

Large Scale Pattern Detection in Videos and Images from the Wild

Craig Darren Mark Henderson

Submitted in partial fulfilment of the requirements of the Degree of
Doctor of Philosophy

School of Electronic Engineering and Computer Science
Queen Mary University of London
United Kingdom

January 2017

Statement of Originality


I, Craig Darren Mark Henderson, confirm that the research included within this thesis is my own work or that where it has been carried out in collaboration with, or supported by others, that this is duly acknowledged below and my contribution indicated. Previously published material is also acknowledged below.

I attest that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge break any UK law, infringe any third party's copyright or other Intellectual Property Right, or contain any confidential material.

I accept that the College has the right to use plagiarism detection software to check the electronic version of the thesis.

I confirm that this thesis has not been previously submitted for the award of a degree by this or any other university.

The copyright of this thesis rests with the author and no quotation from it or information derived from it may be published without the prior written consent of the author.

Signature: 
Date: January 12, 2017

Details of collaboration and publications:

1. C. Henderson and E. Izquierdo, (2017). "Scalable pattern retrieval from videos using a Random Forest index," In *International Conference on Internet of Things, Data and Cloud Computing* (ICC 2017). Cambridge, UK
2. C. Henderson and E. Izquierdo, (2016). "Feature Correspondence in Low Quality CCTV Videos," In L. Chen, S. Kapoor, & R. Bhatia (Eds.), *Emerging Trends and Advanced Technologies for Computational Intelligence* Vol. 647, pp. 261–281. Springer International Publishing.
3. C. Henderson and E. Izquierdo, (2016). "Symmetric stability of low level feature detectors," *Pattern Recognition Letters*, vol. 78, pp. 36–40, July 2016

4. C. Henderson and E. Izquierdo, (2016). "Robust feature matching in long-running poor-quality videos," In *IEEE Transactions on Circuits and Systems for Video Technology*, 26(6), 1161–1174, June 2016.
5. C. Henderson and E. Izquierdo, (2016). "Rethinking random Hough Forests for video database indexing and pattern search," *Computational Visual Media*, March 2016.
6. C. Henderson and E. Izquierdo, (2016). "Multi-scale reflection invariance," in *SAI Computing*, London, July 2016, pp. 420–425.
7. C. Henderson and E. Izquierdo, (2015). "Robust feature matching in the wild," in *2015 Science and Information Conference (SAI)*, 2015, pp. 628–637.
8. C. Henderson and E. Izquierdo, (2015). "Minimal Hough Forest training for pattern detection," in *22nd International Conference on Systems, Signals and Image Processing (IWSSIP)*, 2015, pp. 69–72.
9. C. Henderson, S. G. Blasi, F. Sobhani, and E. Izquierdo, (2015). "On the impurity of street-scene video footage," in *6th International Conference on Imaging for Crime Prevention and Detection (ICDP-15)*, 2015.
10. C. Henderson, and E. Izquierdo, (2014). "Large-scale forensic analysis of security images and videos". In *6th Doctoral Consortium at BMVC 2014*, Nottingham.

Acknowledgments

This work is funded by the European Union's Seventh Framework Programme, specific topic "*framework and tools for (semi-) automated exploitation of massive amounts of digital data for forensic purposes*", under grant agreement number 607480 (LASIE IP project).

I am grateful to my supervisor, Ebroul, for giving me the opportunity to work full time on my research with the award of a studentship for study in the Multimedia and Vision Research Laboratory at Queen Mary University of London. My application to study here was accompanied by a letter of recommendation from Judy Graham with whom I have had the pleasure to work in two companies so far, and hope to work with again some time in the future. Professor Alison Noble at University of Oxford provided a reference directly to QMUL. I am very grateful for your support, and I am sure it played no small part in my successful application.

I thank the Metropolitan Police at Scotland Yard, London, UK, for the supply of and permission to use CCTV images in my research, publications and this thesis. Particularly, recently retired Detective Chief Inspector Mick Neville of SCO4 Central Forensic Image Team, Richard Beckley at CFIT Training (MetCU) and the volunteers in the Central Forensic Image Team (CFIT) for their help and support.

The practical aspects of my research would have been a lot more complex and time consuming without the generosity of open-source projects of OpenCV, ffmpeg, Boost, and TeXStudio, and community minded software vendors who have made applications available for free use; Microsoft (Visual C++) and Mendeley. I am especially grateful to Juergen Gall et al. not only for research inspiring much of my own, but for making their source code available online for me to use as a basis for experimentation.

I am indebted to the many anonymous reviewers at peer-reviewed conferences and journals to which I have submitted papers, not always successfully. Their feedback and suggestions have inevitably improved the quality and clarity of my writing, as has the help of Stella Smyth at QMUL.

Finally, and perhaps most importantly, I thank my wife Kirstin for her unwavering love and support in all my personal ambitions.

Abstract

Pattern detection is a well-studied area of computer vision, but still current methods are unstable in images of poor quality. This thesis describes improvements over contemporary methods in the fast detection of unseen patterns in a large corpus of videos that vary tremendously in colour and texture definition, captured “in the wild” by mobile devices and surveillance cameras.

We focus on three key areas of this broad subject;

First, we identify consistency weaknesses in existing techniques of processing an image and its horizontally reflected (mirror) image. This is important in police investigations where subjects change their appearance to try to avoid recognition, and we propose that invariance to horizontal reflection should be more widely considered in image description and recognition tasks too. We observe online Deep Learning system behaviours in this respect, and provide a comprehensive assessment of 10 popular low level feature detectors.

Second, we develop simple and fast algorithms that combine to provide memory- and processing-efficient feature matching. These involve static scene elimination in the presence of noise and on-screen time indicators, a blur-sensitive feature detection that finds a greater number of corresponding features in images of varying sharpness, and a combinatorial texture and colour feature matching algorithm that matches features when either attribute may be poorly defined. A comprehensive evaluation is given, showing some improvements over existing feature correspondence methods.

Finally, we study random decision forests for pattern detection. A new method of indexing patterns in video sequences is devised and evaluated. We automatically label positive and negative image training data, reducing a task of unsupervised learning to one of supervised learning, and devise a node split function that is invariant to mirror reflection and rotation through 90 degree angles. A high dimensional vote accumulator encodes the hypothesis support, yielding implicit back-projection for pattern detection.

Table of Contents

1	Introduction	10
1.1	The impurity of street-scene video footage	11
1.2	Contributions	17
1.3	Thesis Organisation	18
2	Related work	20
2.1	Image features	20
2.2	Visual search	36
2.3	Shot Boundary Detection	37
2.4	Random Decision Forests	37
3	Asymmetric image analysis	41
3.1	Scale and orientation significance	42
3.2	Symmetric stability of low level feature detectors	43
3.3	Reflection Invariance in learned representation systems	53
3.4	Conclusion	57
4	Feature correspondence in poor quality images	59
4.1	Measuring image blur	59
4.2	Duplicate frames and static scenes	60
4.3	Blur sensitive feature detection	62
4.4	Combinatorial Texture and Colour feature matching	64
4.5	Evaluation	68
4.6	Conclusion	81
5	Random Forests for pattern indexing	82
5.1	Minimal training datasets	83
5.2	Random Forests for a video database index	91
5.3	Conclusion	113
6	Conclusion	115
6.1	Future work	117
	References	121

List of Figures

1.1	Comparing broadcast quality film definition with typical CCTV images . . .	12
1.2	CCTV image variations	13
1.3	Examples of video corruption in security videos	16
2.1	Example global feature detectors	21
2.2	Example feature channels; RGB and HSV	28
2.3	Distance ratio	34
2.4	Spatial Consistency	35
3.1	Reflection Invariance motivating example	42
3.2	Pyramid of Scales and Orientation significance	43
3.3	Bilateral Symmetry (mirror reflection) at different scales	44
3.4	Reflecting a keypoint	45
3.5	<i>Informative regions</i> of images and their mirror	53
3.6	Age guessing examples from Microsoft's How-Old.Net	55
4.1	Inactive video sequence with counting time stamp	61
4.2	Difference images of inactive video sequence with counting time stamp	61
4.3	Gaussian kernel size vs. image sharpness	63
4.4	Correspondence F_1 scores of composite features	67
4.5	Matching accuracy improvements of <i>blur sensitive feature detection</i>	70
4.6	Correspondence improvements of SIFT and SURF descriptors	72
4.7	Matching SURF features on a coloured bag from a query frame	73
4.8	<i>Good matches</i> of SURF features on a coloured bag from a query frame	74
4.9	Improvements of OpponentSIFT and OpponentSURF colour descriptors	75
4.10	Clarity of colour is sensitive to the distance between an object and camera	76
4.11	F_1 accuracy variation using only the Hue component on near-grey images	77
4.12	Comparing our F_1 score with state-of-the-art colour descriptors	78
4.13	Summary of results for our method	79
4.14	Correlation between descriptor size and matching accuracy	80
5.1	Pattern queries used in five videos in our experiments	83

5.2	Tracked regions are extracted before and after the query frame	85
5.3	Performance and precision change with the number of training images n^{\oplus} .	88
5.4	Precision variance with parameters affecting the size & shape of the forest .	89
5.5	Changes in the conventional use of random forests	91
5.6	The video database index is a collection of independent forests	92
5.7	Automatic labelling of positive and negative training patches	95
5.8	Back-projection of votes from high-dimension voting space to video frames .	101
5.9	Representative thumbnails of videos in the dataset	105
5.10	Query patterns for which ground truth annotations are provided	105
5.11	Precision/Recall curves for search patterns in our video corpus	109
5.12	Feature correspondence in low-quality images	110
5.13	Comparison of pattern detection using SIFT feature matching and our method	110
5.14	Result of searching for multiple patterns in one video using a single forest .	114
6.1	Distribution of feature channel contributions	118

List of Tables

2-A	Colour histogram descriptor dimensionalities	26
3-A	Keypoint detector metrics using the CALTECH101 dataset	49
3-B	Measurements of accuracy in the spatial domain	49
3-C	Measurements of accuracy based on SIFT feature descriptors	49
3-D	Predictions from Deep Learning Scene Recognition systems	54
3-E	Object recognition results from the <i>Wolfram Language Image Identity Project</i>	56
3-F	Feature detector invariance characteristics	57
4-A	Colour palette used in our experiments	70
5-A	Performance and accuracy training parameters of a Hough Forest	86
5-B	Parameter values used in our experiments	87
5-C	Experimental video dataset	87
5-D	Precision results of dynamic patch sizes	90
5-E	GLCM Entropy of pattern in reflection and rotation	99
5-F	Dataset video attributes and pattern occurrences	103
5-G	Key to the query patterns in the dataset	106
5-H	Detection complexity	112

List of Code Snippets

3.1	C++ code to assess reflection invariance of a Gaussian filter in OpenCV	52
-----	---	----

Chapter 1

Introduction

Pattern detection is a well studied and yet still unsolved area of Computer Vision research. Much progress has been made but still the methods are delicate, working with images of limited variety, captured in controlled environments, or with subjects that are segmented from the natural environment. General purpose pattern detection remains elusive despite years of research. *Visual Search* is an overarching term used to describe a search of a visual environment for a *target* among *distractors* [1]. It is a term that has increased in popularity and is now commonplace in contemporary literature describing systems of pattern search of a provided exemplar. The objective is well defined, and the rewards are high; provide the capability to find a previously unseen pattern within a corpus of images or videos, and show the user where the patterns (most likely) occur. This thesis focusses on query-by-example pattern detection in videos of low quality, observing and investigating limitations of current methods of image analysis and understanding (Chapter 3). We contribute to the body of literature on pattern description and matching with methods and techniques that can produce better results than have been achieved before (Chapter 4), in customarily small increments of improvement. Further, a more radical model of indexing pattern occurrence in videos is introduced (Chapter 5), which has demonstrated promise in natural video sequences captured *in the wild*.

The research has been performed with the specific focus of unconstrained pattern search and detection. Unconstrained in all meanings of the word; the environment in which the video is captured is uncontrolled – lighting, camera movement and weather all add complexity to the images – and videos are unedited and can be very long in duration. The sources of our images and videos are street-scene closed-circuit television, shop and private residences' security cameras, police body-mounted cameras and a plethora of mobile devices such as digital cameras, mobile phones and tablets. Our challenge is to be

able to find patterns such as corporate brands and logos, tattoos, clothing material and other distinctive markings on clothing, bags, vehicles and buildings in a large corpus of data. Our motivating use case is police investigations into criminal activity.

In large criminal investigations, police forces employ numerous officers and volunteers to watch many hours of camera footage to locate, identify and trace the movements of suspects, victims, witnesses, vehicles, luggage and other inanimate objects. Their goal is to piece together a story of events leading up to an incident, and to determine what happened afterwards. For example, the 2014 investigation by the Metropolitan Police (the *Met Police*) in London into the disappearance of school girl Alice Gross collected 8 days (8×24 hours) of continuous video camera footage from local authority street cameras along with footage for shorter durations obtained from private residences, shops and other businesses. In all, 30 Terabytes of video and image data were gathered. To assist investigations, the Met Police have looked to computer vision technology, but found existing systems to be severely limited in their ability to analyse real-world street-scene videos, because of the practical constraints in the variety of poor quality of videos.

1.1 The impurity of street-scene video footage

Research algorithms in the literature are typically demonstrated to work with high quality video. Two common examples are Hollywood movies *Groundhog Day* (Ramis, 1993) and *Run Lola Run* (Tykwer, 1998) used in [2–4] and subsequent comparative papers, and *Casablanca* (Curtiz, 1942) in [4]. These high quality videos have a high frame rate and good image resolution. The context in which street-scene videos are recorded differ in a number of significant ways from a feature film and produce scenes that are challenging to computer vision algorithms (Figure 1.1).

1.1.1 Long-running video sequences

Sivic and Zisserman [3] were the first to apply text retrieval theory and practices to video searching, and defined *visual words* to describe structure in images. The method tracks features across *shots*, a contiguous sequence of frames taken from a single camera within a scene. The number of frames and discovered features to process is manageable because of the relatively short period of time covered by a shot. The average shot length in feature films was 8-11 seconds before 1960 and had reduced to 4-6 seconds by 2006 [5]. Localised processing of feature movement is an aid to the algorithm by reducing the data volumes and providing a natural delineation of processing. Where necessary, cross-shot feature tracking can be considered at a later processing stage once features have been tracked within a shot. Boundary shot detection is well documented as an important prerequisite



Figure 1.1: Comparing the definition of broadcast quality films (top) with typical CCTV images. Top left, a frame from the 1993 film *Ground Hog Day* starring Bill Murray and Andie MacDowell; Top right, a frame from the 1998 German science-fiction action thriller *Run Lola Run* starring Franka Potente. Bottom row, scenes from CCTV footage have far less definition and clarity.

step to automatic video content analysis [6] as shots are regarded as the basic unit by which to organise the sequenced content of video and primitives [7].

Videos used in criminal investigations are very different. Fixed surveillance cameras fall into three categories; those that do not move and continuously record the same field of view, automated movement cameras that follow a defined motion such as a figure-of-eight to try to maximise area coverage, and human operated cameras that can be pivoted up and down, rotated around 360° and zoomed to varying depths¹. Each of these cameras produce a video consisting of a single shot that can last for hours. Body-mounted cameras and mobile phone footage also produce uninterrupted video sequences that can last several minutes, and hundreds of frames. Without a natural delineation of shot change, contemporary methods of object tracking and mining become less manageable, demanding large computing resource to process.

1.1.2 Camera movement

In static surveillance cameras the focal length and field of view are both fixed, and do not follow any activity. A car or a person that subsequently becomes of interest to police does not stay within shot, or even within focus. These fleeting glances can be important to

¹these cameras are called *PTZ*, reflecting their capability to *Pan*, *Tilt* and *Zoom*

an investigation but could easily be missed by reviewers scanning many hours of CCTV video.

An alternative to static cameras are those which passively record following a pre-defined motion path, with the camera mounted on a bracket that automatically moves around a loop or figure-of-eight. Objects will move in and out of view regularly within a sequence of frames. Other cameras are human-operated and can record very erratic movement with dramatic changes of focus and rapid zoom as the camera operator wrestles with the controls to record action on the streets. Individual frames can therefore be very blurred. The fast movement in pan and zoom, either in the manually controlled camera or to a lesser extent in a fixed-path motion camera degrades the image quality further, and is somewhat unique to the security videos such as those that we analyse.

1.1.3 Environmental

Security cameras record in uncontrolled environments. The footage is continuous, without any controlled change in focus, lighting and position. As a result, images have poor colour clarity and little discriminative or representative texture definition.

Many variations occur over a long-running video sequence. The sun changes through the day in position and intensity, and at night the scene changes to artificial ambient lighting and spot lighting from vehicle headlights, for example. The quality of images from each security camera therefore varies considerably, and this inconsistency can cause



Figure 1.2: Six images of a person wearing the same jacket, taken on different days with different street cameras, showing variation in texture and colour definition exhibited in street-scene videos from CCTV.

difficulties in finding correspondences in images even from the same camera (Figure 1.2).

Variations in weather over time cause very different images to be captured by a camera at different times. A change from sun to cloud affects the light intensity and colour definitions within the image. Rain or snow can appear as noise and even occlusions in extreme conditions. Fluctuating lighting conditions can also be caused by burning fire and by emergency vehicle lights, especially at night, and are commonplace in video that undergoes forensic analysis.

Closed-circuit television (CCTV) cameras are often sited very high and cover a long field of view where objects in the distance lack colour definition and texture clarity and can be difficult to identify even for a human. Fast camera movement pan or zoom, frenzied motion within a frame, or a combination of both can cause significant blurring in frame images which results in a lack of texture. Camera instability in free-hand or body-mounted cameras cause serious image blur and erratic movement.

1.1.4 Video acquisition and recapturing

The source of video footage used in a police investigation can be varied, as there is a lack of standardisation in CCTV systems. Obtained footage is often in a proprietary format that can only be viewed on-screen by a manufacturer-supplied application, and the flexibility and usability of these applications vary tremendously. To achieve their goal of forensic analysis and examination of segments of video, and to be able to edit videos into a *story* that can be used in a criminal court, the Met Police have employed creative solutions to overcome the limitations of the source video images. The result is a tedious and time- and resource-intensive activity to transcode the video footage by re-capturing the video as it is played on a computer screen. The resulting video file is in a standardised format that can be viewed and edited as required, and can also then be used in computer vision applications and research.

A consequence of the difficulty of acquisition is that the standardised video is often without meta-data which may have been useful, such as the video frame rate and date/-time stamps. These difficulties contrast with environments in most research which use Hollywood films with fast, and known, frame rates, high resolution images with consistent lighting, and where scenes are repeatedly re-shot until the quality meets an acceptable standard.

A further complication with the frame rate is introduced by the recapturing process. Re-capturing records at a fixed frame rate, perhaps 25 fps. If the video being played is at a lower frame rate, then multiple frames will be captured for each frame in the

original video file. The playback is visually unaffected, but the irregular duplication of consecutive frames adds another complication for computer vision applications as the amount of movement between pairs of adjacent frames is inconsistent. A second piece of meta-data is time sequence data. Time sequences would enable software to be able to synchronise video captured from multiple cameras, for example, based upon the time information associated with the video sequence. Edelman [8] reported on a system at the Netherlands Forensic Institute which uses Optical Character Recognition to read video timestamps from the video frame images. Such a technique is not reliable enough to provide sufficient meta-data for steering algorithms, however; the Met Police observe that camera timestamps are unreliable as the accuracy of the time is dependent on the ongoing maintenance of the CCTV system, and varies considerably between local authority, police and private owners of surveillance systems. Standard police procedure now is to record the actual current time and the presented CCTV time when a security video is seized for an investigation. This enables the police to calculate the offset of the CCTV time, but is fragile to the system clock having been altered since footage of interest was recorded.

1.1.5 Visual image quality

In contrast to Hollywood movies, CCTV cameras videos vary considerably in their frame rate and image resolution. Established methods of feature detection, extraction and matching perform less well on these videos than on high-definition images with sharp focus and controlled lighting conditions, as we describe in [9].

