size: 128
 threshold score: 0.059, true positive rate: 0.00



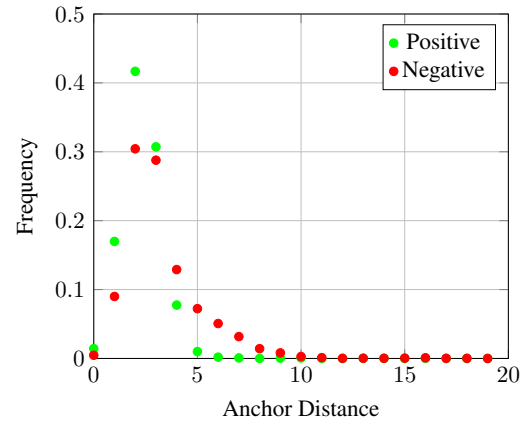
(b) NNET algorithm, b18test_8

frameLagForNegative: 2
batchSize: 128, tripletLossMargin: 1.0
matchPairScoreThreshold: 150.0
epochs: 50, learningRate: 0.0001, newMean: 0.0
newStd: 1.0, useUnknownForAnchor: False
Saved: s18_9.pt loss: 0.0444863 batch size: 128
threshold score: 0.226, true positive rate: 0.00



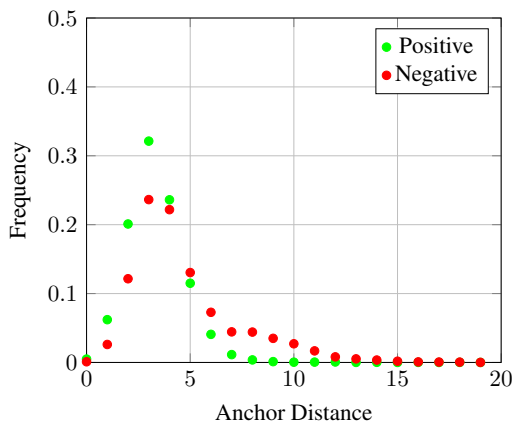
(a) NNET algorithm, b18test_9

frameLagForNegative: 2
batchSize: 128, tripletLossMargin: 2.0
matchPairScoreThreshold: 150.0
epochs: 50, learningRate: 0.0001, newMean: 0.0
newStd: 1.0, useUnknownForAnchor: False
Saved: s18_10.pt loss: 0.06363927 batch size: 128
threshold score: 0.066, true positive rate: 0.00



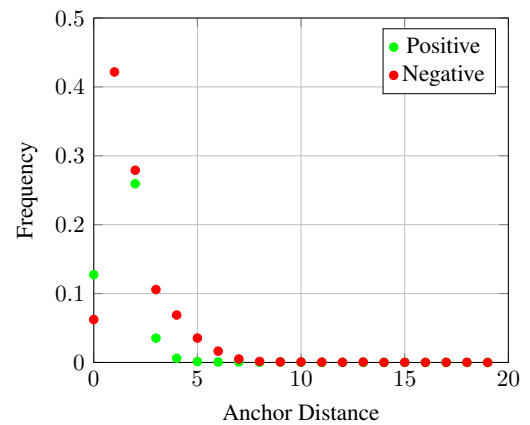
(b) NNET algorithm, b18test_10

frameLagForNegative: 2
batchSize: 128, tripletLossMargin: 4.0
matchPairScoreThreshold: 150.0
epochs: 50, learningRate: 0.0001, newMean: 0.0
newStd: 1.0, useUnknownForAnchor: False
Saved: s18_11.pt loss: 0.091085374 batch size: 128
threshold score: 0.143, true positive rate: 0.0006



(a) NNET algorithm, b18test_11

frameLagForNegative: 2
batchSize: 128, tripletLossMargin: 1.0
matchPairScoreThreshold: 150.0
epochs: 100, learningRate: 0.0001, newMean: 0.0
newStd: 1.0, useUnknownForAnchor: False
Saved: s18_12.pt loss: 0.011355658 batch size: 128
threshold score: 0.063, true positive rate: 0.00