The frame rate of a video is measured by the number of frames per second, *fps*, that are recorded. With a lower frame rate, the time between frames is greater, features are further away relative to the previous frame and move greater distances relative to each other. Adjacent frames therefore have a greater visual difference than those from a high frame rate video. This difference can significantly affect the robustness of computer vision algorithms that often rely on the *a priori* knowledge that two adjacent frames in a video are very similar. As an example, a feature tracking algorithm makes the determination of whether features are related or not based on *spatial consistency* [3], which observes the similarity of the spatial arrangement in matched covariant regions of two images [10, 11]. In a low frame rate video, such determination becomes less robust as the movement threshold must be increased to compensate for the additional movement, and this can introduce noise and mis-classifications. It would be possible to configure spatial consistency algorithms using a video's meta-data, for example to adapt the spatial distance threshold of related features based on the frame rate of the video. In our area of interest, surveillance videos very often have no associated meta-

data, and cannot therefore be used as a reliable input into algorithmic choices for spatial consistency parameters.

Low frame rates reduce the number of images that make up the video sequence and a low resolution reduces the size of each video frame. Together these two attributes can significantly reduce the amount of storage required, and therefore the cost of storing the captured video and so are often reduced by organisations who seek to minimise the overhead of their security operations. The clarity of images from different security cameras also vary considerably, and this inconsistency can cause difficulties. Images are often low resolution with poor colour definition and have little discriminative or representative texture definition, and images from these need to be matched with those from higher definition images. Quality is further reduced by varying weather conditions where the changes in light, presence of rain, snow, mist or fog, direct sunlight and shadows can all affect the image, and the ability for a feature extractor to consistently describe an image region.

1.1.6 Video formats

There is no industry standard that defines video resolution, frame rate or file formats for CCTV security manufacturers. AVI is a commonly used container for storing videos, but most of the CCTV manufacturers are not compliant with encoding standards of file

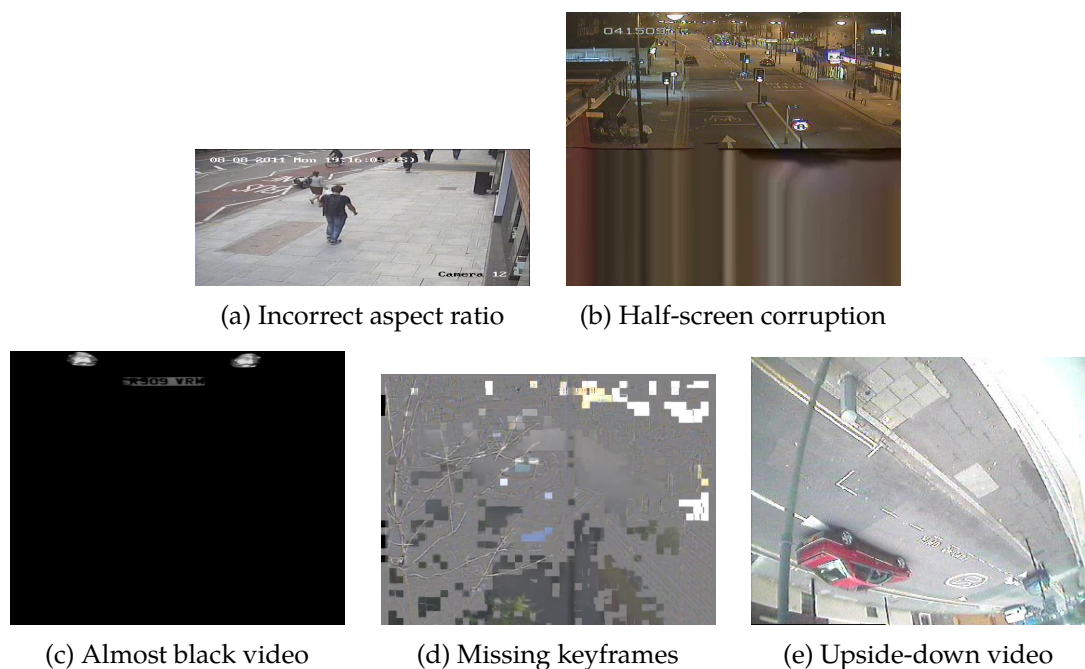


Figure 1.3: Examples of video corruption common in security videos from uncontrolled sources.

formats, encoding and compression, causing observable *corruption* when the video is played using standard players [12]. In 3.07 Tb of video data provided to us by the Met Police, only 929 Gb is in a non-proprietary format. Although these files purport to be Standard formats, viewing many of the videos using standard video players exhibit incorrect aspect ratio, half screen corruption, long sequences of blackness, missing keyframe or upside-down videos (Figure 1.3). This is because most CCTV manufacturers that we experienced in our study use proprietary video codec schemes [13] instead of relying on widespread video coding standards. Reasons for this were presented in our 2015 paper [12].

1.2 Contributions

The goal of this thesis is to assess methods of pattern detection that are robust in videos and images recorded in natural and uncontrolled environments – said to be *in the wild* – such as those that are used by police forces in criminal investigations. We contribute original thought and observations in the complexities of automated processing of videos from the wild (§1.1) and the asymmetric consistency of contemporary *Deep Learning* systems in reflection invariance (§3.3). The seminal work of Sivic and Zisserman [3] applied text retrieval methods to pattern search, and a large body of work has resulted using low-level features from image frames to match patterns from a query to the targets. We therefore begin our focus on such features and their effectiveness in our use case;

1. It is important for our work to be able to recognise patterns in many variations, including as a mirror image of itself. In §3.2 we begin our work with low-level features by conducting a comprehensive assessment of the stability of popular feature detectors in images and their mirror reflection. To our knowledge, this is the first study of its kind, and the results were published in *Pattern Recognition Letters*, 2016 [15]. We introduce five measurements of error that we show to be useful in determining the invariance to bilateral symmetry of a feature detector; *mean distance error*, *mean size error*, *mean angle error*, *mean descriptor distance error* and the *mean descriptor match error*. Further, we measure the accuracy of bilateral keypoint position, size and angle of orientation in an established dataset of 8,677 images and evaluate the capability of popular detectors to find consistent interest points.
2. To improve feature detection, we develop simple and fast algorithms that combine to provide memory- and processing-efficient feature matching. We demonstrate that these improve on current methods that use Euclidean distance to match intensity- and colour-feature descriptors. *Preprocessing to eliminate redundant information* – a method to robustly eliminate duplicate frames caused by video recapturing or

static frames of no activity, resilient to the presence of an on-screen timer, clock or frame counter. *Adaptive blur-sensitive feature detection*, an adaptive approach to the detection of features that will correspond between two images, guided by the sharpness of the two images. *Combinatorial Texture and Colour feature matching*, a novel technique to combine texture and colour features and measure distance between descriptors for robust feature matching.

3. We then study random decision forests, specifically *Hough Forests*, for pattern detection. First we perform an investigation to effective training with small training datasets, contributing (a) a means to *discover* training images from a single query region provided by the user, (b) an assessment of the impact of tuning standard Hough Forest parameters on the application of pattern detection using a very small training set of data, and (c) a novel method by which to improve runtime performance and precision in pattern detection using an adaptive patch size and calculated number of patches.
4. Finally, we arrive at a new method of pattern indexing in video sequences with contributions (a) to automatically label positive and negative image training data, reducing a task of unsupervised learning to one of supervised learning, (b) a random node split function that is optimised for image patch clustering and invariant to mirror reflection and rotation through 90° angles, (c) a high-dimensional vote accumulator that encodes the hypothesis *support* yielding implicit back-projection for pattern detection, and (d) multi-vote casting and area-based peak detection in voting subspace.

1.3 Thesis Organisation

Much of this thesis has previously been published in peer-reviewed Research Journals and UK Conferences, and as an invited book chapter as well as presentations and posters at internal Queen Mary University of London research dissemination events. Parts of §1.1 were presented, with colleagues, at the *6th International Conference on Imaging for Crime Detection and Prevention (ICDP) 2015* [12]. My sole contribution to this paper is contained in the thesis. Chapter 2 provides background reading for the subjects discussed and developed in the rest of the thesis and an overview of related work in these areas.

In Chapter 3, we observe reflection invariance in computer vision algorithms (presented in a position paper at *SAI Computing 2016* [14]), and present related work on the assessment of symmetric stability in popular low-level detectors (§3.2) that was published in Elsevier *Pattern Recognition Letters* [15].

Chapter 4 is dedicated to our work on feature correspondence in low quality videos, which was presented at the *Science and Information Conference 2015* [9] (winning the **Best Paper Award**). Extended work is published in *IEEE Transactions on Circuits and Systems for Video Technology* [16] and further as an invited Book Chapter in *Emerging Trends and Advanced Technologies for Computational Intelligence* published by Springer International Publishing, 2016 [17]. Early results were presented as posters at *BMVA Summer School 2014* and *Visual Image Interpretation in Humans and Machines (ViiHM)¹ Workshop 2015*.

In Chapter 5, we explore a new approach to building an index representation of a video that can be quickly searched for unseen patterns, and is effective in poor quality real-world videos. Early ideas with Decision Forests were presented at *22nd International Conference on Systems, Signals and Image Processing (IWSSIP) 2015* [18], and *4th International Conference on Computational Visual Media (CVM) 2016* [19]. A research paper describing the full method as presented herein is accepted for oral presentation in the *Data* track of *2nd International Conference on Internet of Things, Data and Cloud Computing (ICC 2017)* [20].

¹<http://www.viihm.org.uk/>

Chapter 2

Related work

In this chapter we position the research of the thesis within the context of recent and contemporary literature and provide some background theory as a foundation for the following chapters.

2.1 Image features

One of the most important questions in computer vision algorithms is that of how to represent images or parts of an image such that they can be separated and distinguished from each other better than in the image domain. Images are distilled from the spatial description of individual pixel colour or intensity to a more abstract notion of *features* that can be identified in an image (*detected*) and then described using (*extracted into*) a *feature descriptor*. The feature descriptor is typically of a high dimension, which can cause its own problems (§2.1.3), but are designed to capture the texture, and sometimes colour, of the image at a given point such that another descriptor extracted from another position, or from another image, from a visually similar image section will be located close-by in the high-dimensional feature space. Features can be extracted to describe the image as a whole (*global features*) or small regions of interest (*local features*).

2.1.1 Global features

Global features describe an image as a whole, such as a colour histogram (Figure 2.1a), and have been used effectively for whole image similarity [21, 22] providing a computationally light retrieval method. The distance between histograms can be efficiently calculated using the Earth Mover's Distance [23] or the normalised Histogram Intersection [21]. Furthermore, Stricker and Orengo [24] observed that most colour histograms are very

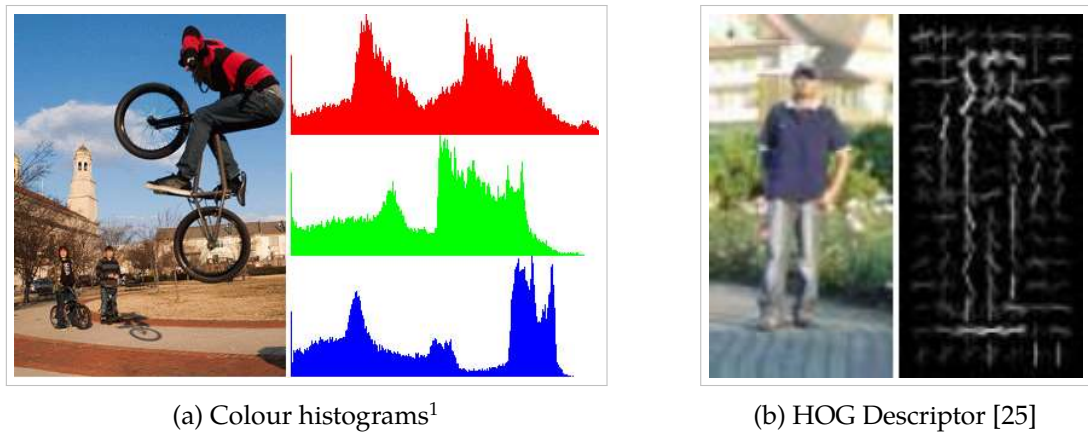


Figure 2.1: Example global feature detectors; colour descriptor, left, and texture descriptor, right.

sparse and therefore sensitive to noise, and so proposed using the cumulated colour histogram.

In images of less distinct colour, global texture descriptors have been used as an effective representation of images describing edges found in the image. Earlier descriptors based on the Haar wavelet transform such as that described in [26] have subsequently been out performed by the Histogram of Oriented Gradients (HOG) descriptor, demonstrated by Dalal and Triggs for pedestrian detection (Figure 2.1b) [25].

2.1.2 Local features

Contrary to global features, *local features* describe a small area of the image around an interest point. Extraction of local features is typically divided into two stages. First, *feature detection* locates areas within an image that contain sufficient texture or colour information to be considered *interesting*. Second, *feature descriptors* are extracted from the detected area to provide a representation of the image at that position.

Feature detectors

Local features are centred on a pixel in the image that is somehow different from its neighbourhood, and are consequently referred to as *keypoints* or *interest points*. A common early choice in locating interest points was to detect corners in a grey-scale image. The still-popular *Harris Corner Detector* published in 1988 [27] is a combined corner and edge detector based on the local auto-correlation function, derived from the earlier work of Moravec [28]. The detector was extended by Shi and Tomasi [29] with a small modification

¹http://billmill.org/the_histogram.html

to create the *Good Features to Track* (GFTT) detector that yields corners that are more likely to remain stable between consecutive frames of a video sequence.

Prior to 2002, interest point detectors concentrated on the spatial position of the interest point, then Mikolajczyk and Schmid [30] described an extension to the Harris corner detector that was robust against scale and affine transformations (out-of-plane rotation leading to shape deformation). In 2004, Lowe published a new interest point detection technique that, while not affine-invariant, was invariant to image scale and rotation and partially invariant to illumination, along with a descriptor that can describe the interest point in a scale-invariant way [31]. The *Scale-Invariant Feature Transform* (SIFT) applies the Difference-of-Gaussians operator at multiple scales and interest points are determined at the minima and maxima. It is accepted to be one of the most effective general-purpose detectors.

Whilst SIFT changed the field of feature detection and description – and is still widely used today – it is not without limitations. The SIFT detector is computationally expensive, and feature detection on large images can take too long to be used in real-time or large scalable systems. *Speeded-Up Robust Features* (SURF) [32] is a SIFT-inspired scale- and rotation-invariant detector and descriptor [32] that is more robust than SIFT in the presence of noise and less computationally expensive. The SURF method is based on a Hessian matrix and has good performance in computation time and accuracy. To reduce computation time, SURF's *Fast Hessian* detector uses integral images [33] which allow for the fast implementation of a box type convolution filter. The entry of an integral image $I_{\Sigma}(\mathbf{x})$ at a location $\mathbf{x} = (x, y)$ represents the sum of all pixels in the input image I of a rectangular region formed by the point \mathbf{x} and the origin; $I_{\Sigma}(\mathbf{x}) = \sum_{i=0}^{x-1} \sum_{j=0}^{y-1} I(i, j)$. With I_{Σ} calculated, it only takes four additions to calculate the sum of the intensities over any upright rectangular area, independent of its size [34].

Rosten and Drummond [35] learn a ternary decision tree that can detect points with high repeatability, to create FAST; *Features from Accelerated Segment Test*. The *Binary Robust Invariant Scalable Keypoints* (BRISK) detector [36] extends FAST with *an assembly of a bit-string* descriptor from intensity comparisons retrieved by dedicated sampling of each keypoint neighbourhood. ORB [37], is also based on the FAST detector from where the name is derived; *Oriented FAST and Rotated BRIEF*, where *Binary Robust Elementary Independent Features* (BRIEF) [38] is a feature descriptor. CenSurE [39] is described as a fast variant of the upright SURF descriptor, and sometime called STAR.

Some researchers consider interest points to be an irregular region of the image, rather than a single pixel keypoint. Maximally Stable (MS) regions, for example, are discovered using a common *Maximally Stable Extremal Region* (MSER) detection algo-

rithm [40, 41]. MSER is accepted to be a reliably effective and computationally efficient method of detecting feature regions in single channel images. Early work to extend MSER to multi-channel colour images was presented in [42] but did not achieve bottom up feature detection as in [43] where the author presents a derivative work specifically for *maximally stable colour regions*, MSCR. Cheng et al. [44] combine MSCR and a colour histogram to create an ID signature, and define a distance calculation to match IDs using a Bhattacharyya distance [45] between the histograms and the distance between MSCRs.

Symmetric stability Tuytelaars and Mikolajczyk [46] identified repeatability to be an important property of an interest point detector, meaning that two similar images of the same scene should yield a high number of corresponding interest points. In Chapter 3, we extend this thinking by proposing another important invariance property; that of *reflection invariance*. Although not the first to consider such a property, our assessment of popular interest point and region detectors with respect to it [15] is unique. In a recent work, [47] assessed object part localization and observed that the state-of-the-art methods augment the training set with mirrored images, but they did not result in bilaterally symmetric results. The authors introduced the term *mirrorability* and a *mirror error* that correlated with localization errors in human pose estimation and face alignment.

Feature descriptors

The orientation of a SIFT interest point is computed by extracting the gradient magnitude and direction of the neighbourhood, thus yielding the descriptors invariant to rotation as well as scale. However, they are sensitive to distortion [48], and the 128-dimension size of the descriptors is a drawback for using the method in large systems. However its accuracy has lent itself to widespread adoption and many descriptors have been presented that are based on SIFT and extend its capability in many different ways. Morel and Yu introduced an affine invariant version, called ASIFT [49] to overcome robustness to distortion, and to address the problem of the size of the descriptor, Ke and Sukthankar [50] applied PCA on the gradient image. The result is a 36-dimension descriptor that is fast for matching, but proved to be less distinctive than SIFT in a comparative study by Mikolajczyk et al. [51]. *Gradient Location and Orientation Histogram* (GLOH) [51] showed to be more distinctive than SIFT with the same number of dimensions, but is even more computationally expensive. Many other SIFT-based descriptors have been developed, and described by many. Examples include Local Contrast magnitude (LC-SIFT) [52] and Differential Excitation magnitude (DE-SIFT) [53], gradient magnitude approaches such as Orientation Restricted (OR-SIFT) [54], Gradient Orientation Modification (GOM-SIFT) [55], and Weber Local Descriptor (WLD) [53]. Robustness against non-linear intensity changes between

multi-spectral images was the goal of [56] who introduced NG-SIFT using *Normalised Gradients*. However this descriptor does not perform well on textured scene images, mainly due to binary nature of the NG features [57]. Later in the paper, the authors proposed MN-SIFT as a modification, based on *Modified Normalised* gradients. Another popular technique aside from the SIFT-based approach is that of *Local Binary Patterns*, of which *Centre Symmetric Local Binary Patterns* (CS-LBP) [58] and *Local Binary Pattern of Gradients* (LBPG) [59] are two examples.

RootSIFT SIFT descriptors, which are histograms, were designed for use with Euclidean distance measures for comparison and matching [31]. However, it is well known that using Euclidean distance to compare histograms often yields inferior performance than χ^2 or Hellinger measures. Arandjelović and Zisserman embraced this observation and proposed RootSIFT [60], which transforms the SIFT descriptor such that the Euclidean distance between two descriptors is equivalent to using the Hellinger kernel, also known as Bhattacharyya's coefficient. RootSIFT is dubbed *Hellinger distance for SIFT*, and can yield a significantly more accurate result in calculating the distance between two descriptors used in feature descriptor matching.

The Euclidean distance $L_2(\vec{p}, \vec{q})$ between two n -dimension vectors is given by

$$\begin{aligned} L_2(\vec{p}, \vec{q}) &= \|\vec{p} - \vec{q}\|_2 \\ &= \sqrt{\sum_{i=1}^n (p_i - q_i)^2} \end{aligned} \quad (2.1)$$

and is related to the similarity (kernel) $S_E(\vec{p}, \vec{q}) = \vec{p}^\top \vec{q}$, thus

$$\begin{aligned} L_2(\vec{p}, \vec{q})^2 &= \|\vec{p} - \vec{q}\|_2^2 \\ &= \|\vec{p}\|_2^2 + \|\vec{q}\|_2^2 - 2\vec{p}^\top \vec{q} \end{aligned} \quad (2.2)$$

When \vec{p} and \vec{q} are normalised to unit length $\|\vec{p}\|_2^2 = \|\vec{q}\|_2^2 = 1$, then this simplifies thus

$$\begin{aligned} L_2(\vec{p}, \vec{q})^2 &= 2 - 2\vec{p}^\top \vec{q} \\ &= 2 - 2S_E(\vec{p}, \vec{q}) \end{aligned} \quad (2.3)$$

Observing that the Hellinger kernel (Bhattacharyya's coefficient) for two L_1 normalised histograms is defined as

$$H(\vec{p}, \vec{q}) = \sum_{i=1}^n \sqrt{p_i q_i} \quad (2.4)$$

Arandjelović and Zisserman proceed to demonstrate that SIFT vectors can be compared

using a Hellinger kernel with two simple steps of algebraic manipulation: (i) L_1 normalise the SIFT vector, and (ii) square root each element of the vector. Since

$$S_E(\sqrt{\vec{p}}, \sqrt{\vec{q}}) = \sqrt{\vec{p}}^\top \sqrt{\vec{q}} = H(\vec{p}, \vec{q}) \quad (2.5)$$

and the resulting vectors are L_2 normalised, then

$$S_E(\sqrt{\vec{p}}, \sqrt{\vec{p}}) = \sum_{i=1}^n p_i = 1, \quad (2.6)$$

RootSIFT is then defined as an element-wise square root of the L_1 normalised SIFT vectors [60]. Their key observation is that comparing RootSIFT descriptors using Euclidean distance is equivalent to using the Hellinger kernel to compare the original SIFT vectors: $L_2(\sqrt{\vec{p}}, \sqrt{\vec{q}})^2 = 2 - 2H(\vec{p}, \vec{q})$

Colour feature descriptors

There have been a number of proposals for colour descriptors that describe colour attributes of an image. These are conveniently small in dimensionality (Table 2-A) and represent the colour information around a keypoint using a colour histogram. A detailed description of histogram based colour descriptors is provided in [61]. There have been many descriptors proposed to use colour and texture in combination, such as the biologically inspired SODOSIFT [63] and those that use texture descriptors in various colour channel combinations, concatenating the texture from each channel. Many colour descriptors aim to find and describe features found in three-channel colour images, based on the SIFT descriptor resulting in a large $128 \times 3 = 384$ dimension descriptor. HSV-SIFT [64] calculates a SIFT descriptor on each of the three channels in HSV colour space and RGB-SIFT [61] is a similar algorithm using RGB channels, with values equal to the *Transformed Colour SIFT* method [61]. rgSIFT [61] builds descriptors on the r and g chromacity components of the normalised RGB colour model

$$(r, g, b)^\top = \left(\frac{R}{R+G+B}, \frac{G}{R+G+B}, \frac{B}{R+G+B} \right)^\top \quad (2.7)$$

Biologically inspired *opponent colour* [62] describes two pairs of colours that cannot be seen in one location – red and green, and yellow and blue – explaining why humans don't see colours such as “bluish yellow” or “reddish green”. The three opponent colour channels, red-green (RG), yellow-blue (YB), and intensity (I) are calculated

$$(RG, YB, I)^\top = \left(\frac{R-G}{\sqrt{2}}, \frac{R+G-2B}{\sqrt{6}}, \frac{R+G+B}{\sqrt{3}} \right)^\top \quad (2.8)$$

Table 2-A: Colour histogram descriptors and their dimensionality. See [61] for a details of these descriptors and their properties.

Descriptor	Dimension
Normalised RG Histogram	30
Hue-Histogram	37
Opponent Histogram	45
RGB-Histogram	45
Transformed Colour Histogram	45

OpponentSIFT computes SIFT descriptors in each of the channels and OpponentSURF uses the same technique with SURF features. C-SIFT [65] uses a normalised opponent colour space, dividing the first two channels by the intensity channel I , $\frac{RG}{I}$ and $\frac{YB}{I}$, making it invariant with respect to light intensity [61]. A comprehensive review of colour descriptors is presented in [61].

In each of these, the descriptors are constructed from intensity descriptors that are extracted from colour channels and concatenated into a vector of $3 \times$ the dimension (one for each colour channel). These descriptors are implicitly discriminative by virtue of their construction, however, the colour detail of the image area around the feature is not encoded into the descriptor and is not used to discriminate between similar features. HueSIFT [66] describes a concatenation of a quantised Hue histogram of 37 dimensions with the SIFT descriptor, concentrating on the effective detection of features without consideration for the descriptor encoding. SIFT was also used as a base for the *bag-of-colours* algorithm in the context of image search in [67]. Our method presented in Chapter 4 takes a similar approach in descriptor concatenation, but we do not limit our focus on SIFT descriptors and we describe a robust approach to feature distance calculations.

The three fold increase in the descriptor dimension of these colour descriptors becomes problematic for efficient computation and storage at scale. Principal Component Analysis (PCA) [68] was used to reduce the dimensionality in PCA-SIFT [50], but is computationally expensive; given n data points, each represented by p features, the computational complexity is the sum of the covariance matrix computation $O(p^2n)$ and its eigen-value decomposition $O(p^3)$, hence $O(p^2n + p^3)$ [18]. A method of fast PCA calculation has been suggested [69], which finds the desired number of leading eigenvectors using less computation, but with a slightly larger mean-squared error.

In the same year that MSCR was described by Forssén [43], the use of colour and texture features were described [70] for use in an image retrieval system combining an MSCR detector with PCA-SIFT [50] and subsequently MSCR was reported to produce

a small but significant improvement in repeatability and an increase in the number of correspondences by more than 50% compared to MSER [71].

The recently introduced Fused-Colour GPHOG (FC-GPHOG) [72] builds on successive works on Histogram of Oriented Gradients (HOG) [25], Pyramid of HOG (PHOG) [73] and Gabor-PHOG (GPHOG) to create an integration of PCA features of GPHOG descriptors in six colour spaces to combine colour, shape, local and wavelet features into a single descriptor. The authors report an improvement over Pyramid of Histograms of Visual Words (PHOW) [74], Colour SIFT four Concentric Circles (C4CC) [75], Spatial Envelope [76] and Local Binary Patterns (LBP) [77].

Other methods include creating compact local descriptors using an approximate affine transform between image space and colour space [78] and an edge orientation difference histogram (EODH) feature descriptor, which is a rotation-invariant and scale-invariant feature representation [79], integrated with Colour-SIFT. Learning local feature descriptors has begun to attract research attention using Linear Discriminant Analysis [80] and Convex Optimisation [81], for example. The contribution of colour in matching video frames from multiple views was a motivator for the recent work of by Fezza et al. [82], using a Histogram Matching algorithm in a group of pictures considering pixels within a square window around each SIFT feature that is proportional to the scale parameter of the SIFT keypoint.

Object and Scene Recognition is an area of research that uses colour descriptors extensively [61, 83, 84]. The body of work concentrates on the influence of colour and intensity variance in local or global image representations based on a colour profile, typically using colour histograms. Histograms are built using grids of empirically-derived size, passed over the image in a sliding window approach. Dominant or average colours [85] in each position are tallied in histogram bins to build a profile of the appearance of the image. van de Sande et al. [61] provide a comprehensive review of colour descriptors and analysis of intensity and colour scale and shift invariance. In their evaluation of colour feature descriptors they conclude that OpponentSIFT is generally a better performing descriptor and is a good choice where there is no prior knowledge of the data set or object/scene categories. Other evaluations of feature descriptors in the literature are provided by Mikolajczyk and Schmid [51], and Vedaldi et al. [86] five years later.

Feature Channels

A *feature channel* is a representation of an image that is used to extract features of interest. They can be a simple colour component channel representing, for example, red, green and blue components of an RGB image or hue, saturation and intensity of an HSV



Figure 2.2: Feature channels; top row RGB channels, bottom row HSV colour space channels

image (Figure 2.2), or a result of more complex image processing to create an alternative representation, such as the HoG features shown in Figure 2.1b on page 21. HSV is a popular choice for colour channel representations, e.g. [87], because it is close to human vision [77], however the choice is not universal and many researchers choose others, or combinations of colour channels. Several papers in the literature [88–91] have described the use of the eight unique channels of three colour models RGB, YUV and YCbCr as feature channels. YUV and YCrCb share a similar luminance/brightness channel in V and Y respectively, so only one of these channels is used. Gray and Tao [89] extend the notion of feature channels further by distinguishing *colour channels* from *texture channels*. The colour channels are the unique channels of the colour models described previously, and the texture channels are created by convolving each of eight Gabor filters and thirteen Schmid filters with the luminance channel. They define parameters for the filters but admit the methodology was *haphazard* so the channels are unlikely to be optimal. Working in the HSV colour space, Shi et al. quantise colour into a 256 dimensional vector with 16 bins from H , and 4 bins from each S and V [70].

2.1.3 Dimensionality and Scale

An individual image can contain a large number of features that will be found by any individual detector. It is not uncommon to use a combination of detectors, for example a keypoint detector in conjunction with a region detector, to try to maximise the opportunity to identify salient areas of the image. An image can therefore contain a large number of features, often thousands, and each feature is represented by a descriptor of a fixed size. The popular SIFT descriptor is 128 dimensions, and colour variants can be up to $3 \times 128 = 384$ dimensions. The quantity and size of the image features becomes a

significant concern in managing more than a handful of images because memory capacity, processing time and storage constraints limit a system's capacity. For a system to be capable of processing a large number of images, each image representation needs to be compact but descriptive and efficient to compute and compare.

Reducing the number of features can easily be achieved through quantisation [3] where a cluster of similar vectors (close in feature space) are replaced with a vector of averaged values, representing *visual words*. Sivic and colleagues have published widely on their Video Google [3, 4, 93, 167] project, applying the principles of text-based search to object search and retrieval in images and videos. They devised a *Bag-of-Words* (BoW) implementation that has become known as Bag-of-Visual-Words (BoVW) [94, 95] and subsequently, Bag-of-Features [96, 97]. They use two region descriptors of monochrome images from the original videos; an elliptical shape adaptation about an interest point [30, 98] referred to as *Shape Adapted* (SA) features and an intensity watershed image segmentation [99] referred to as *Maximally Stable* (MS) features. Colour information was not used in the algorithms. Shape Adapted features use an affine invariant interest point detector [30] based on Harris corner detector [27] and a Gaussian scale-space [100]. Maximally Stable (MS) regions are discovered using a common *Maximally Stable Extremal Region* (MSER) detection algorithm [40, 41]. A decent introduction to BoVW and literature review is in [96] (another literature review is at [101], but the text is almost the same as [102] in part). BoVW centres on quantisation of feature descriptors, but quantisation can inevitably lose features during assignment. This can be somewhat overcome using a soft assignment of features to a weighted set of words [97, 103, 104].

An alternative feature reduction technique is Fisher Vectors [105, 106] which is often used to compress the representation of thousands of descriptors into a global image descriptor in visual classification and large scale image retrieval [107]. Perronnin and Dance applied Fisher kernels in the context of image classification and large scale image search [107] and their experimental results demonstrate that this choice significantly outperforms BoVW for the same size on most data sets. Moreover, it is cheaper to compute because fewer visual words are required. Four years later, Tao et al. [108] report "superior" results over a BoVW model using VLAD and Fisher vectors.

Fisher Vectors have also been used to encode local descriptors into a simple 7-dimensional vector encoding pixel co-ordinates, centre pixel intensity and first and second order derivatives (*Local Descriptors encoded by Fisher Vectors* (LDFV) [109]). The authors somewhat naïvely extend this to colour images by simply applying the same extraction on each of the H , S and V channels and concatenating into a single 21-dimensional descriptor and specifying the distances between two LDFV descriptors as the Euclidean distance. In the context of searching for objects in videos, Snoek et al. encode the colour

descriptors with the aid of difference coding using Fisher vectors with a Gaussian Mixture Model codebook [110]. Jiang et al. propose a fusion of Bag-of-Features (BoF) with Colour Moments and Wavelet Texture [111] and report a 40% improvement compared with BoF alone [97]. They highlight the importance of selecting an appropriate detector and soft-weighting scheme in quantisation and use an SVM learning kernel with a *Difference of Gaussians* (DoG) detector and χ^2 [112] radial basis function (RBF) kernel to attain their results. Experiments to reduce the size of the 128-dimensional SIFT descriptor using Principal Component Analysis (PCA) [68] resulted in a descriptor of 20 dimensions encoding the salient aspects of the intensity gradient in the feature point's neighbourhood [50]. Instead of using SIFT's smoothed weighted histograms, the authors applied PCA to the normalised gradient patch and demonstrate that their compact descriptors were more distinctive and more robust to image deformations than the standard SIFT representation.

A simplification of the Fisher kernel representation is the *Vector of Locally Aggregated Descriptors*, VLAD [113] where the dimension reduction and indexing algorithm are jointly optimised so that it best preserves the quality of vector comparison, yielding a significant improvement on the state of the art with search accuracy comparable to the bag-of-features approach for an image representation that fits in a very small dimensionality, e.g. 16 bytes per image. Improvements to VLAD [114] change the normalisation method to significantly improve retrieval performance and use vocabulary adaptation to alleviate problems caused when images are added to the dataset after initial vocabulary learning. Multiple Spatial VLAD (*MultiVLAD*) [114] extends further to allow the retrieval and localisation of objects that only extend over a small part of an image, without requiring use of the original image SIFT descriptors. MultiVLAD claims the new state-of-the-art over all benchmarks investigated therein for both mid-dimensional (20k-D to 30k-D) and small (128-D) descriptors [114].

As well as thinking about the dimensionality of individual descriptors, it is important to consider the quantity of descriptors for an image. Lowe [31] reported that reliable recognition is possible with as few as three features, with a typical image containing over 2,000 features. This suggests that it may not be the number of features that is critical to the success of a recognition and/or matching algorithm, but the saliency of the features [115], and that selective analysis of salient features may be a reasonable way to manage the volume of features in an image library. By understanding a feature's saliency [116], the features containing less information can perhaps be considered less important and be ignored through appropriate *feature selection* [117]. Burstiness is a property given to visual elements that appear more times in an image than a statistically independent model would predict [118]. Early detection and removal of these can reduce memory

and improve efficiency in image retrieval systems [119]. Identified features can be said to form a stop-list defining the most frequently occurring visual words as noise [3] that can be eliminated in feature matching and tracking. Meng et al. [120] report that the use of a stop-list removes the top 5% and bottom 10% of features from their 7Gb inverted index file, however the effectiveness of stop-lists has been questioned [121]. In similar thinking, Turcot and Lowe [122] describe an additional step to reducing memory requirements by selecting only a small sub-set of training features to use for recognition, based on the observation that many local features are unreliable or represent irrelevant clutter.

2.1.4 Feature matching

Robust feature matching is important for many tasks such as image alignment, stitching, object tracking, and search and retrieval. Typical methods for matching feature descriptors find the closest descriptor to another in n -dimensional space and use a threshold to determine if the descriptors are *close enough* to assume that they match.

In object tracking systems, matching features between adjacent frames is crucial, and many systems use keypoint features. Sun and Liu [123] describe a selective method of tracking in which they calculate a confidence of feature matches by two measures – a mean value of the maximum inner product for every descriptor, and a ratio of reliably matched features to the total number of features - and adapt the tracking algorithm appropriately. Fast keypoint detectors and corner detectors such as [124] enable use of keypoint tracking in real time systems by reducing the computation overhead [125], and real-time applications may employ probabilistic methods [126] for data association to discover matching consensus [36].

Distance measures

Feature matching is typically performed using a distance measure between descriptor vectors and methods such as approximate nearest neighbours, k -nearest neighbours, a radius match or randomised kd -trees [127] to find closest matches. Common distance measures between two vectors $\vec{p} = (p_1, p_2, \dots, p_n)$ and $\vec{q} = (q_1, q_2, \dots, q_n)$ are,

the L_1 , rectilinear (*taxicab*) distance

$$L_1(\vec{p}, \vec{q}) = \|\vec{p} - \vec{q}\|_1 = \sum_{i=1}^n |p_i - q_i| \quad (2.9)$$

the L_2 Euclidean distance see Equation (2.1) on page 24

the cosine distance

$$\text{cosine distance} = 1 - \frac{\vec{p} \cdot \vec{q}}{\|\vec{p}\| \cdot \|\vec{q}\|} = 1 - \frac{\sum_{i=1}^n p_i q_i}{\sqrt{\sum_{i=1}^n p_i^2} \sqrt{\sum_{i=1}^n q_i^2}} \quad (2.10)$$

the Mahalanobis distance [128, 129] is a measure of the distance between an observation \vec{p} and a distribution with mean $\vec{\mu} = (\mu_1, \mu_2, \dots, \mu_n)^\top$ and covariance matrix S ;

$$D_M(\vec{p}) = \sqrt{(\vec{p} - \vec{\mu})^\top (\vec{p} - \vec{\mu}) S^{-1}} \quad (2.11)$$

Two random vectors from the same distribution has a dissimilarity measure

$$d(\vec{p}, \vec{q}) = \sqrt{(\vec{p} - \vec{q})^\top (\vec{p} - \vec{q}) S^{-1}} \quad (2.12)$$

and, where the covariance matrix is the identity matrix, the Mahalanobis distance D_M reduces to the Euclidean distance L_2 . Further, if the covariance matrix is diagonal (all off-diagonal elements are zero), then D_M is the normalised Euclidean distance;

$$d(\vec{p}, \vec{q}) = \sqrt{\sum_{i=1}^N \frac{(p_i - q_i)^2}{s_i^2}} \quad (2.13)$$

Histograms Where vectors represent histograms with identical bins, their similarity can be calculated using the Normalised Histogram Intersection (NHI) [21],

$$I(\vec{p}, \vec{q}) = \frac{\sum_{j=1}^n \min(p_j, q_j)}{\sum_{j=1}^n p_j} \quad (2.14)$$

Subtracting the NHI from one gives a dissimilarity, which is a distance measure between two histograms

$$H(\vec{p}, \vec{q}) = 1 - \left(\frac{\sum_{j=1}^n \min(p_j, q_j)}{\sum_{j=1}^n p_j} \right) \quad (2.15)$$

For histograms that do not have identical bins, the Earth Mover's Distance [23] has shown to provide an effective measure and can be used as a coarse measure of image

similarity.