(b) NNET algorithm, b18test_12

frameLagForNegative: 2

batchSize: 128, tripletLossMargin: 1.0

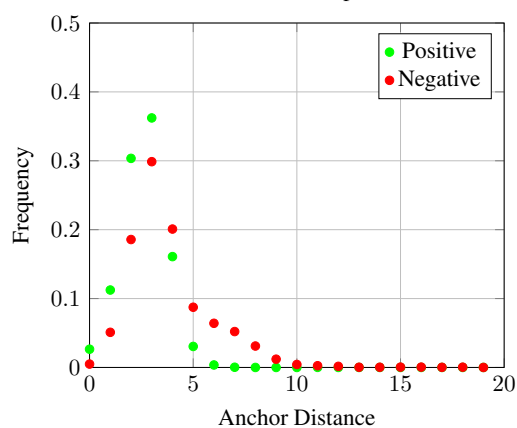
matchPairScoreThreshold: 150.0

epochs: 50, learningRate: 0.0001, newMean: 0.0

newStd: 1.0, useUnknownForAnchor: True

Saved: s18_13.pt loss: 0.046078812 batch size: 128

threshold score: 0.000, true positive rate: 0.01



(a) NNET algorithm, b18test_13

frameLagForNegative: 2

batchSize: 128, tripletLossMargin: 1.0

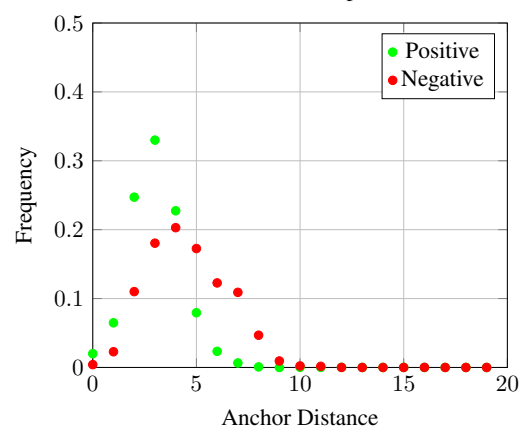
matchPairScoreThreshold: 0.456

epochs: 50, learningRate: 0.0001, newMean: 0.0

newStd: 0.224, useUnknownForAnchor: False

Saved: s18_14.pt loss: 0.0009985929 batch size: 128

threshold score: 0.086, true positive rate: 0.00



(b) NNET algorithm, b18test_14

APPENDIX C





The Sports Dataset

The sports dataset used in this project consists of around 22 hours of 8 separate sports videos. These videos are split into 5 minutes clips creating 263 reference videos. To create simulated 'unknowns' for testing, these reference videos are distorted in the 31 ways described in this appendix. The videos are distorted by the ffmpeg commands shown below. Following the ffmpeg commands are example images taken from the distorted videos.

Code	Distortion	ffmpeg Command
a1	Color change 1	ffmpeg -y -i <input file> -vf eq=brightness=0.06:saturation=2 -c:a copy <output file>
a2	Color change 2	ffmpeg -y -i <input file> -vf eq=contrast=1.5 -c:a copy <output file>
b1	Even cropping 1	ffmpeg -y -i <input file> -filter:v "crop=(3*in_w)/4:in_h:in_w/8:0" -c:a copy <output file>
b2	Even cropping 2	ffmpeg -y -i <input file> -filter:v "crop=3*in_w:(3*in_h)/4:0:in_h/8" -c:a copy <output file>
b3	Even cropping 3	ffmpeg -y -i <input file> -filter:v "crop=(3*in_w)/4:(3*in_h)/4:in_w/8:in_h/8" -c:a copy <output file>
c1	Uneven cropping 1	ffmpeg -y -i <input file> -filter:v "crop=(3*in_w)/4:in_h:0:0" -c:a copy <output file>
c2	Uneven cropping 2	ffmpeg -y -i <input file> -filter:v "crop=(3*in_w)/4:in_h:in_w/4:0" -c:a copy <output file>
c3	Uneven cropping 3	ffmpeg -y -i <input file> -filter:v "crop=in_w:(3*in_h)/4:0:0" -c:a copy <output file>
c4	Uneven cropping 4	ffmpeg -y -i <input file> -filter:v "crop=3*in_w:(3*in_h)/4:0:in_h/4" -c:a copy <output file>
c5	Uneven cropping 5	ffmpeg -y -i <input file> -filter:v "crop=(3*in_w)/4:(3*in_h)/4:0:0" -c:a copy <output file>
c6	Uneven cropping 6	ffmpeg -y -i <input file> -filter:v "crop=(3*in_w)/4:(3*in_h)/4:in_w/4:in_h/4" -c:a copy <output file>
d1	Reduced frame size	ffmpeg -y -i <input file> -vf scale=iw/2:ih/2 -c:a copy <output file>
e1	Frame rate reduction 1	ffmpeg -y -i <input file> -r 15 -c:a copy <output file>
e2	Frame rate reduction 2	ffmpeg -y -i <input file> -r 10 -c:a copy <output file>