Distance ratio

A feature can be said to correspond to its closest match in a set of candidate features where the descriptor with the smallest distance is selected, irrespective of the value of the distance or its relationship to its neighbours. Lowe [31] refined this method using a *distance ratio* to determine if the closest match was a *good* match. The distance ratio method finds the closest two features f_c and f_{c+1} and divides the nearest distance by the second closest distance,

$$\text{distance ratio} = \frac{\|f - f_c\|_2}{\|f - f_{c+1}\|_2} \quad (2.16)$$

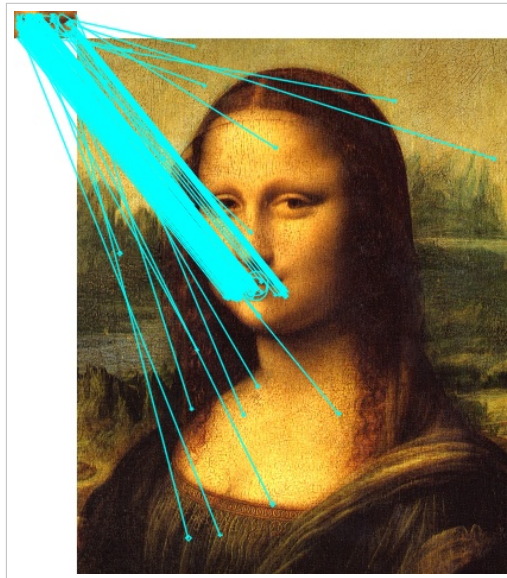
This ratio helps to determine how reliable the match is. If the nearest feature has another feature close to it, then there is a lesser likelihood that the match is correct. Tests in the original paper suggest that 0.8 is a reasonable threshold for this ratio and that matches with a distance ratio greater than 0.8 should be considered less reliable, thus,

$$\text{match} = \begin{cases} \text{true} & \text{if equation (2.16)} \leq 0.8 \\ \text{false} & \text{otherwise} \end{cases} \quad (2.17)$$

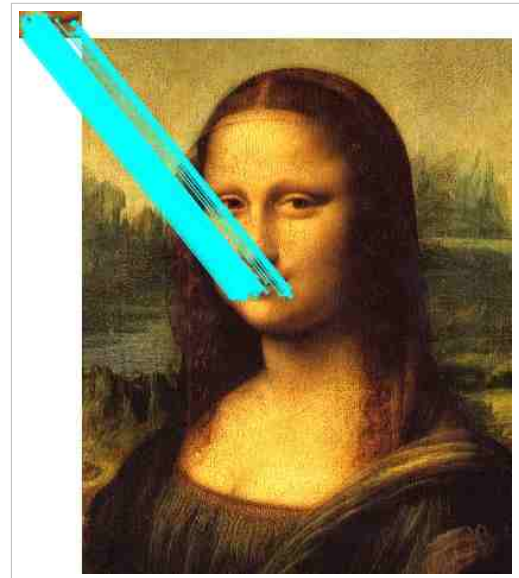
Lowe observed using a database of 40,000 keypoints that discarding matches with a distance ratio greater than 0.8 eliminates 90% of the false matches while discarding less than 5% of the correct matches [31] (Figure 2.3). The same statistics are somewhat misleadingly reported again by Shi and Yan in 2010 [71] as fact without citation, rather than an observation in their own experiments. The same authors used a *matching ratio* to measure accuracy of their algorithms, defined as $r = \frac{N_1}{N_2}$ where N_1 is the number of correct matches and N_2 is the number of features in the query image. However, feature matching accuracy is more typically measured using Precision-Recall [130].

Spatial consistency

A raw bag-of-words is not robust enough for object detection because there is no identification or localisation of objects to distinguish that a match of features represents a given shape of object [96]. However, a refinement of the descriptor-space using the spatial distance between features as a complementary measure can help. *Spatial Consistency* is a measure of neighbouring matches in the query region that lie in a surrounding area in the corresponding image [4]. A more strict measure requires that matching features have the same spatial layout in the two images. Figure 2.4 shows spatial consistency filtering



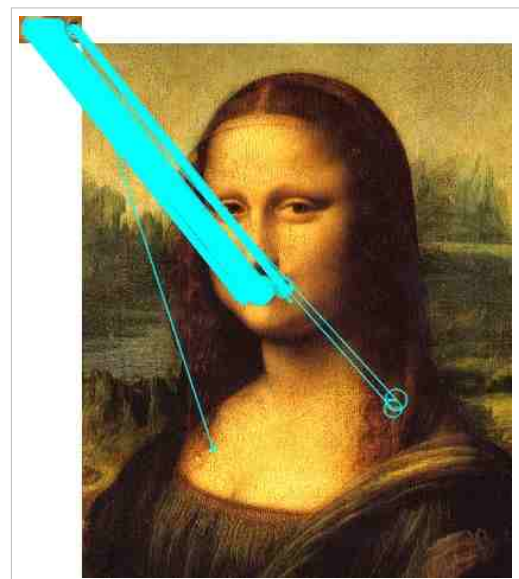
(a) Without distance ratio filtering, 146 SIFT features are matched from the query image; not all of these matches are correct



(b) 117 SIFT features matched using distance ratio filtering. All features are correctly matched

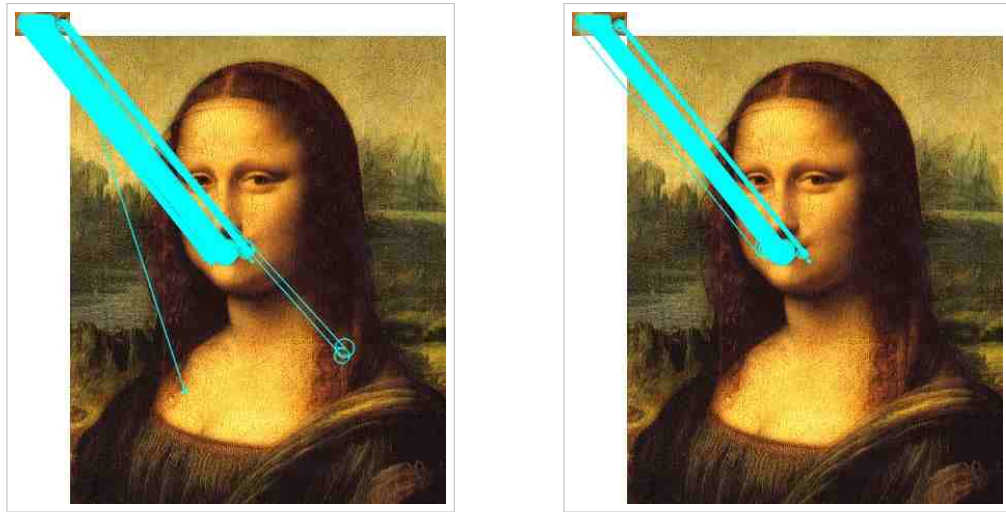


(c) 842 ASIFT features matched without distance ratio filtering



(d) 321 ASIFT features matched using distance ratio filtering

Figure 2.3: Matching SIFT feature descriptors. The query image in the top-left of each sub-figure is the Mona Lisa's famous smile. Blue lines track features matched from the query image to the high-resolution image (1927×2403) and their angular spread provide a visual indication of match accuracy. Matching features without distance ratio filtering yields SURF:156 matches (not shown), SIFT:146 matches (2.3a), ASIFT:842 matches (2.3c). Matching the same features using distance ratio filtering yields SURF:57 matches (not shown), SIFT:117 matches (2.3b), ASIFT:321 matches (2.3d).



(a) Matching *Distance Ratio* filtered ASIFT features without spatial consistency filtering (from Figure 2.3d)

(b) Matching the *Distance Ratio* filtered ASIFT features with spatial consistency filtering yields 215 matches

Figure 2.4: Spatial Consistency applied to *Distance Ratio* filtered ASIFT features in a high-resolution image reduces 321 matched features to 215 high quality matches.

applied to Figure 2.3d.

In the field of logo detection, Chu et al. [131] introduce encoding angles between features as relative headings to describe the spatial relationship between features, as do Romberg et al. in building a hierarchical database using a cascading index of edges and triangles [132]. Future work to build *visual language models* that describe distribution of visual words in images was proposed [94] but never published.

2.1.5 Indexing

Indexing provides a mechanism for a quick look-up of search data – in the case of image matching, feature descriptors – in a large collection of data. A popular method is the standard weighting, “term frequency-inverse document frequency” (*tf-idf*) [133], and is described in [134]; suppose a vocabulary of V words, then each document (image) is represented by a vector $\vec{v}_d = (t_1, \dots, t_{|V|})^\top$ of weighted word frequencies with components *word frequency* and *inverse document frequency*

$$t_i = \frac{n_{id}}{n_d} \log \left(\frac{N}{N_i} \right) \quad (2.18)$$

where n_{id} is the number of occurrences of word i in document d , n_d is the total number of words in document d , N_i is the number of documents containing the term i , and N is the number of documents in the database. At retrieval, documents are ranked by the

normalised scalar product (cosine of angle, equation (2.10)) between the query vector \vec{v}_q and all document vectors \vec{v}_d in the database.

tf-idf is the most commonly used indexing method in text- and image-based systems, such as in [3, 4, 95, 104, 134–138]. Nistér and Stewénus [121] developed break-through scalability and demonstrated image retrieval on a million-image data set [96], proposing a hierarchical *tf-idf* scoring using hierarchically defined visual words that form a vocabulary tree hierarchical quantisation that is built by hierarchical *k*-means clustering. Others have used alternatives, such as [139] who describe a bag-of-features image representation model combined with SVM classification in a Microsoft SQL Server database, and general randomised forests were used as part of a solution to index short video sequences using spatio-temporal interest points in [140], which is the closest contemporary research related to our work in Chapter 5 which we introduced in [19].

2.2 Visual search

While this thesis concentrates on pattern detection, our motivating use-case is query-by-example in the context of visual search. Most search and retrieval systems are object-class [84, 141] or object-instance [120, 142] search using learning algorithms that must be trained using a large set of ground truth images containing many positive and negative sample images. However, Chu et al. [131] developed a single image query system for logo recognition without training a model. They used *mean shift clustering* [143, 144] of SIFT [31] keypoints to select a candidate region, and verify the presence of the object in subsequent frames using a visual word histogram [3] describing appearance and their own *visual patterns* to provide spatial context. Meng et al. [120] use FLANN [127] to build a 250,000-word vocabulary of RootSIFT features from one keyframe extracted from every 30th frame, and Kalal et al. [145] present *Training-Learning-Detection*, a method to track unseen objects in a video stream from a single image query. The system learns the appearance of the object as it is tracked (inspiration for our work in §5.1), using *P-N Learning experts* that learn errors by uniquely identifying missed detections (P-experts) and false alarms (N-experts), and provide mutual compensation of their errors through independence.

Object search was recently (June 2015) described [146] using the familiar combination of BoVW, a stop-list and an inverted file for indexing, and the authors propose a spatial context using a *spatial random partition*. The authors claim benefits by aggregating the matching scores over multiple random patches to provide robust local matching and efficient object localisation in a pixelwise confidence map.

2.3 Shot Boundary Detection

The complexity of shot boundary detection techniques [6, 147] varies considerably. Wang and Qureshi [148], for example use an Embedded Hidden Markov model (EHMM) [149, 150] trained on GIST-like descriptors (GIST [76]), while Zhai and Shah [151] describe a statistical framework for the temporal scene segmentation of videos using Markov chain Monte Carlo (MCMC) [152], and Huang et al. 2008 [153] match local keypoints across frames. Anjulan and Canagarajah [2], shot boundaries are based on the number of *Local Invariant Regions* (LIR) matched between adjacent frames. A consistency measure is calculated, thus

$$CM = \frac{N_{uv}}{\max(N_u, N_v)} \quad (2.19)$$

where N_{uv} is the number of matched features descriptors in consecutive frames u and v . The value of CM is reported to vary significantly between inter-shot frames and intra-shot frames and the partition is sufficiently large that a threshold value can be chosen easily. The authors chose to fix their threshold based on their observations during experiments, but a learned approach that can adapt to the video sequence in process would be preferable.

In solving the problem of keyframe extraction, Porter et al. [154] represent frames in a shot and their correlations using a directed weighted graph. The shortest path in the graph is found and the vertices in the shortest path which correspond to the minimum correlation between frames designate the keyframes [155]. Others use clustering and identify a keyframe and those closest to the centre of each cluster [156–158], and Wang reported on motion projection of segmented regions between frames and subsequent region merging for adaptive tracking in the context of video segmentation [159].

2.4 Random Decision Forests

In Chapter 5 we will use Random (Decision) Forests [160] (Random Forests²) machine learning techniques to build a searchable index of patterns in video corpus. Random Forests are a well-studied and popular supervised learning method, and have been applied to a number of machine learning problems. Based on independent sets of binary trees, random forests are fast to train and scale well with increased volumes of training data [92]. The ensemble method constructs multiple diverse predictive models from

² The algorithm for inducing Breiman's random forest was developed by Leo Breiman and Adele Cutler, and "Random Forests" is their trademark (U.S. trademark registration number 3185828). The method combines Breiman's "bagging" idea and the random selection of features, introduced independently by Ho [161, 162] and Amit & Geman [163] in order to construct a collection of decision trees with controlled variance. [https://en.wikipedia.org/wiki/Random_forest]

adapted versions of the training data that correct for individual decision trees' habit of over-fitting to their training set [164] pp. 587–588.

2.4.1 Bootstrap Aggregation (Bagging)

Bootstrap aggregating, also called *bagging*, is designed to improve the stability and accuracy of machine learning ensemble algorithms, and is a key part of random decision forests, reducing variance and helping to avoid over-fitting. The training set is uniformly sampled *with replacement*; i.e. data will be duplicated in each sample. Drawing with replacement n' values out of a set of n (different and equally likely), the expected number of unique draws is $n(1 - e^{-n'/n})$ [165]. If $n = n'$, then for large n , $n(1 - \frac{1}{e}) \approx 63.2\%$ of draws are expected to be unique, and the rest duplicates.

2.4.2 Hough Forests

Hough Forests [92, 166] use a random forest framework that is trained to learn a mapping from densely-sampled D -dimensional feature *cuboids* to their corresponding votes in a Hough space $\mathcal{H} \subseteq \mathbb{R}^H$. The Hough space encodes the hypothesis $\mathbf{h}(c, \mathbf{x}, s)$ for an object belonging to class $c \in C$ centred on $\mathbf{x} \in \mathbb{R}^D$ and with size s . The term *cuboid* refers to a local image patch ($D = 2$) or video spatio-temporal neighbourhood ($D = 3$) depending on the task [166]. Since we are interested in image pattern detection in this thesis, we consider each video frame as a static image, with $D = 2$.

In work inspiring ours in Chapter 5, Gall and Lempitsky [92] extend feature channels (§2.1.2) to tease out nuanced artefacts, and create 32 feature channel images, J_1, \dots, J_{32} ; three colour channels of the Lab colour space, the absolute values of the first-order derivatives $\frac{\partial}{\partial x}$ and $\frac{\partial}{\partial y}$, the absolute values of the two second-order derivatives $\frac{\partial^2}{\partial x^2}$ and $\frac{\partial^2}{\partial y^2}$, and nine HOG [25] channels, obtained as the soft bin count of gradient orientations in a 5×5 neighbourhood around a pixel. The 16 resulting channels are further processed by applying the minimum and maximum filtration filters – also known as *erosion* and *dilation* filters, respectively – with 5×5 filter size, yielding 32 feature channels.

Each tree in the forest is trained with independently and randomly selected negative and positive training samples from the 32 features channels. The unit of data in a tree is a patch, \mathbf{v} and the appearance of a patch can be described as [92]

$$\mathbf{v}_i = (J_i^1, J_i^2, \dots, J_i^j) \quad (2.20)$$

where each J_i^j is a 16×16 feature channel image and $j = 32$ is the number of channels.

Each patch is a constant size in training and testing; 16×16 pixels is commonly used, including in this thesis. Patches are extracted from random positions and a randomly selected feature channel $a \in \{1, 2, \dots, 32\}$, in each training image. A forest is constructed using a set of patches $\mathcal{S} = \{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n\}$ of uniform size. Trees are grown using subsets of patches \mathcal{S}_i at each node i in the tree, starting with the full training set $\mathcal{S}_0 = \mathcal{S}$. A random split test function is applied to each patch in \mathcal{S}_i and the patch is assigned to subset ${}^L\mathcal{S}_i$ or ${}^R\mathcal{S}_i$ based on the result, such that $\mathcal{S}_i = {}^L\mathcal{S}_i \cup {}^R\mathcal{S}_i$ on conclusion. At each leaf node, the probability that a patch belongs to the foreground is calculated³

$$p_f = \frac{|\mathcal{S}_i^\oplus|}{|\mathcal{S}_i^\ominus| \frac{|\mathcal{S}_0^\oplus|}{|\mathcal{S}_0^\ominus|} + |\mathcal{S}_i^\oplus|} \quad (2.21)$$

where \mathcal{S}_i^\oplus and \mathcal{S}_i^\ominus represent positive and negative training patches respectively, at node i and $i = 0$ at the root of the tree.

Discussion

In this thesis we are concerned with the detection of unseen patterns in a large corpus of videos and images. Manually designed feature channels have been the foundation of computer vision since the very early explorations of trying to “understand” images by computer. Research have also begun to try to find more general ways to represent image patterns to reduce complexity and increase robustness and performance. Early attempts used combinatorial techniques. For example, short- and long- range tracking methods were used to define tracks (continuously matched feature descriptors) over a sequence of frames to mine objects [167] and again recently in [168] where *Temporally Aggregated Patch Set* (TAPS) was described using the *Temporally Coherent Detector* (TCD) [169]. Sivic and Zisserman [3] argue that averaging descriptors in a track improves signal-to-noise ratio, without evaluating the idea, but Araujo et al. [168] provide a justification for averaging and demonstrate a quantitative improvement in search quality.

In recent years, machine learning has become fashionable once again, particularly Deep Learning techniques – a development of artificial neural networks (ANNs) from the 1980s-1990s – where no manually designed features are used at all, and the algorithms *somehow* learn representations of the training images. Research is still ongoing to understand quite what goes on inside these hugely complex models, but the results are undeniably successful in a number of computer vision tasks, and others. Interestingly, the *learned features* that are produced in later layers of a Convolutional Neural

³<http://www.iai.uni-bonn.de/~gall/projects/houghforest/houghforest.html>

Network (CNN) have been demonstrated to be generic features and transferable to other domains [170–172].

This hugely important contemporary research area is very active in many vision tasks relevant to our interests, such as image description, captioning and understanding, and we are excited to see developments that may be applicable to query-on-demand. We focus our study in other areas of feature mining and learning from data that have longer established success in pattern detection and classification tasks, and that we can study specifically in the context of our complex use case of detection in low quality videos.

Chapter 3

Asymmetric image analysis

We observe that computer vision algorithms are sensitive to the reflection of an image as if looking in a mirror, and that this sensitivity has not received very much attention in the literature. A recent work assessed object part localization [47] and observed that the state-of-the-art methods augment the training set with mirrored images but do not result in bilaterally symmetric results. The authors introduced the term *mirrorability* and a *mirror error* that correlated with localisation errors in human pose estimation and face alignment.

In this chapter, we explore a property of *reflection invariance*, specifically studying horizontal reflection as an introduction to the concept, assessing popular low level feature detectors for their invariance in reflected images and observing contemporary high-level systems based on popular Deep Learning techniques. Just as scale invariance seeks to neutralise the size of a feature to avoid bias in scale, we propose *reflection invariance* to avoid bias in mirror reflection about an arbitrary axis. We suggest reflection invariance is an important property in designing and implementing algorithms, and as a metric for measuring their success. It is important that algorithms should be consistent in applications such as object recognition and scene classification, and we demonstrate that current state-of-the-art methods do not exhibit consistency when an image is reflected horizontally.

The importance of reflection invariance in CCTV pattern detection is perhaps not immediately obvious, but is demonstrated in the two images from London street CCTV cameras in Figure 3.1. A man wearing a hoodie exhibiting a Nike sportswear logo is later captured wearing the hoodie inside-out, with the Nike logo showing in reverse. In a wider context, low-cost, good quality cameras built into mobile phones and tablet computers has fuelled a proliferation of images and short videos made available on



Figure 3.1: Motivating example. Cropped frames from a CCTV camera capture images of a man wearing a Nike hoodie (left) and later in the video having turned the hoodie inside-out, showing the Nike logo in reverse.

websites and social media. Many devices now have multiple cameras; one on the back of the phone that shows a live image on the screen, and one on the front of the phone alongside the screen. The latter has sparked a phenomenon for *selfies* – images taken by the camera user of themselves, often with others or in a scene. The two cameras work differently, consistent the user’s intuitive expectations. Cameras on the back show the image as a conventional camera would, as if the user is looking through their device at a scene. Cameras on the front produce a reflected-image consistent with the user looking at themselves in a mirror. This satisfies their expectation of what the camera should show them on the live screen display alongside the camera lens. The explosion in selfie images has lead to a large number of mirror-reflected images and videos available online. As they become more prevalent, systems are increasingly in need of matching features between regular images and mirror images of the same subject.

3.1 Scale and orientation significance

Feature detectors fulfil the common need to identify interest points within an image. Information at these positions is extracted into a *descriptor* – a fixed length vector of numeric coefficients or a binary string – that can used to match similar features in applications such as image retrieval, alignment, stitching, and classification.

Low-level keypoint features describe a neighbourhood of a few pixels, where the co-location of pixel intensities is an important attribute used to describe the feature. Most feature descriptors, including the some of the most popular such as SIFT [31] and HoG [25], use the orientation of pixel gradients in a colour space or channel in some way to represent distinct feature characteristics. These algorithms are inherently sensitive to orientation, however others are sensitive only in practice, caused by poor implementation

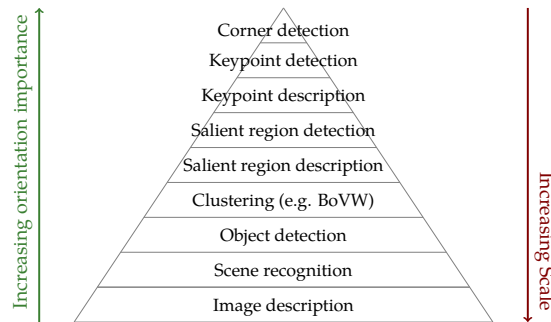


Figure 3.2: Pyramid of Scales and Orientation significance: as the scale increases, the importance of orientation diminishes

choices and mathematical rounding errors (§3.2.3) that accumulate to affect the result and cause dependence on image orientation.

A collection of descriptors can be composed to describe a distinctive pattern or region, such as in the popular *Bag of Visual Words* method [3]. In such a collection, the orientation of individual features *relative to each other* is important, but the orientation of the collection as a whole is less significant. As the scale of description increases further, orientation becomes less important and indeed becomes a limitation when considering high-level features in an image. The significance of orientation can therefore be considered inversely proportional to the scale of description, diminishing with the increase in distance from the pixel detail (Figure 3.2).

Reflection has the same scale of sensitivity as rotational-orientation. Consider an example of scene recognition. A human is likely to describe a city-scape scene, and identify a familiar city regardless of the horizontal reflection of the image; if the image is reflected about its vertical centre, this mirrored image would still be recognisable to a human and would not influence their description or identification. Computer vision algorithms are, however, more sensitive and often produce different results for these images, as we'll show later in Table 3-D on page 54.

The challenge is to generalize the description as the scale increases, with orientation becoming less relevant to the point where it is irrelevant at image scale. We first assess in detail the symmetric stability of low level feature detectors (§3.2) and then experiment with reflection invariance properties in high level learned systems (§3.3).

3.2 Symmetric stability of low level feature detectors

Most work on *bilateral symmetry* – a symmetry through a central vertical plane in an image or region – concentrate on the detection of symmetry [173] and accurate positioning

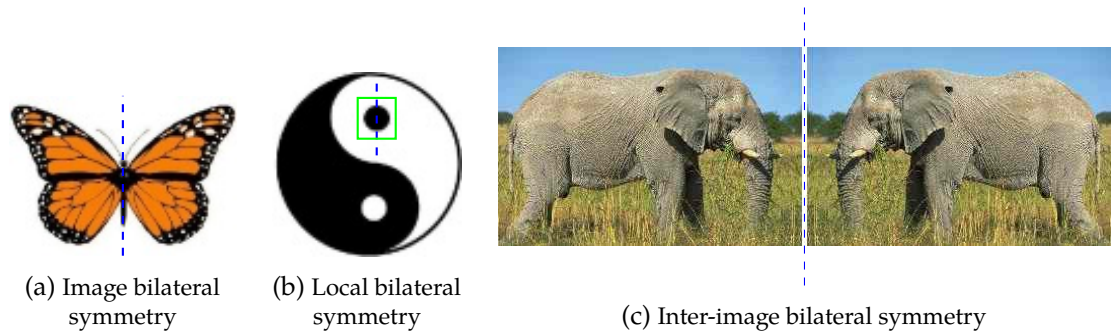


Figure 3.3: Bilateral Symmetry (mirror reflection) at different scales. (a) and (b) show areas within an image where there exists a local line of symmetry. Detection of the line of symmetry position is a large research area in itself. (c) shows our test case for assessing low level feature detectors, where we horizontally mirror the image to assess inter-image bilateral symmetry.

of the line of symmetry within a single image, for example [174–176]. Many reflection invariant feature descriptors have been proposed but none have corresponding reflection invariant feature *detectors*; Mirror reflection Invariant Feature Descriptor (MIFT) [177, 178], Mirror and Inversion invariant generalization for SIFT (MI-SIFT) [179], Flip-invariant SIFT (F-SIFT) [180], Flip INvariant Descriptor (FIND) [181], Flip Invariance Shape detector (FIS) [182], Max-SIFT [183], Mirror reflection invariant HOG descriptors [184], and the Fourier descriptor [185]. This is a serious omission because consistency in the detected position of a keypoint between an image and its horizontal reflection is important to enable a reflection invariant descriptor to be extracted from the same point in the logo and maximise the potential to achieve a correspondence.

Bilateral symmetry can occur at different scales, demonstrated with two examples from the CALTECH101 dataset [186] in Figure 3.3; (a) the image as a whole is bilaterally symmetrical because the right hand side of a vertical line drawn down the centre (the dotted blue line) is a mirror image of the left hand side and (b) the highlighted section of the image is bilaterally symmetrical although the image as a whole is not. Detected keypoints in an image are generally very small and detection of bilateral symmetry will be at a finer scale than both of these examples. Figure 3.3c shows our test case for assessing low level feature detectors, where we horizontally mirror the image to assess inter-image bilateral symmetry.

3.2.1 Method of assessment

General invariance properties of detectors and descriptors are important, and in work to date are consistent; an algorithm that provides for feature detection and feature description can provide invariance to scale, rotation, illumination or affine regions in both steps. Many research papers combine the two stages of detection and description

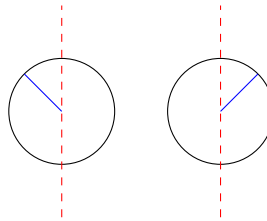


Figure 3.4: Reflecting a keypoint

into a single step, but each are independent.

The goal of feature detection is to find keypoints or regions in an image that contain *interesting* information. The definition of *interesting* is specific to the goal of the detector, but it is reasonable to expect that a location that is *interesting* in an image should also be *interesting* in the same image that is horizontally reflected.

To be reflection invariant, a feature detector must show that the set of keypoints or regions found in an image are equivalent to those found in the a mirror reflection of the image. The orientation of a feature is an important and discriminating attribute, and extracted descriptors should generally maintain local orientation so that established methods of feature matching, for example, can accurately measure the magnitude and position of a feature vector in high-dimensional space. Reflection invariance in low-level descriptors can be especially useful for detecting intra-image lines of symmetry, such as water reflections in scene analysis. Research has explored reflection-invariant HoG [184] and, more frequently, SIFT-based methods such as RIFT [187], MI-SIFT [179] and MIFT [178]. Generally, rotational invariance can be achieved by finding the dominant gradient and rotating the image patch so that the gradient is always in the same direction. RIFT, for example, divides normalized patches into four concentric rings of equal width, from each of which eight gradient orientation histograms are computed. The orientation is measured at each point relative to the direction pointing outward from the centre, thus maintaining rotation invariance.

A keypoint is defined by its (x,y) co-ordinates, size, and sometimes, angle of orientation (Figure 3.4). We reflect a keypoint in its centre line (red dotted line). Let I be the image in which the keypoint is found, and I^w be the width of the image. Let α be the angle of orientation, measured clockwise from 0° parallel to the x -axis. Then, the new values for the x position of the keypoint is x' and the new angle of orientation is α' , thus

$$x' = I^w - x - 1 \quad (3.1)$$

$$\alpha' = \pi - \text{atan2} \left(\frac{-\sin \alpha}{-\cos \alpha} \right) \quad (3.2)$$

where¹

$$\operatorname{atan2}\left(\frac{y}{x}\right) = \begin{cases} \arctan\left(\frac{y}{x}\right) & \text{if } x > 0, \\ \arctan\left(\frac{y}{x}\right) + \pi & \text{if } x < 0 \text{ and } y \geq 0, \\ \arctan\left(\frac{y}{x}\right) - \pi & \text{if } x < 0 \text{ and } y < 0, \\ +\frac{\pi}{2} & \text{if } x = 0 \text{ and } y > 0, \\ -\frac{\pi}{2} & \text{if } x = 0 \text{ and } y < 0. \end{cases} \quad (3.3)$$

Numerous detector methods have been described in the literature, and many have become popular for different tasks. Two distinct categories of feature detectors exist; *keypoint* detectors and *region detectors*. Recent trends in Deep Learning yield features that are discovered automatically during the training process that can be used in place of traditional features [170–172]. Here we concentrate on ten low-level feature detectors that are popular in contemporary literature of feature detection and can be described algorithmically (see §2.1.2), assessing how they perform in respect to reflection invariance; eight keypoint detectors BRISK, FAST, GFTT, HARRIS, ORB, SIFT, CenSurE, and SURF, and two region detectors MSER and MSCR.

3.2.2 Experiments and data

We assess the detectors using the well established CALTECH101 dataset [186]. The dataset consists of 8,677 JPEG images of approximately 300×200 pixels, grouped into 101 categories. Image are in a variety of styles including cartoons and photographs of objects, human faces, animals and natural scenes. Each category contains between 40 and 800 images, but most categories have about 50 images. MSCR is the only detector that works with 3-channel colour images and for all other detectors, the original colour images are first converted to single channel intensity images.

To measure the reflection invariance of the detectors, we use SIFT descriptors and measure their distance in feature space. Feature descriptors are themselves not invariance to bilateral symmetry and descriptors from an original image cannot be compared to a corresponding feature in a mirrored image. To overcome this, we extract feature descriptors from the original image using the detected keypoint attributes, and from reflected attributes detected in the mirror image. Let I represent an original image and M be the mirror of I . Then K_I and K_M represents keypoints detected in each of I and M respectively. Keypoints K_M are reflected as $k'_M \in K'_M$ using equations (3.1) and (3.2), and feature descriptors are extracted from I using K'_M .

¹atan2 is undefined if $x = 0$ and $y = 0$, but that case not possible for our use in Equation (3.2)

Our assessment is based on keypoint size and position. For features found by region detectors, we define a keypoint at the centre of the non-orthogonal (rotated) bounding rectangle of the region, and measure the size of the region as the encasing circle. We ran experiments across the entire dataset, counting the number of keypoints found in each of the images from the dataset (the *original image*) and in the horizontally reflected image of the original (the *mirror image*). Horizontal reflection was performed using basic pixel swapping without interpolation to ensure that no artefacts were introduced into the image data. Statistics were collected for each detector. Keypoints were matched using brute-force exact matching, based only on their (x,y) position in each image.

For each image I and its mirror M , the difference in the number of keypoints found were tallied per detector. If more keypoints were found in I , then the difference is accumulated in D_1^d otherwise the difference is accumulated in D_2^d , where d denotes the detector. Values of D_1^d and D_2^d then demonstrate the variability in the detectors and their inability to find even a consistent number of keypoints in an image and its mirror.

Of course, counting the number of keypoints alone is not sufficient to measure quality, so we proceed to quantify the accuracy of the detected keypoints in position, scale and orientation. We first measure accuracy based upon keypoint position. Keypoints $k_I \in K_I$ and $k_M \in K_M$ are spatially matched to their nearest neighbour and the sub-pixel distance between each matched pair is accumulated and divided by the total number of keypoints to establish *mean distance error* for keypoints for each detector.

At each keypoint $k_I \in K_I$, and reflected keypoint $k'_M \in K'_M$, a SIFT descriptor is extracted from image I , giving S_I and S'_M respectively. These descriptors are matched using a brute-force L_2 distance matching algorithm in 128-dimensional space, and their distances accumulated to compute a mean distance error per detector. This *mean descriptor distance error* measures the average error found in matching a SIFT descriptor extracted from the original image at an original keypoint location, size and orientation, with a SIFT descriptor extracted from the original image using the keypoint attributes found in the mirror image, adjusted by horizontal reflection.

Our final metric of reflection invariance is a *mean descriptor match error*. We compare the results of matching keypoints $k_I \in K_I$ with $k'_M \in K'_M$ using two methods;

1. common descriptor matching based on the L_2 distance between the SIFT descriptors in 128-dimension feature space, and
2. spatial matching of keypoint position in the (x, y) image co-ordinate space.

We count the number of keypoints that do not correspond identically using the two

methods and divide by the total number of matched keypoint pairs to give a mean value per descriptor. By comparing the results of the two keypoint matching strategies, we can determine an overall measure of how closely aligned the two sets of keypoints are, and therefore how robust the detector is to bilateral symmetry.

3.2.3 Results

Using the ten detectors, 41.78 million keypoints were found in the 8,677 original images (Table 3-A). Overall, 1,330 more keypoints were found in mirror images than in the original images, but this varied by descriptor. BRISK, SIFT, and MSER for example found 735, 644 and 437 more keypoints in the mirror images, but this represents only 0.05%, 0.02% and 0.06% increases. SURF found 494 fewer keypoints in the mirror image, 0.01%. Columns 4 and 5 of Table 3-A shows the number of keypoints found per detector, with highlights showing where the number of features is inconsistent in the original and reflected images.

Two detectors appear to perform well in handling bilateral symmetry of an image and its mirror. The FAST and CenSurE detectors both find an identical number of keypoints in every image, across all categories of the 8,677 image dataset. Both also have a zero *mean distance error* (Table 3-B column 2) indicating an exact match. GFTT and HARRIS have very small errors $< 10^{-6}$. The same protocol is followed to measure the mean error in the size of the keypoint (column 3) and the mean error in the angle of orientation of the keypoint (column 4). Four of the keypoint detectors do not define an angle of orientations; FAST, GFTT, HARRIS and CenSurE. For these detectors, there is no *mean angle error*.

In a keypoint detector that is perfectly invariant to bilateral symmetry, the *mean descriptor distance error* (Table 3-C column 2) value is 0.0, as is the case for four of the tested descriptors; FAST, GFTT, HARRIS and CenSurE. However, *mean descriptor distance error* of 0.0 alone cannot determine perfect invariance to bilateral symmetry. Our observations in Table 3-A tell us that the GFTT and HARRIS feature detectors produce a different number of keypoints in I and M , so while the descriptors can be matched with zero error, the spatial position of matched keypoints k_I and k_M have not been proven to be consistent.

The *mean descriptor match error* for each detector is shown in Table 3-C column 3. The four detectors FAST, GFTT, HARRIS and CenSurE all show a 0.0 match error, demonstrating that keypoints $k_M \in K_M$ found by these detectors can be matched to identical keypoints in $k_I \in K_I$ in the original images.

Table 3-A: Breakdown of keypoint metrics per detector, across all 8,677 images in the CALTECH101 dataset. From left to right; (a) Detector name, (b) Number of keypoints found in all the original images, and in the mirror images (c), (d) Number of keypoints matched between the images and their mirror image, (e,f) tally of keypoints found in each set of images and not the other.

Detector	# keypoints in the original images	# keypoints in mirror images	# matches	excess keypoints in original images D_1^d	excess keypoints in mirror images D_2^d
BRISK	1 443 461	1 444 196	1 401 422	16 409	17 144
FAST	17 013 915	17 013 915	17 013 915	0	0
GFTT	6 865 916	6 865 915	6 865 887	15	14
HARRIS	3 361 261	3 361 262	3 361 257	1	2
ORB	3 566 926	3 566 931	3 566 797	5	10
SIFT	3 066 878	3 067 522	3 005 498	20 858	21 502
CenSurE	395 124	395 124	395 124	0	0
SURF	4 664 275	4 663 781	4 641 957	8 474	7 980
MSCR	663 284	663 287	662 871	304	307
MSER	740 002	740 439	722 555	6 682	7 119
	41 781 042	41 782 372	41 637 283		

Table 3-B: Measurements of accuracy in the spatial domain. The highlighted detectors FAST, GFTT, HARRIS and CenSurE are those which are shown to be reflection invariant.

Detector	Mean Distance Error	Mean Size Error	Mean Angle Error
BRISK	1.98	0.434	9.28
FAST	0.00	0.000	-
GFTT	8.07E-06	0.000	-
HARRIS	4.19E-07	0.000	-
ORB	0.48	1.324	2.61
SIFT	2.10	0.214	11.52
CenSurE	0.00	0.000	-
SURF	0.22	0.125	0.76
MSCR	0.01	0.005	0.22
MSER	0.86	0.317	1.49

Table 3-C: Measurements of accuracy based on SIFT feature descriptors

Detector	Mean Descriptor Distance Error	Mean Descriptor Match Error
BRISK	58.02	26.81
FAST	0.00	0.00
GFTT	0.00	0.00
HARRIS	0.00	0.00
ORB	30.53	39.38
SIFT	113.32	62.46
CenSurE	0.00	0.00
SURF	3.65	7.32
MSCR	0.70	0.88
MSER	13.68	5.92

Error measurements

The *mean descriptor distance error* and *mean descriptor match error* are perhaps the most important measurements in assessing invariance to bilateral symmetry. *Mean descriptor distance error* is the average Euclidean distance between matched descriptors measured in 128-dimension descriptor space. A mean of 0.0 indicates perfect matching. *Mean descriptor match error* is a measure of the matching accuracy based on descriptors against matching spatially. In a perfect set of bilateral feature keypoints, the feature descriptor match would yield the same keypoint pairings as matching keypoints spatially.

Eight out of ten feature detectors that we tested found a different number of keypoints in an image and in the mirror of the image. FAST and CenSurE detectors were the two exceptions. The initial test identified that these two detectors were consistent in the number of keypoints that they were able to detect and further experiments confirmed that there was consistency across all 101 categories. Our measures of error – *mean distance error*, *mean size error*, *mean descriptor distance error* and *mean descriptor match error* – all confirmed perfect bilateral symmetry in all of the 17 013 915 and 395 124 keypoints, respectively. It is important to note, though, that these detectors only determine location and size of a feature, and do not define the angle of orientation of the feature, which we conclude to be a significant factor in their invariance. The fifth error measure, *mean angle error* is therefore omitted for these detectors. Nonetheless, the invariance in an important attribute of the detectors for location and size.

Other detectors that do not identify the angle of orientation of the feature keypoint – GFTT and HARRIS – also performed well in our error measurement tests. The number of keypoints detected in the images and their mirror reflected images varied in 18 and 3 categories respectively and the detectors therefore cannot be seen as perfectly invariant to bilateral symmetry. However, all the error measurements are 0.0, except for the *mean distance error* which are $8.07E^{-06}$ and $4.19E^{-07}$ respectively. With such small errors across 6 865 916 and 3 361 261 feature keypoints, it is fair to conclude these to also be invariant to bilateral symmetry, given the *mean descriptor distance error* and *mean descriptor match error* are both 0.0. It should be noted though, that feature matching and filtering is required to achieve correct correspondence between non-identical sets of keypoint features.

The region-based detectors MSCR and MSER generally identify fewer features – 663 284 and 740 002 in the original images – and demonstrated a greater invariance to bilateral symmetry than keypoint detectors with orientation. MSCR feature positions are very consistent, with a *mean distance error* of 0.01 pixels and *mean size error* of 0.005 pixels and the *mean angle error* is only 0.22° . Corresponding MSER error values are 0.86 pixels, 0.317 pixels and 1.49° . MSCR (on colour images) has a low *mean descriptor match error*

of 0.88. MSER using the same underlying algorithm on grey-scale images has a larger error of 5.92 suggesting that colour helps with invariance to bilateral symmetry in the maximally stable region algorithm.

The final four detectors have much larger error measurements in some or all of our position, size and angle metrics (Table 3-B), which cause some large error values in the descriptors (Table 3-C). SURF has the lowest *mean descriptor match error* of 7.32 and BRISK, ORB and SIFT show much higher error values of 26.81, 39.38 and 62.46 respectively. In an image and mirror image pair, correspondence can be expected to fail with this accuracy on this number of keypoint matches.

Causes of reflection variance

Many algorithms described in the research literature – especially saliency based feature detectors – are not inherently sensitive to orientation. Nonetheless, no mention is made of reflection invariance in the papers, suggesting a general unawareness of this property. Consequently, we have observed several cases where commonly used, freely available code – including reference implementations from original authors – have an invariance worsened by, or *caused by*, choices made in the implementation.