Code	Distortion	ffmpeg Command
f1	Add news ticker	ffmpeg -y -i <input file>-i <ticker file>-filter_complex “[0:0][1:0] overlay=x=0:y=main_h-overlay_h-10” -shortest -map 0:1 -c:a copy <output file>
g1	Add static watermark	ffmpeg -y -i <input file>-i <logo file>-filter_complex “overlay=x=main_w-overlay_w-10:y=main_h-overlay_h-10” <output file>
h1	Change aspect ratio	ffmpeg -y -i <input file>-vf scale=ih:iw -c:a copy <output file>
i1	Flip video vertical	ffmpeg -y -i <input file>-vf vflip -c:a copy <output file>
i2	Flip video horizontal	ffmpeg -y -i <input file>-vf hflip -c:a copy <output file>
j1	Rotate 10°	ffmpeg -y -i <input file>-vf “rotate = (1*PI)/18” -c:a copy <output file >
j2	Rotate 20°	ffmpeg -y -i <input file>-vf “rotate = (2*PI)/18” -c:a copy <output file>
j3	Rotate -10°	ffmpeg -y -i <input file>-vf “rotate = -(1*PI)/18” -c:a copy <output file>
j4	Rotate -20°	ffmpeg -y -i <input file>-vf “rotate = -(2*PI)/18” -c:a copy <output file>
k1	Moving frame PIP	ffmpeg -i <input file>-probesize 100M -analyzeduration 100M -i <input file> -filter_complex “[0]scale=768:432[2];[1]scale=384:216[3];[2][3]overlay=x=t*4;y=t*2” -c:a copy <output file>
l1	RHS slant perspective	ffmpeg -i <input file >-vf perspective=“0:0:W:-(H/4);0:H:W:((H*5)/4)” <output file >
l2	LHS slant perspective	ffmpeg -i <input file >-vf perspective=“0:-(H/4):W:0:0:((H*5)/4):W:H” <output file >

Code	Distortion	ffmpeg Command
m1	PIP half size	<pre>ffmpeg -i <input file > -i <input file > -filter_complex "[1]scale = iw/2:ih/2 [pip]; [0][pip] overlay=(main_w-overlay_w)/2:(main_h-overlay_h)/2" <output file ></pre>
m2	PIP third size	<pre>ffmpeg -i <input file > -i <input file > -filter_complex "[1]scale = iw/3:ih/3 [pip]; [0][pip] overlay=(main_w-overlay_w)/2:(main_h-overlay_h)/2" <output file ></pre>
m3	PIP quarter size	<pre>ffmpeg -i <input file > -i <input file > -filter_complex "[1]scale = iw/4:ih/4 [pip]; [0][pip] overlay=(main_w-overlay_w)/2:(main_h-overlay_h)/2" <output file ></pre>

Distortion Code	Distortion Type	Image
a1	color model change	
a2	contrast increase	
b1	rhs/lhs crop	
b2	top/bottom crop	

b3 rhs/lhs/top/bottom crop



c1 lhs crop



c2 rhs crop



c3 bottom crop



c4 top crop



c5 lhs/bottom crop



c6 rhs/top crop



d1 rescale (half size)



e1 15 fps



e2 10 fps



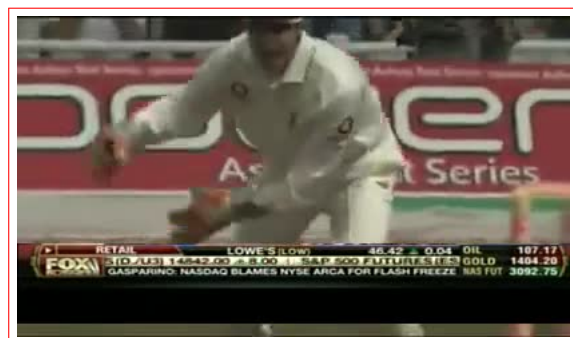
f1 ticker overlay 1



f2 ticker overlay 2



f3 ticker overlay 3



g1 static logo



h1 rescale (swap w:h)