The FAST detector analyses the set of pixels in close proximity to a candidate pixel and classifies each pixel as a corner or non-corner pixel without regard of the relative spatial layout of the neighbourhood. The BRISK derivative introduces sampling of the pixel neighbourhood and consequently loses the inherent reflection invariance. ORB, another FAST derivative, uses an intensity centroid [188] to measure orientation, which may suggest a cause for the error that we have observed. HARRIS is invariant through its use of local auto-correlation, and GFTT largely maintains this property in its extension. SIFT sub-samples the image at higher scales, losing pixel-precision as [39] observes, and this pixel-level imprecision is further observed in our experiments. SURF's use of a Hessian matrix improves on SIFT's invariance. The two region detectors look for stable regions, similar to a watershed algorithm. The use of three colour channels in MSCR shows a large improvement in the region stability in the oriented image.

Design and implementation choices both contribute to invariance to bilateral symmetry in common feature detectors. Scale-invariant detectors, for example, smooth pixel values when scaling an image, which introduces pixel value changes sensitive to surrounding values and leads to invariance. This is a design issue. Other invariance is caused by implementation choices. For example, algorithms that use a Difference-of-Gaussian pyramid (such as SIFT) for sub-pixel feature detection can inadvertently increase their reflection dependence by using 32-bit floating point arithmetic for intermediate

calculations.

Using the popular OpenCV library [189] – version 2.4.12 for C++ – we tested the `GaussianBlur()` function that convolves an image with a specified Gaussian kernel. We found that the library implementation that uses 32-bit floating-point arithmetic produces reflection-sensitive convolutions for many images that we tested. We re-implemented the algorithm using 64-bit floating-point arithmetic and all convolutions of our test images were reflection invariant (Listing 3.1). This demonstrates that the implementation choice of using 32-bit floating-point arithmetic introduces a rounding error which can subsequently cause invariance in the dependent interest point calculations.

Conceptually, one would expect salient regions to be less biased to horizontal orientation, because they use neighbourhood colour and intensity measures and are less dependent on pixel gradients. However, common implementations of salient region detectors such as *maximally stable extremal regions* (MSER) [40] can suffer in the initial step of the algorithm blurring the image with a Gaussian kernel. In their saliency detector reference implementation, Cheng et al. [190] exhibit orientation sensitivity due to many reasons including floating point errors in colour quantization which are realized differently dependent on the order in which the data is processed, which is determined by the image orientation. Increasing floating point arithmetic to double-precision 64-bit calculations correct the quantisation sensitivity to reflection invariance.

```
1   cv::Mat src = imread("image.png", CV_LOAD_IMAGE_GRAYSCALE);
2
3   cv::Mat fpt;
4   src.convertTo(fpt, CV_32F, SIFT_FIXPT_SCALE, 0);
5
6   cv::Mat fpt_r;
7   flip(fpt, fpt_r, 1);
8
9   double const sigma = 1.24899971;
10  GaussianBlur(fpt, fpt, cv::Size(), sigma, sigma);
11  GaussianBlur(fpt_r, fpt_r, cv::Size(), sigma, sigma);
12
13  assert(countNonZero(fpt - fpt_r) == 0);
```

Listing 3.1: Example C++ code to assess reflection invariance of a Gaussian filter in OpenCV. Using 32-bit floating point arithmetic – `CV_32F` on line 4 – will often result in an assertion failure on line 13 indicating that a Gaussian filter on a horizontally flipped image does not produce the same as the result as applying the same filter to the original image. Changing to use 64-bit double precision arithmetic – `CV_64F` – produces identical results on all of our test images, with no assertion failures.

3.3 Reflection Invariance in learned representation systems

While the recent adoption and development of neural network techniques have undoubtedly produced impressive results in computer vision tasks, and object and scene recognition in particular, they are not at all robust to variation in data. Studies have shown that changing an image in a way imperceptible to humans can cause a deep neural network (DNN) to label the image as something else entirely [191] and that it is easy to produce images that are completely unrecognisable to humans, but that state-of-the-art DNNs believe to be recognizable objects with 99.99% confidence [192].

Recently published research on a scene recognition system [193] includes an online demonstration. Figure 3.5 shows a set of four images and their mirror reflections (top row) with the *information regions* that the author’s online demo produce. The information regions are salient areas that the system has identified in its quest to understand and describe an image. We note the difference in the information regions and suggest that this may demonstrate a bias to the horizontal orientation of the image.

Table 3-D shows the detailed results from scene recognition using the system, determining the environment, semantic categories and SUN scene attributes [194]. The category

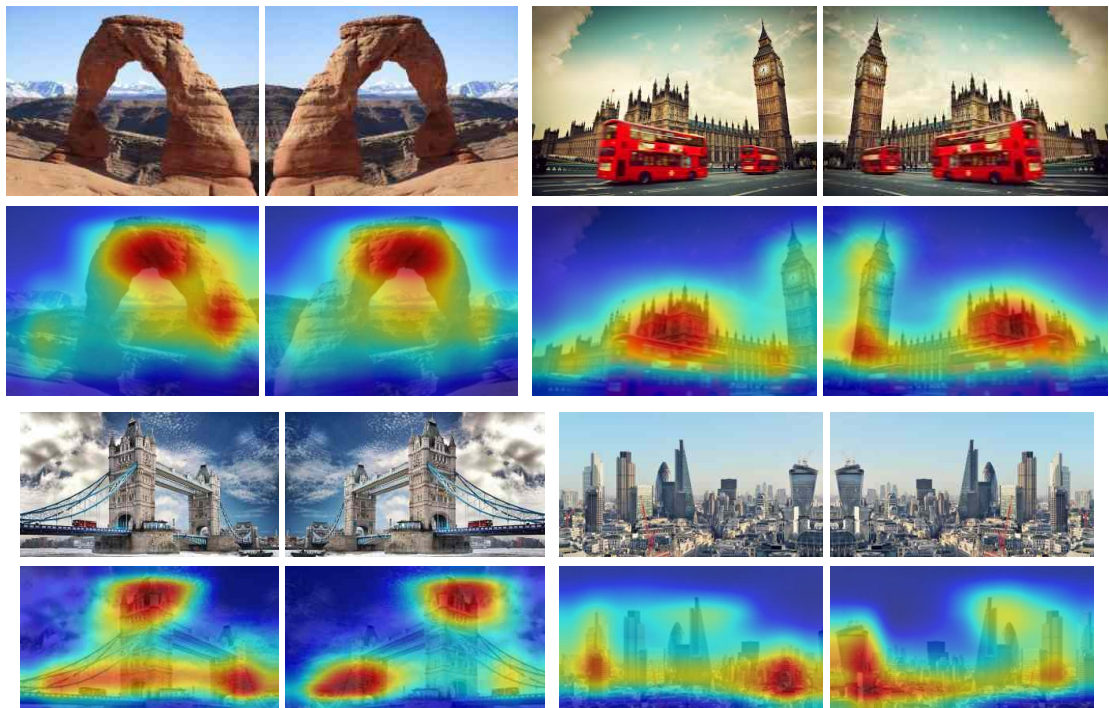
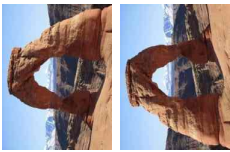
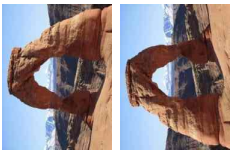








Figure 3.5: *Informative regions* of images and their mirror, identified by [193]. Note where the informative regions are not perfect mirror images which may suggest the algorithm’s sensitivity to the horizontal orientation of the image.

Table 3-D: Predictions from Deep Learning Scene Recognition systems [193]

Environment	Semantic categories	SUN scene attributes	Category
	rock_arch:0.75, arch:0.24	naturallight, openarea, ruggedscene, climbing, rockstone, directsun, dry, vacationingtouring, natural, warm	rock_arch
	rock_arch:0.74, arch:0.25	naturallight, ruggedscene, rockstone, openarea, climbing, directsun, dry, vacationingtouring, warm, natural	rock_arch
	tower:0.50, bridge:0.25, viaduct:0.12	man-made, clouds, openarea, naturallight, mostlyverticalcomponents, metal, vacationingtouring, nohorizon, directsun, sunny, congregating	tower
	tower:0.50, bridge:0.25, viaduct:0.12	man-made, clouds, openarea, naturallight, mostlyverticalcomponents, metal, vacationingtouring, nohorizon, praying, directsun, sunny	tower
	skyscraper: 0.72, tower: 0.13, office_building: 0.06	mostlyverticalcomponents, openarea, man-made, naturallight, directsun, sunny, far-awayhorizon, clouds, metal, driving, transportingthingsorpeople	skyscraper
	skyscraper:0.66, tower:0.13, office_building:0.11	mostlyverticalcomponents, openarea, man-made, naturallight, directsun, sunny, driving, transportingthingsorpeople, clouds, far-awayhorizon, metal	skyscraper
	abbey:0.64, palace:0.16	man-made, clouds, openarea, mostlyverticalcomponents, naturallight, vacationingtouring, praying, nohorizon, electricindooringlighting, metal	abbey
	abbey:0.66, palace:0.15	clouds, man-made, openarea, mostlyverticalcomponents, naturallight, praying, vacationingtouring, nohorizon, metal, electricindooringlighting	abbey

column summarizes the highest scoring semantic category. Despite the differences in salient areas of the images, the overall categorisation has not been affected. Each image and its mirror image are categorized the same in these examples. However, there are differences in the detail, which illustrate inconsistencies that, in boundary cases, could change the categorization. The semantic categories are rated with a likelihood. The Rock Arch – a stock image from the author’s own demonstration – reduces in likelihood by 0.01 in the mirror image, the Palace of Westminster [195] is classified exactly the same in each pair, Tower Bridge – another stock image from the author’s own demonstration – appears less like a skyscraper and more like an office building in the reflected image than in the original, and the City of London skyline [196] increases its likelihood of being an abbey in the reflection image. The inconsistency in the ratings, albeit very small, further strengthens our resolve that computer vision systems are commonly bias to image horizontal orientation.

We used a second neural network based object recognition system, *The Wolfram Language Image Identification Project* [197], to test classification of our images, this time using different sizes of the same image. Table 3-E shows the results; the Rock Arch is classified differently in its original orientation at a small scale, the Palace of Westminster was classified consistently at each scale, Tower Bridge is classified differently in its original orientation at a large scale and the London Skyline is classified differently in its mirror orientation at a large scale. These results show that this system is sensitive to scale, and that the scale change also influences the invariance to horizontal reflection.

Finally, Microsoft’s *How-Old.net* [198] asks “*How Old Do I Look?*” and uses machine learning to guess the answer to the question from a photograph. We used photographs

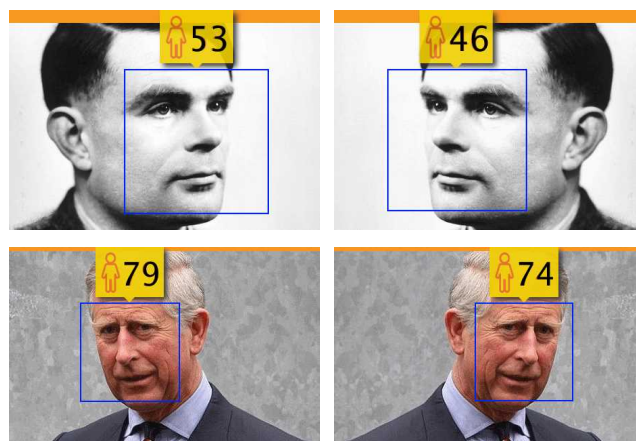











Figure 3.6: Microsoft’s *How-Old.Net* demonstration [198] attempts to guess a person’s age from a photograph image. These two examples demonstrate that the system is sensitive to image orientation – and not head orientation – as the ages are quite different for each pair.

Table 3-E: Object recognition results from the *Wolfram Language Image Identity Project*

Resolution	Original	Mirror
550 × 412	 arch	 arch
244 × 183	 broken arch	 arch
736 × 490	 fire truck	 fire truck
275 × 183	 building	 building
607 × 338	 bascule ¹	 church
329 × 183	 church	 church
4370 × 2383	 oil refinery	 industrial park
336 × 183	 oil refinery	 oil refinery

¹ *bascule* – a moveable bridge with a counterweight to balance an upward swing that provides clearance for boat traffic [https://en.wikipedia.org/wiki/Basculle_bridge]

of Alan Turing [199] and Charles, Prince of Wales [200] and observed the difference in age that was guessed for each image and its reflection (Figure 3.6). In both cases, the ages decreased in the reflected image (*right*), despite the orientation of the head being different in each case.

This inconsistency in results is perhaps more surprising as the image orientation affects the guess of the person’s age, but the system does not appear to be intrinsically biased towards the orientation of the head itself. On close examination, the bounding boxes of the identified *faces* are different sizes – smaller in the reflected image in both cases – by 5 pixels in each *x*- and *y*-axis in the case of the photograph of Alan Turing and 1 pixel in each axis in the case of Prince Charles. The detected face of Alan Turing is in a consistent corner position relative to the visible ear, and the detected face of Prince Charles is consistent in the opposite top corner. Thus we conclude that the face detection algorithm used in the system is sensitive to head orientation and this may affect the subsequent learned system of age estimation, which may or may not be orientation-sensitive itself.

3.4 Conclusion

In this chapter we have assessed ten popular image feature detectors to determine their invariance to bilateral symmetry. We focussed on the accuracy and consistency of feature detection between an image and its mirror reflection. We conclude (Table 3-F) that FAST and CenSurE detectors are perfectly invariant and GFTT and the Harris Corner detector are invariant after feature matching and filtering algorithms are applied to find the correct correspondences in uneven sized sets of detected interest points. BRISK, ORB, SIFT and SURF cannot be considered invariant to bilateral symmetry, and SIFT is the least

Table 3-F: Conclusions of the invariance characteristics of ten feature detectors used in our experiments

Detector	Invariant
BRISK	No
FAST	Perfect
GFTT	Yes, after matching
HARRIS	Yes, after matching
ORB	No
SIFT	No
CenSurE	Perfect
SURF	No
MSCR	Somewhat
MSER	Somewhat

invariant of all the detectors that we have experimented with. Region-based detectors MSCR and MSER were also assessed based on a common approach of defining a keypoint at the centre of the detected region. In this case, MSCR is largely invariant and MSER is somewhat invariant, indicating that colour plays an important role in the invariance of maximally stable region algorithm.

We have proposed *reflection invariance* to be an important consideration when designing and implementing algorithms. It is evident from the cited contemporary research projects that many inconsistencies exist within applications of scene classification, object detection and age-guessing when systems are presented with images and their horizontal reflections. In each of our examples, the systems have produced results that are different for each reflected image orientation. We have described where some of the sensitivity is exhibited in feature detection and descriptors, and the interested reader is referred to [47] for a detailed analysis and experiments in alignment and localization.

A pattern detection scheme that is robust to reflection invariance is presented in Chapter 5 with specific discussion in respect to this invariance in §5.2.2.

Chapter 4

Feature correspondence in poor quality images

Feature correspondence is the common and important technique of matching features found in an image to those in another image, and is used extensively in tasks such as image alignment, stitching, object tracking, and search and retrieval (§2.1.4). These image features are represented using feature descriptors that typically describe the texture (pixel intensity structure) of a small neighbourhood of pixels. The challenging conditions of video acquisitions illustrated in §1.1 suggest that texture alone is not sufficient to find correspondences between frames in security videos, and the effectiveness of matching popular intensity descriptors such as SIFT and SURF is therefore limited. In this chapter, we establish a method to improve the robustness of finding and matching features in poor quality images by combining colour and texture information to increase discriminative properties of a feature descriptors. The methods that we develop are useful in improving the effectiveness of object tracking etc. in low quality videos, and are tested using street-scene videos from real crime investigations in London.

4.1 Measuring image blur

The first step in dealing with images that vary in quality is to be able to somehow quantify the degree of blurriness of a frame image so we can adjust some of the processing parameters accordingly. Accurate models for calculating the motion blur of an image have been described using a multitude of techniques, from estimating the parameters of a Point Spread Function [201] to using machine learning [202]. Our intent is not to accurately calculate the blur parameters such that the blurred image can be restored to a

sharp image, but to *quickly* be able to estimate the degree to which an image, or part of an image, is blurred. For this use a straightforward method that is fast to calculate and is shown to give a reasonable estimation of blurriness for our purposes.

We derive an efficient technique from the intuition that a blurred image will contain fewer sharp edges than a non-blurred image. The number of edges in an image can therefore be used as an expression of image blurriness (or, conversely, image sharpness). We use a Canny edge detector [203] with a 3×3 Gaussian kernel, a lower threshold of 175 and an upper threshold of 225. The small Gaussian kernel balances execution time with sensitivity to salt-and-pepper noise that can be caused by analogue-to-digital converter errors or bit errors in transmission. The threshold values have been chosen empirically to avoid breaking noisy edges (if the lower threshold is too high) and to reduce fragmentation if the upper threshold is too low.

The Canny edge detector is used to construct a binary edge map E from image I of size $m \times n$,

$$\begin{aligned} E_I = e_{ij} \quad & i = 1, 2, \dots, m \\ & j = 1, 2, \dots, n \\ & e_{ij} \in \{0, 1\} \end{aligned} \quad (4.1)$$

The *sharpness* of image I is then determined by a function $S(I)$ that calculates the fraction of non-zero pixels in the edge map E , which is the fraction of pixels representing edges in the image I .

$$S(I) = \frac{1}{mn} \sum_{i=1}^m \sum_{j=1}^n e_{ij} \quad (4.2)$$

We use this measure of image sharpness to eliminate duplicate frames and static scenes (§4.2) and in blur-sensitive feature detection (§4.3).

4.2 Duplicate frames and static scenes

Frame rate instability can cause non-deterministic duplicate frames to appear in a video sequence. Visually duplicate frames are seldom identical in their pixel values, and so eliminating them from the processing stream requires more attention than simple pixel comparison. A related problem in video surveillance, particularly with fixed cameras, are static scenes; those where nothing happens for periods of time, but the camera continues to record and subsequently produces large sequences of video with no activity. A security camera will often produce a time stamp on the captured image, and frames are therefore different in the counting clock (and date roll-over at midnight) but otherwise unchanged (Figure 4.1).



Figure 4.1: Five frames of a video sequence with a counting time stamp but no activity in the scene

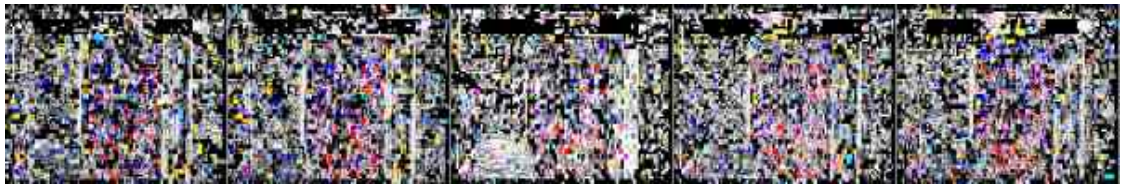


Figure 4.2: Difference images of the frames in Figure 4.1 with their previous frame. The colours represent pixel value differences in the RGB channels

Processing duplicate images is time-consuming, wasteful of resource and complicates algorithms that are designed to work with movement. It is therefore desirable to eliminate duplicate frames and static scenes from the processing queue. The sparse optical flow is calculated using the multi-scale Lucas-Kanade algorithm [204], with *good features to track* [29] keypoint features. Corner features are compared in each pair of temporally adjacent frames and if no features change position in the second frame, then the frames are treated as identical. We ignore extreme small or large movements and only consider movements between 1 pixel and 2σ pixels, as 95% of values are within two standard deviations of the mean. An exception case is made if one or other of the images is measured as blurred (Equation (4.3)) and the other isn't. In this case, the feature matching algorithm may fail but given a difference in sharpness measure, the frames are considered not duplicate and not static.

The determination of a *blurred image* is to find a suitable threshold below which $S(I)$ from Equation (4.2) must fall to represent an image that is blurred. Our definition of a blurred image is therefore;

$$\text{image } I \text{ is blurred} = \begin{cases} \text{true} & \text{if } S(I) \leq \varphi \\ \text{false} & \text{otherwise} \end{cases} \quad (4.3)$$

In our experiments, we empirically discovered a value of $\varphi = \frac{1}{32}$ performs well on our dataset; an image is classified as a blurred image where edges are present in 3.125% or less of the image. This value is sensitive to the content of the scene. Our dataset of CCTV footage contains scenes that are generally *busy*, in built-up areas containing a lot of activity (moving people and vehicles) and structures (e.g. buildings, etc.). The duplicate

frame elimination is an optimisation to reduce processing frames where the result will be the same as a previous frame. As such, there is some flexibility in the robustness of the algorithm. The value of φ determines the classification boundary of a blurred frame and false positives may cause a frame to be processed where perhaps it need not have been, or in the worst case, incorrectly identified as a duplicate, and not processed at all.

The method is robust to time stamp counters within the frame because movement between frames is only identified if a matching feature is in a different position in the two frames. In the case of counting digits of a time stamp, features surrounding a 4, for example, are not matched in an adjacent frame with the features surrounding the 5 that replaces the 4, and therefore no movement is detected. This technique therefore detects movement between frames, not simply differences between the frames.

Figure 4.1 shows an example of frames from a video sequence recorded with a static camera, with a time stamp counter in the corner. Each of these frames were determined to be duplicates by the above algorithm, despite the substantial pixel value differences shown in Figure 4.2, caused by stability of the optical sensor in an outdoor scene.

4.3 Blur sensitive feature detection

Image blur is a very significant hindrance to matching features between frames in low quality images. This observation leads us to adapt feature detection to maximise correspondence accuracy in a technique we call *blur sensitive feature detection*. The method is designed to optimise a local region within an image with respect to its *blurriness* and that of the adjacent frame image to which correspondence is to be established. Applying a localised blur of the area before detecting features can help to find more similar features to the corresponding image, if the amount of image blur can be more closely matched.

In the absence of any prior knowledge of the properties of the image blur, applying a Normal distribution Gaussian filter is a sensible choice for two reasons. First the central limit theorem shows that the sum of many independent random variables is approximately normally distributed and many complex systems can therefore be successfully modelled as normally distributed noise. Second, of all possible probability distributions with the same variance, the normal distribution encodes the maximum amount of uncertainty and inserts the least amount of prior knowledge into the model [205; §3.9.3, §19.4.2].

A relationship map M is established to correspond the properties of a 2D Gaussian kernel G_k of size k with the sharpness measure of the query image I_Q after a convolution with G_k . The map holds sharpness values for the query image after convolution with 2D

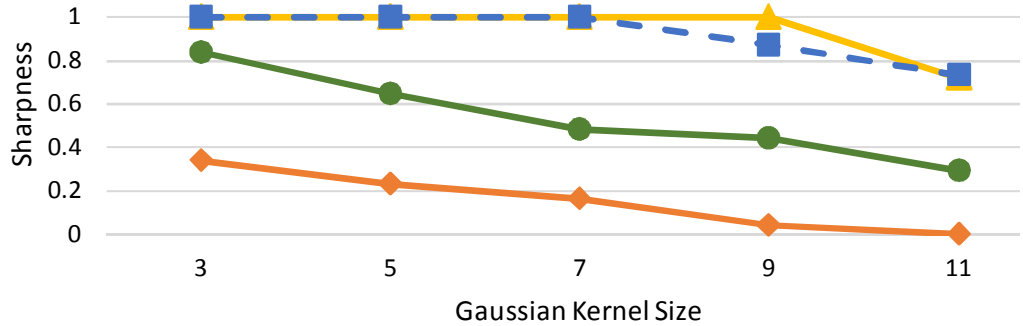


Figure 4.3: Relationship between the size of a Gaussian kernel (x -axis) used to artificially blur example query images and the sharpness of the resulting image. Each curve represents a sample query image and their measured sharpness at different Gaussian kernel sizes. The kernel size steadily increases while the decline in sharpness (increase in image blur) varies with different query image regions. The relationship map M therefore needs to be calculated for each query region used for correspondence matching.

Gaussian filters of kernel size in the set Γ . Let

$$\Gamma = \{p \in \mathbb{N} | p = 2q - 1 \wedge q \in \mathbb{N}\} \quad (4.4)$$

The sharpness is calculated for each kernel size and stored in an associative map $k \rightarrow S(I)$,

$$M(k) = S(I_Q * G_k) \quad \forall k \in \Gamma \wedge k \leq \alpha \in \mathbb{N} \quad (4.5)$$

where

- $M(\cdot)$ represents an entry in an associative map
- $S(\cdot)$ image sharpness, from Equation (4.2)
- G_k Gaussian filter of kernel size k
- $*$ represents 2D convolution
- α an upper bound on the Gaussian kernel size

Figure 4.3 shows some examples of the relationships between the size of a Gaussian kernel used to artificially blur example query images and the sharpness of the resulting image in M . This demonstrates the variance in the correlation between the steepness in the decline in sharpness (increase in image blur) with steadily increasing kernel sizes for different query image regions, and therefore the need to calculate M for each query region used for correspondence matching.

A sharpness adjustment S_a is calculated as the difference between the sharpness of the original (unconvolved) query image region I_Q and the target image I_T to which

correspondence is to be established.

$$S_a = S(I_Q) - S(I_T) \quad (4.6)$$

The value of S_a is used to find the corresponding Gaussian kernel size k in M which, when convolved with I_Q will produce an image I'_Q with sharpness that will most closely match $S(I_T)$ from all Γ

$$m = \arg \min_k \{S_a - S(I_Q * G_k)\} \quad m \geq 0 \quad (4.7)$$

$$I'_Q = I_Q * G_m \quad (4.8)$$

Features are detected in, and extracted from I'_Q and I_T and correspondences are found between these feature sets.

Matching performance is considerably improved by aligning the sharpness of the images before feature detection. However, blurring an image reduces texture structure, which generally reduces the effectiveness of feature detectors, especially corner-based detectors such as FAST and BRISK. If no features are found in I'_Q , we repeat the process with I_Q as the entire query image, not bounded to the query region of interest. We do this with the understanding that the bounded region of interest contains little texture so retrying with greater kernel sizes would offer only minor improvements, whereas the sharpness of the query image as a whole provides more information with respect to blur induced by camera movement. In the unlikely event that no features are found in the revised I'_Q , we fall back to features found in I_Q . In all of our experiments, this fall back position is never required as the unbounded I_Q image always produces a usable feature set.

4.4 Combinatorial Texture and Colour feature matching

We create a new combinatorial feature descriptor representing local features with colour information – or grey scale intensity structure – from the surrounding region. First, any local feature detector is used to find feature locations and both a keypoint and a region are defined for each. In the case of a keypoint detector such as SIFT, a circular region is created with its center at the keypoint co-ordinates. For region based feature detectors such as *Maximally Stable Extremal Regions* (MSER), the region is approximated using an ellipse fitting algorithm through the region boundary points and a keypoint is defined at the center of the ellipse.

With the resulting set of keypoint locations and region definitions, we extract a texture

descriptor at each keypoint. The texture descriptor is a standard feature descriptor that will be extended by our method to improve its discriminative capability in colour images. Using the region shape as a mask over the colour image, pixels that fall within the shape are quantised into a colour histogram. This histogram is transformed into a feature descriptor using the histogram bins as the feature dimension and the texture descriptor and colour histogram are concatenated into a composite descriptor.

The RGB colour space is known to be a poor representation for colour segmentation as there is no straightforward correlation between the RGB channel values and the intensity of a particular colour that lends itself to simple thresholding. We therefore transform the RGB image to the HSV colour space for our algorithm. The Hue (H) channel determines the colour, the Saturation (S) is the intensity of the colour and the brightness or luminance (V) can be used to find non-colour white, grey and black.

Allocating a pixel value to its closest histogram bin is done by calculating a partial distance in HSV colour space. For colour entries in the histogram, the distance is determined by the Euclidean distance of the Hue and Saturation components, $d = \sqrt{H_i^2 + S_i^2}$. Distance to non-colour entries in the histogram, white, black and grey, are calculated using the Euclidean distance of the Saturation and Value (luminance) components, $d = \sqrt{S_i^2 + V_i^2}$. Measuring colour distances in the HSV colour space in this way maintains robustness to affine illumination changes in the image.

4.4.1 Designing a combinatorial descriptor

In designing an algorithm to extend an existing feature descriptor, consideration is made to the potential of falsely matching dissimilar features of similar colour or moving vectors in feature space closer together where neither their feature descriptor nor the colour are similar. Our goal is to produce a generic extension that can be used with any underlying texture feature descriptor, and so we focus on a method to combine an n_1 -dimension texture feature descriptor with an n_2 -dimension colour histogram in such a way as to discriminate similar features of different colours without these pitfalls.

Consider a naïve implementation that concatenates an n_2 -dimension colour-histogram onto a n_1 -dimension texture descriptor to form an $(n_1 + n_2)$ -dimension feature descriptor, and compares the combined descriptors as single vectors. Extending the dimensionality to accommodate the colour information is intuitive, however this method will treat the colour histogram as an integral part of the feature. The Euclidean distance is calculated for the vectors as a whole, losing the unique properties of the colour histogram. Further, the two individual vector descriptors measure attributes in different scales, and a simple combination will yield bias within the components.

4.4.2 Distance Definition

We follow the understanding of Lowe's distance ratio (§2.1.4) in our method and use the colour information of both features to scale the distance between their descriptors. In doing this, a metric of the difference in the colour histograms logically moves the features apart. The composite feature descriptor \mathbf{f} is conveniently represented as a single n -dimensional vector, where n is the sum of the lengths of the texture \vec{t} , and histogram \mathbf{h} .

$$\mathbf{f} = (\vec{t}, \mathbf{h}) \quad (4.9)$$

In calculating the distance D between two composite descriptors, we first consider a distance measure between each of the two parts independently, d_1 and d_2 , and combine the results. The texture descriptor distance d_1 is a standard calculation of the Euclidean distance between the two vectors and d_2 is the distance between the two colour histogram descriptors, $H(\cdot)$ from Equation (2.15) on page 32.

$$d_1 = \|\vec{t}_1 - \vec{t}_2\|_2 \quad (4.10)$$

$$d_2 = H(\mathbf{h}_1, \mathbf{h}_2) \quad (4.11)$$

The individual distance measures d_1 and d_2 are then combined to yield a representative distance between the two composite descriptors. A simple sum $D = d_1 + d_2$ does not account for the difference in scale within each of the descriptors, which itself will be different depending on the choice of texture descriptor. The product $D = d_1 d_2$ down-scales the texture distance based on the colour histogram distance, effectively moving similar texture descriptors closer together. This reduces the discrimination of similar textual descriptors, increasing the number of mismatches and reduces the overall accuracy. We derive a composition applying a constant multiplier to the normalised histogram distance and summing with the texture distance, in general form,

$$D = d_1 + \lambda d_2 \quad (4.12)$$

The selection of a suitable value for λ has been the subject of many experiments. Any empirically chosen constant value is not robust for the variety of challenging images from surveillance video images, and we therefore look to a dynamic value for λ which represents the conditions within which the feature appears.

Figure 4.4 shows a comparison of performance using a variety of combination methods; $D = d_1 + (d_2)^2$, $D = d_1 + d_2$, $D = d_2(1 + d_1)$, $D = d_1 d_2$, $D = d_1 d_2 + d_1 + d_2$, and

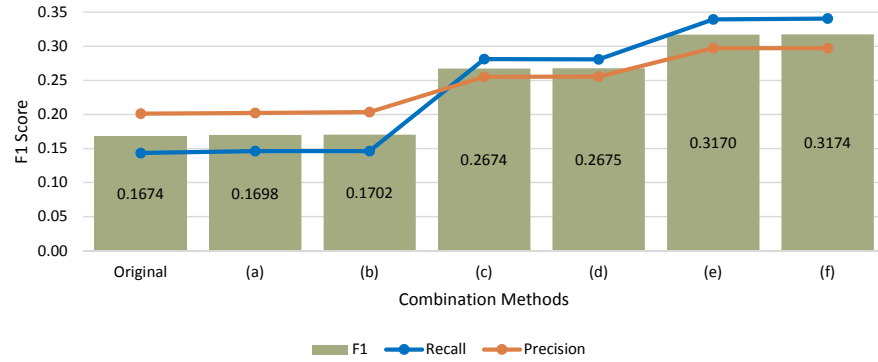


Figure 4.4: F_1 score comparing a variety of combination methods in the distance calculation; (a) $D = d_1 + (d_2)^2$, (b) $D = d_1 + d_2$, (c) $D = d_2(1 + d_1)$, (d) $D = d_1d_2$, (e) $D = d_1d_2 + d_1 + d_2$, and the proposed (f) $D = d_1(1 + d_2)$, against the F_1 measure of an unmodified SIFT descriptor

the proposed $D = d_1(1 + d_2)$, against the F_1 measure of an unmodified SIFT descriptor, highlighting the superior performance of the proposed method in our experiment data.

Using d_2 as a value for λ reduces the impact of the colour histogram distance because d_2 is a normalised value, which when multiplied by itself becomes smaller, and overall less discriminative. However, d_1 is a good candidate. With $\lambda = d_1$, the colour distance is used to scale the distance measure of the texture descriptor so that it discriminates between similar descriptors of different colours.

$$\begin{aligned}
 D &= d_1 + \lambda d_2 \\
 \lambda &= d_1 \\
 \therefore D &= d_1(1 + d_2)
 \end{aligned} \tag{4.13}$$

We see from Equation (4.13) that with $\lambda = d_1$ we apply the colour distance measure as a scalar to the distance between two texture feature descriptors. Increasing the normalised value of d_2 from the range $0 \dots 1$ into $1 \dots 2$, thus upscaling the distance of a texture feature by multiplication. Therefore, let $\mathbf{a} = \mathbf{h}_1$ and $\mathbf{b} = \mathbf{h}_2$, with a_j and b_j being the j^{th} element of each, then the overall distance between two composite descriptors is

$$\begin{aligned}
 D &= d_1(1 + d_2) \\
 &= \left\| \vec{t}_1 - \vec{t}_2 \right\|_2 \left(2 - \frac{\sum_{j=1}^n \min(a_j, b_j)}{\sum_{j=1}^n a_j} \right)
 \end{aligned} \tag{4.14}$$

The use of a scalar applied to the texture descriptor distance ensures that attributes of the texture descriptor such as invariance to affine scale and rotation transformations, are preserved. The calculation of the colour histogram in Hue and Saturation channels

maintains invariance in affine illumination transformations.

Distance Calculation To find the closest descriptor D_c to a given descriptor D_i it is customary to use an algorithm based on Euclidean distance, such as k -Nearest Neighbour. We perform a nearest descriptor calculation in two parts. First, the k -nearest neighbours of the texture descriptor \vec{t} are found using the standard algorithm with $k = 5$, giving $\{\vec{v}_1, \vec{v}_2, \vec{v}_3, \vec{v}_4, \vec{v}_5\}$. For each of the five closest descriptors, we perform the scaling multiplication of Equation (4.14) and determine the descriptor with the smallest resulting distance to be the closest, D_c

$$D_c = \arg \min_i \{D_{v_i}\} \quad (4.15)$$

This is not guaranteed to be optimal; the choice of value k limits the search for candidate descriptors to which to apply the combinatorial algorithm of Equation (4.14), in the interest of runtime complexity. In our tests increasing k to 10 does not improve the final result, and it is reasonable to expect value $k > 10$ will yield no better composition. This calculation does not produce a worse approximation than the common method to reduce computational complexity in a k -Nearest Neighbour search using Approximate Nearest Neighbour algorithm (ANN), which uses a randomised indexing method making the result non-deterministic, but is widely accepted for most matching tasks. Reducing k below 5 is an option for improving runtime complexity further in a system where it is acceptable to trade accuracy for execution speed.

4.5 Evaluation

We evaluate the performance of the proposed descriptor by measuring the accuracy of matching features between pairs of images. The definition of a feature match depends on the matching strategy that is applied [51]. Our intention is to measure the accuracy of our new composite feature descriptor and distance calculation. We therefore compare our results with a nearest neighbour matching algorithm without any threshold filtering, such as Nearest Neighbour Distance Ratio to discard poor matches.

We use seven feature detectors to find initial regions of interest. Five popular intensity based keypoint detectors; Harris Corners detector (HARRIS), SIFT, SURF, BRISK and FAST, and two region detectors; MSER on grey scale representations and *maximally stable colour regions* (MSCR) on colour images. For each of these sets of features, we compare feature matching performance of descriptors extracted using SIFT and SURF, with and without our combinatorial descriptor, and later using OpponentSIFT and OpponentSURF 3-channel descriptors, again with and without our combinatorial descriptor.

The keypoint detectors HARRIS, SIFT and SURF are chosen because of their popularity and widespread adoption in many tasks including object classification and image retrieval [206], and BRISK and FAST for their high performance and relevance for real-time processing. We are keen to demonstrate the universal improvements of our method and therefore also include region based detectors MSER and MSCR in our comparisons.

4.5.1 Blur sensitive feature detection

We evaluate the *blur sensitive feature detection* technique independently using our seven selected feature detectors with state-of-the-art descriptors and Euclidean distance measurements. Our sharpness map contains convolutions with Gaussian kernels up to 11×11 , thus $\alpha = 11$ in Equation (4.5). In experiments, 9×9 was the largest Gaussian filter that produced an improved result, so we include the next largest as our upper limit. The map size is a constraint imposed to control runtime complexity, and can be increased further for other datasets.

Figure 4.5 shows the percentage improvements in matching quality achieved by applying the blur sensitive feature detection algorithm to our test database. The matching accuracy improvement is subject to the choice of feature detector, which is expected because the artificial blurring of the image will affect each detector differently. The matching performance is broadly consistent across all extractors for each detector. The exception are Harris Corner features which vary considerably for each descriptor type, and decreases matching performance in two cases; rootSIFT and OpponentSIFT descriptors. BRISK features yielded consistently small improvements, and matching SURF features was generally more improved. With rootSIFT descriptors extracted from SURF keypoints being improved the most, by 92.8%.

4.5.2 Combinatorial descriptor assessment

We use a fixed colour histogram for all images. In experiments, the 10-bin palette of Park et al. [88] has proven to work well; seven colours and three special considerations for intensities (Table 4-A). This palette has been used for the experiments presented in this paper. The descriptor extension is therefore 10 dimensions in size.

Query by example

A rectangular area of an image is specified as a query region containing features that are to be matched in subsequent frames of the video sequence. In our first test the query region represents a distinctive two-colour back-pack being worn by a person. This

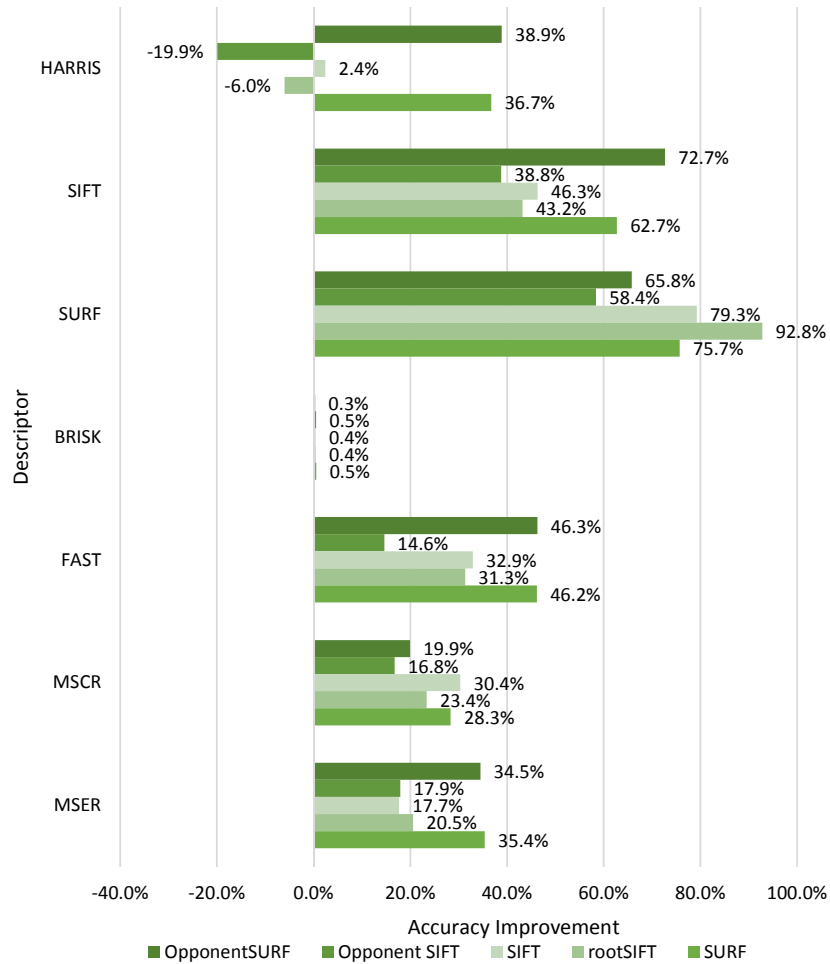


Figure 4.5: Matching accuracy improvements of *blur sensitive feature detection*. Matching accuracy improvement is subject to the choice of feature detector and performance is broadly consistent across all extractors for each detector. However, Harris Corner features vary considerably for each descriptor type, and decreases matching performance in two cases.