i1 flip vertical



i2 flip horizontal



j1 rotate 10 degrees



j2 rotate 20 degrees



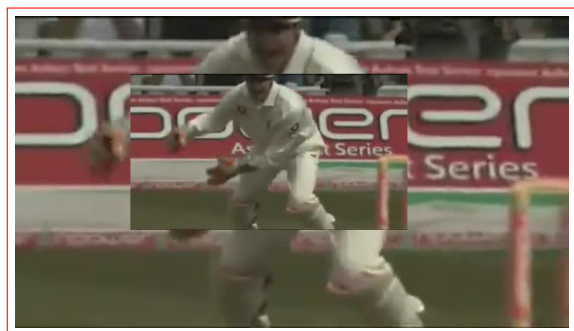
j3 rotate -10 degrees



j4 rotate -20 degrees



k1 moving frame pip



11 rhs slant perspective



12 lhs slant perspective



m1 pip half size



m2 pip third size



m3

pip quarter size



Table C.1: Examples of distorted images

APPENDIX D

The Feature Film Dataset

The feature film dataset used in this study consists of 118 full length feature films with a total duration of 178 hours (references). These feature films are provided to Audible Magic by the content owners in a degraded form. The video characteristics of this degraded form are a frame size of 128x72 pixels, 8 bit greyscale at 4 frames per second (fps). Only the degraded form is supplied to Audible Magic as the content owners have security concerns about the original definitive versions of the feature films.

The video clips to be found (unknowns) in the feature film dataset are short video clips from the feature films, generally between two and five minutes duration, culled from the Internet. These clips have an assortment of frames sizes, frame rates and colour models and are primarily used as advertising for the underlying feature films.

The screen shots D.1, D.2, D.3, D.4 D.5 are taken from an application (ViewVideoSidecar) written by the author that was used to time align the reference and unknown videos. The references are in greyscale on the left hand pane while the unknowns in colour are on the right hand pane.

These screen shots show scaling distortions, particularly in D.1 and D.2, clipping distortions in all these examples, as well as temporal distortions due to fast action sequences D.4.

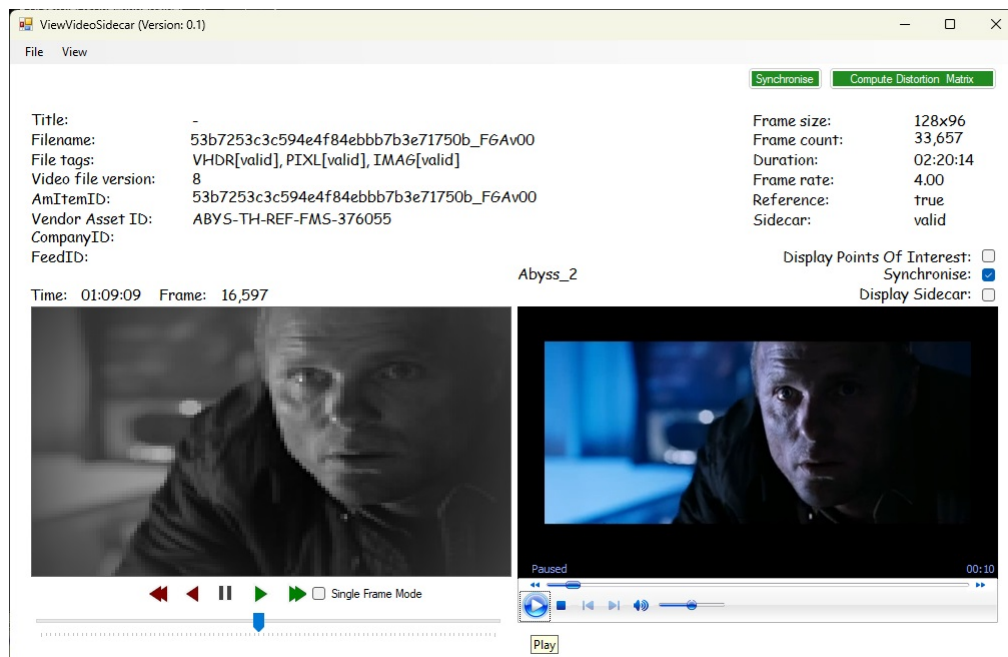


Figure D.1: View video sidecar application - The Abyss (1989)

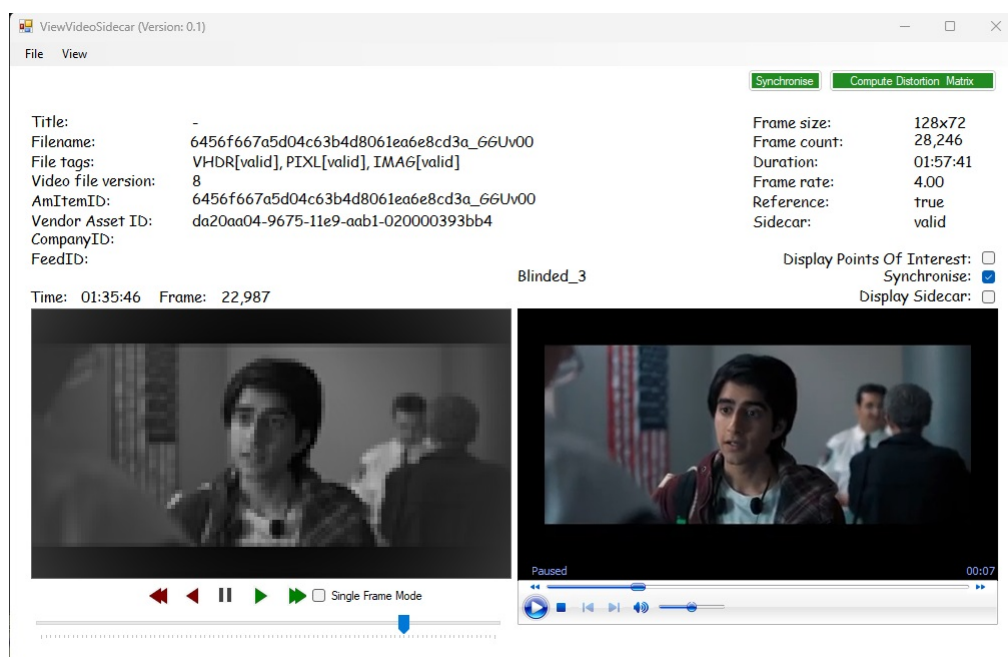


Figure D.2: View video sidecar application - Blinded by the Light (2019)

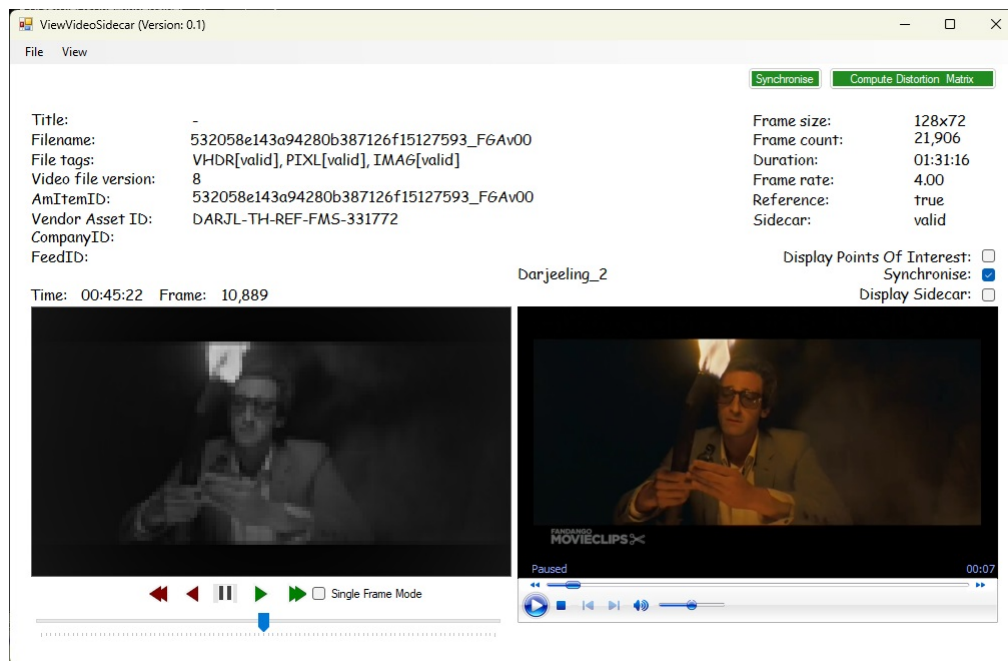


Figure D.3: View video sidecar application - The Darjeeling Limited (2007)

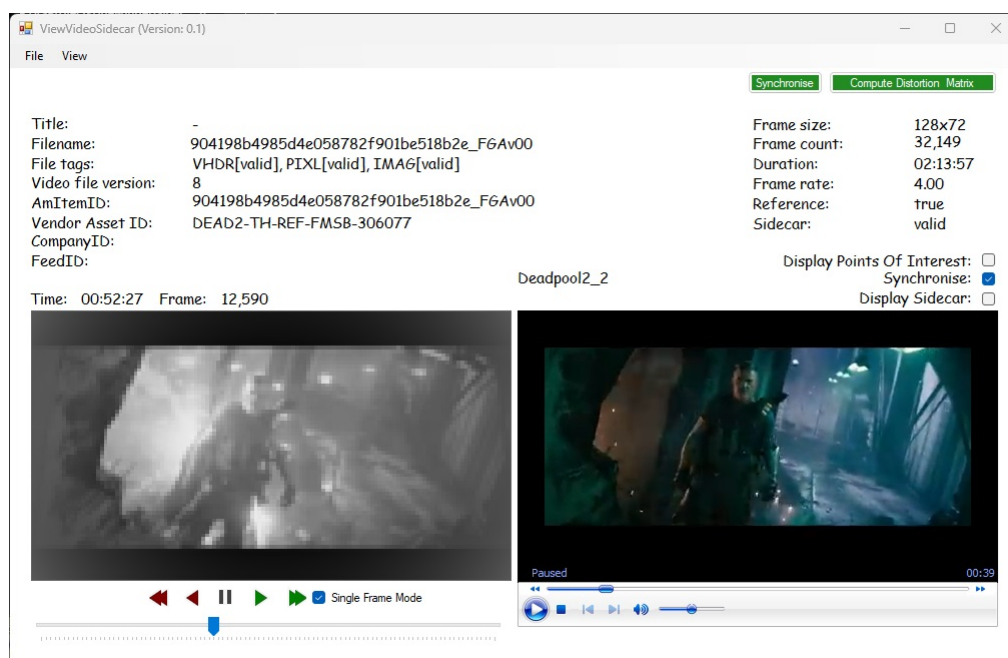


Figure D.4: View video sidecar application - Deadpool 2 (2018)

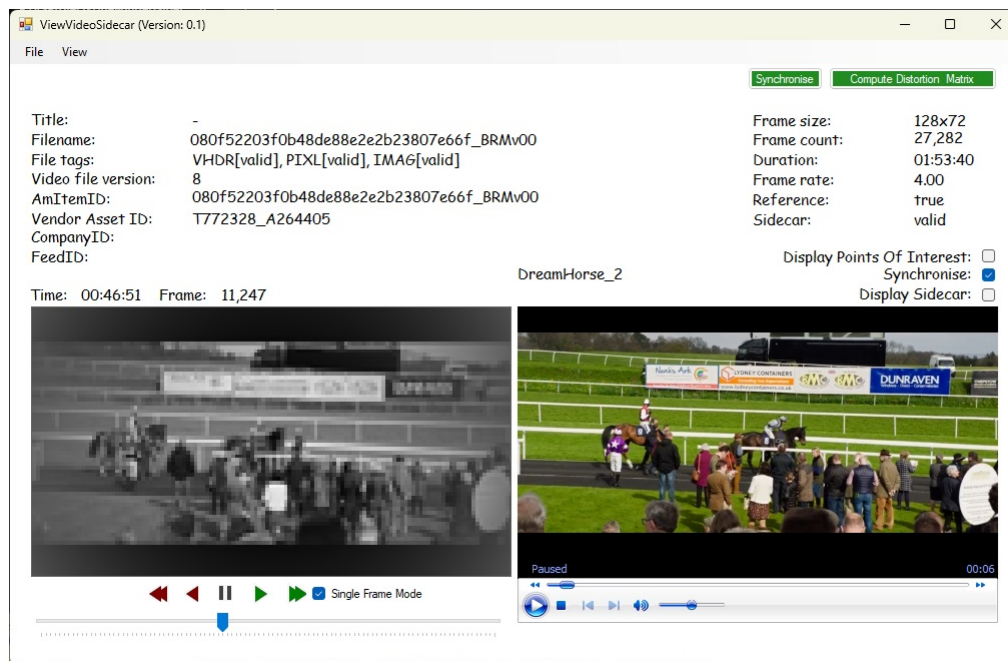


Figure D.5: View video sidecar application - Dream Horse (2020)