Table 4-A: Colour palette from [88] used in our experiments

Colour	H	S	V
Red	0°	100%	100%
Brown	15.1°	74.5%	64.7%
Yellow	60°	100%	100%
Green	120°	100%	100%
Blue	240°	100%	100%
Violet	300°	45.4%	93.3%
Pink	349.5°	24.7%	100%
White	0°	0%	100%
Black	0°	0%	0%
Grey	0°	0	60%

region is matched against 250 video frames, each of which has ground truth information defining the boundaries of the back-pack within it.

Descriptors are created for the query image region and each image under consideration (candidate images) using the method described above. The positions of the features within the candidate image that match with the query region are then assessed relative to the ground truth and determined to be a true or false positive result or a negative result. A true positive result is a feature that matched with the query region (a *query match*) lies within the ground truth region. If a query match falls outside the ground truth then the region is labelled as false negative result. A feature matched between the images from outside the query region that falls within the ground truth region is counted as a false positive result. A match between the images from outside the query region to outside the ground truth region is not used directly within our analysis but are implicitly relative to other metrics.

Results for each feature are tallied for each image, and these are then summed across all of the images in the sequence. The true positive tp , false positive fp and false negative fn totals for the images are then used to calculate the recall and precision measures of performance of each of the four descriptors with and without our extension;

$$recall = \frac{tp}{tp + tn} \quad precision = \frac{tp}{tp + fp}$$

In reporting our results we use the F -measure, the weighted harmonic mean of recall and precision, to measure and compare the accuracy of our combinatorial descriptor and distance measure with well-known descriptors. We favour neither precision nor recall over the other, and therefore use the F_1 score, defined as

$$F_1 = \frac{2 \cdot recall \cdot precision}{recall + precision} \quad (4.16)$$

Intensity descriptors

It is important to compare the feature matching performance with popular intensity descriptors because these have the smallest dimensionality. In a large-scale processing system, size of descriptors is important for minimising memory and disk storage and data processing time.

Our experiments compared the matching performance of SIFT and SURF descriptors against our combinatorial descriptor based on SIFT and SURF with our distance measure, for features detected using Harris Corners (HARRIS), SIFT, SURF, BRISK, FAST, MSCR

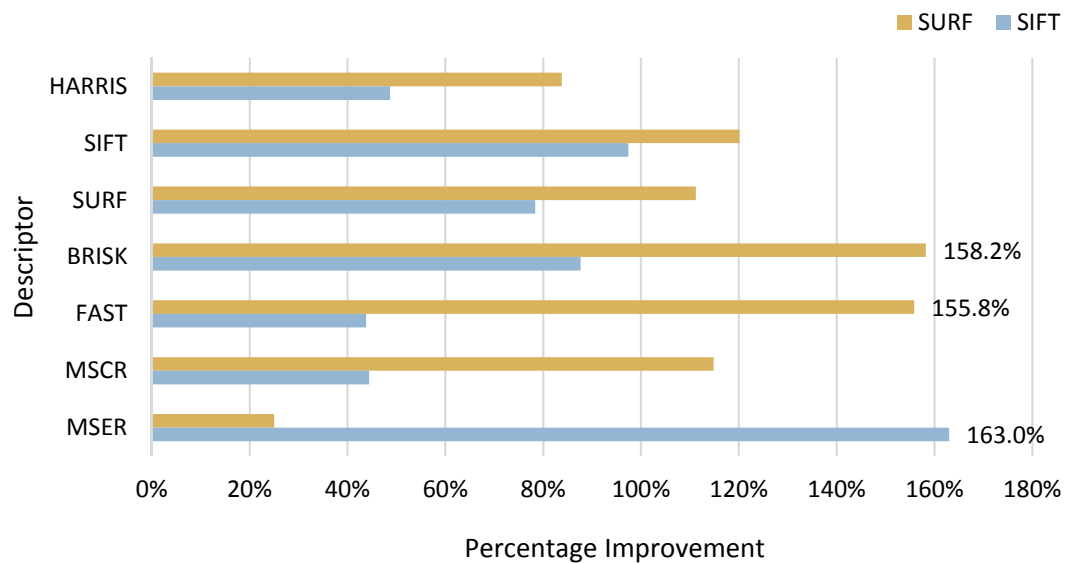
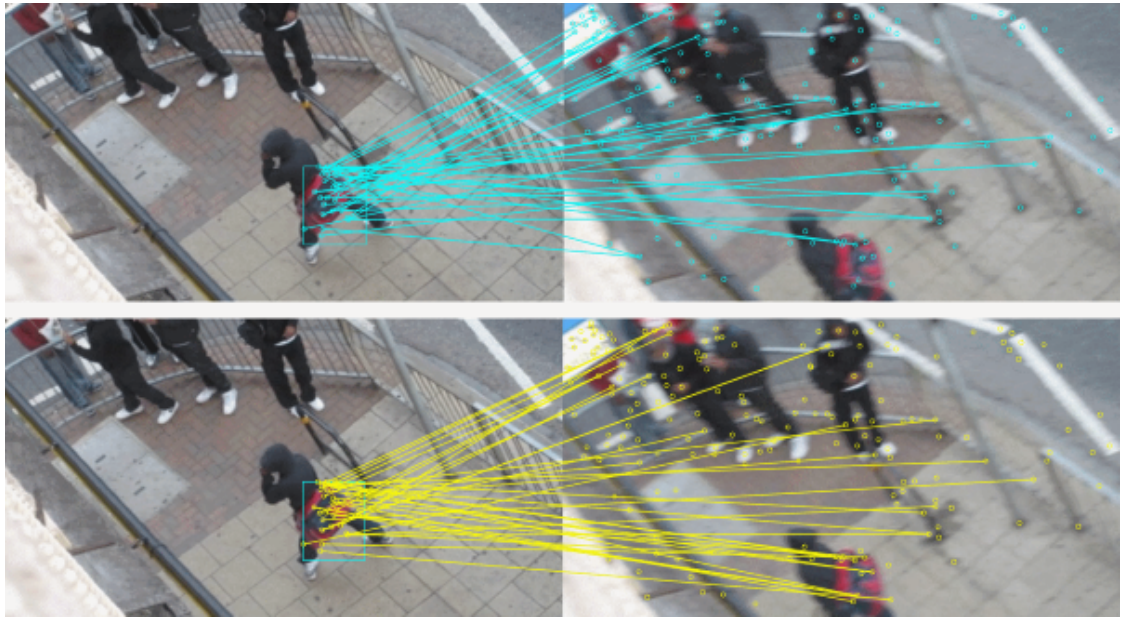


Figure 4.6: Improvement of SIFT and SURF intensity descriptors using our combinatorial descriptor and distance measure. Orange bars show percentage improvements of SURF descriptors using our method, and blue bars show improvements in SIFT. The baseline uses standard descriptors with Euclidean distance measures in feature space. The overall average improvement across all of the feature descriptors in this test was 95.2%.

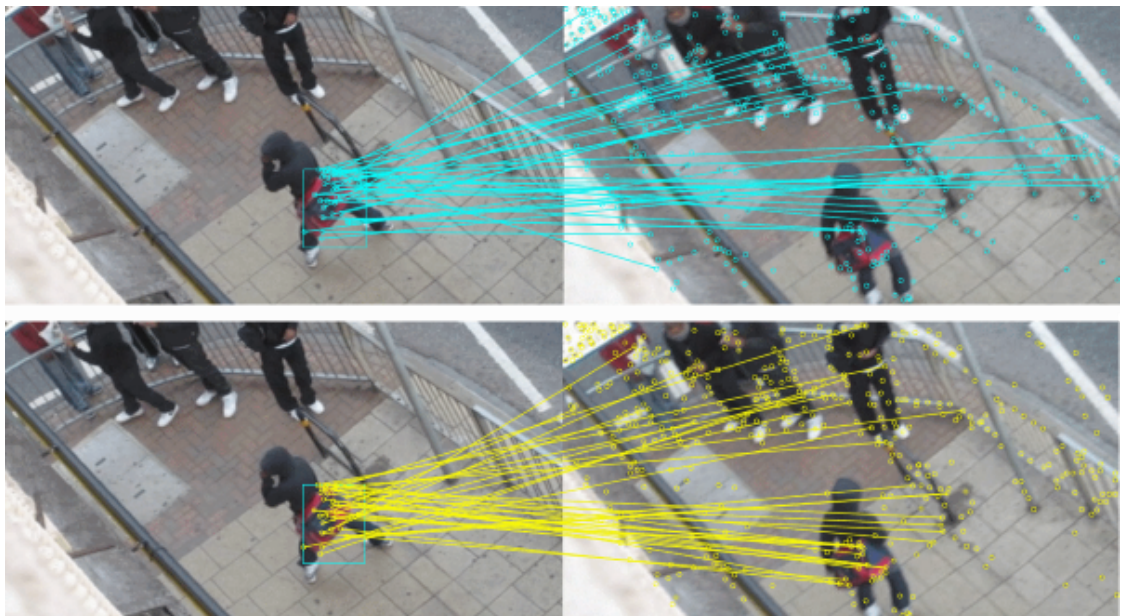
and MSER (Figure 4.6). Feature matching is determined by the nearest neighbour feature in descriptor space. The greatest improvement was achieved with SIFT descriptors extracted from MSER features where the F_1 measure increased by 163% using our method (from 0.064 to 0.167) compared to a plain SIFT descriptor on the same MSER features.

Overall, the average improvement across all feature descriptors in this test was 95.2%.

Figure 4.7 shows two examples of matching feature descriptors from a region of interest within a query image to a subsequent frame in a surveillance video, using a SURF feature detector. Figure 4.7a shows matches of SURF descriptors extracted from the SURF features within the region of interest in the query image (left), and a blurred frame (right). There is a notable increase in the number of features matched into the bag region in the right hand image. Figure 4.7b shows matches from the same query frame to a sharper subsequent frame and demonstrates the reduction in false-positive matches into background features. The less cluttered Figure 4.8 repeats Figure 4.7b using the distance ratio filter (Equation (2.17)) from [31]. There are new positive matches in both images, matching points within the rucksack that are not matched in the top row. In addition, the number of false positives is visibly reduced, with fewer yellow lines matched to the background in the right-hand images.



(a) Matches to a blurred image perform poorly using Approximate Nearest Neighbour (blue matches) and a visually evident increase in matches to the target bag using our method, show in yellow (b). In (a) a single feature is matched, and in (b), eight features matched



(b) Significantly reduced number of false positive matches to the background using our method (yellow) compared with Approximate Nearest Neighbour matching (blue)

Figure 4.7: Two examples of matching SURF features on a coloured bag from a query frame (left in each pair) to a subsequent video frame. This is particularly evident in the railings on the right hand side of the images.

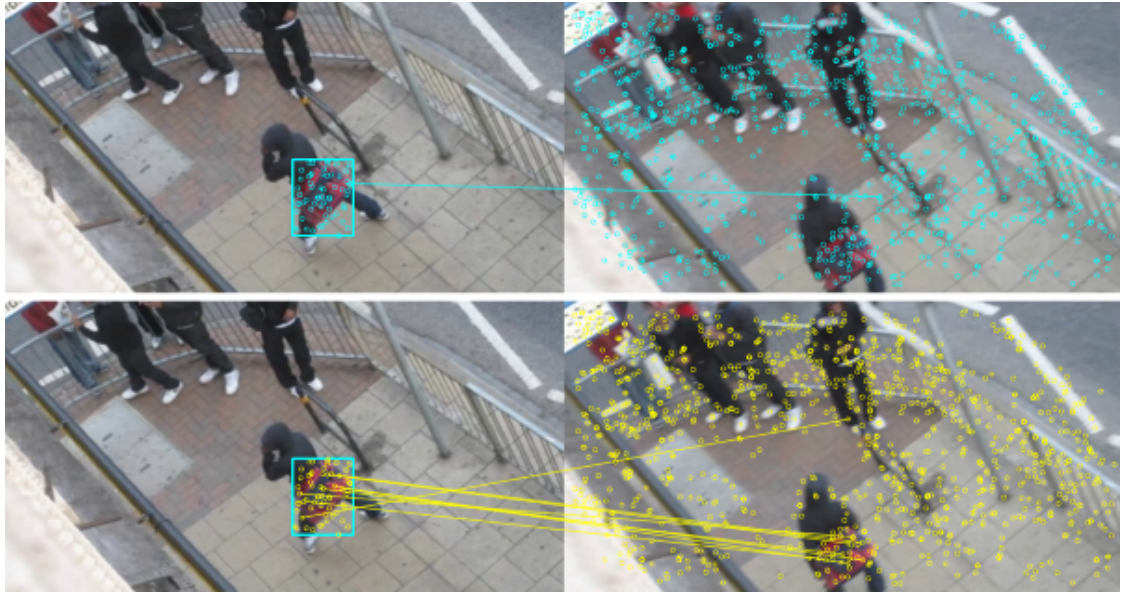


Figure 4.8: *Good matches* (Equation (2.17)) are shown for SURF features (top row) and using our method (bottom row). Only one SURF feature in the distinctive bag is matched, and this has incorrectly been matched to a feature in the background, on the pavement. Using our method, six features are correctly matched to the bag in the second frame, and only one incorrect match to the background remains.

Colour descriptors

We now assess our algorithm using two high-dimensional colour descriptors, Opponent-SIFT (384-dimensions) and OpponentSURF (192-dimensions), with the same features from the previous section.

The F_1 measure on our test video sequence is improved using colour descriptors over using the intensity texture descriptors. This is to be expected as the colour information provides a more discriminative comparison. In our test video sequence, the best match performance was achieved using the combinatorial OpponentSIFT descriptor with FAST features, achieving an improvement of 12% over the original OpponentSIFT descriptors' accuracy of 0.415. Matching OpponentSURF descriptors of FAST features was improved the most of all colour descriptors, by 98.5% but the F_1 score is low, increasing from just 0.149 to 0.296.

OpponentSIFT uses colour information in the extraction of the descriptor and can be expected to out-perform those that do not use colour information in a dataset in which colour is visually distinctive. In their thorough evaluation of colour feature descriptors, van de Sande et al. conclude that OpponentSIFT is generally a better performing descriptor and is a good choice where there is no prior knowledge of the images or object/scene categories [61]. In our tests, results show that our extension method gene-

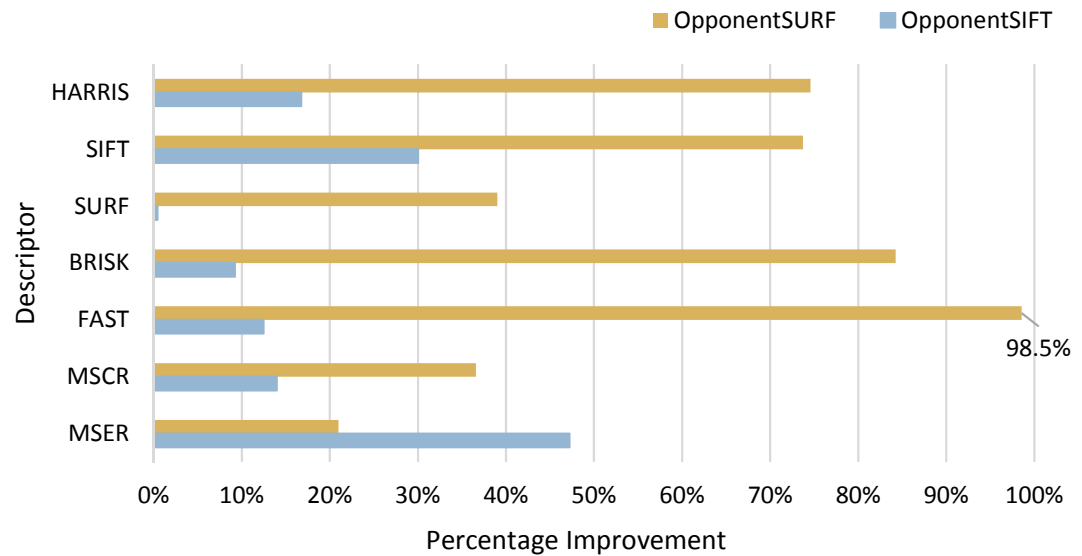


Figure 4.9: Improvements of OpponentSIFT and OpponentSURF colour descriptors using our combinatorial descriptor and distance measure. Orange bars show percentage improvements of OpponentSURF descriptors using our method, and blue bars show improvements in OpponentSIFT. The baseline uses standard descriptors with Euclidean distance measures in feature space. The overall average improvement across all of the feature descriptors in this test was 95.2%.

rally improves matching with this descriptor by up to 47.2% depending on the initial feature detector (Figure 4.9).

Overall the average improvement across all of the colour feature descriptors in this test was 39.8%.

Matching with colour variations

The representation of colour of an object within an image changes with many factors such as illumination, camera, distance, and weather. Our method is invariant to illumination changes by its analysis of Hue and Saturation in the HSV colour space and the quantisation of colour to a fixed palette. The clarity of colour is sensitive to the distance between the object and the camera, and distant objects begin to appear overwhelmingly grey (Figure 4.10).

In video sequences such as these, the colour quantisation to the fixed palette converges to a spike of grey pixels which subsequently reduces the performance of the colour boost algorithm. Instead, such nearly-grey images can be processed using only the Hue channel of the HSV colour space to quantise the colours to the fixed histogram and accentuate the dull colour. The resulting histogram provides increased colour information which is



Figure 4.10: Clarity of colour is sensitive to the distance between an object and the camera, and distant objects begin to appear overwhelmingly grey.

more discriminative than grey.

We have tested combinations of channels for quantising colour to the palette, and found that performance is optimal on a general set of images when the Hue and Saturation channels are used. However, in poor imaging conditions, performance can be improved by using only the Hue channel. The images in Figure 4.10 are taken from a short video sequence of 486 frames. The improvements that were gained using only the Hue channel to generate correspondences of the green checked shirt (highlighted) are shown in Figure 4.11. The discriminative nature of the colour boosting method is clearly demonstrated with two key performance measures. While *true positive* correspondence of features was reduced by 2-4%, which typically represents only one or two features, the number of False Positive matches reduced by up to 9%. The improvement of the overall F_1 -measure, plotted in green against the secondary axis, shows a maximum improvement of 100%, and in only twelve frames the F_1 measure was reduced.

An intelligent implementation can adjust which channels are used for the colour quantisation based upon real-time analysis of the resulting histogram. If HS channels yield a histogram that is biased to grey, then the quantisation calculation should be repeated with only the Hue channel, H .



Figure 4.11: Difference in F_1 accuracy using only the Hue component on near-grey images. The slight reduction (2-4%) in True Positive values (blue) is compensated by a larger reduction (up to 9%) in the False Positive rate (orange). The overall F_1 measure is consistently improved in 92% of the 145 frames (green).

State of the Art Colour Descriptors

There have been a number of colour descriptors proposed, and we compare state-of-the-art colour SIFT descriptors to our extension applied to intensity SIFT descriptors. SIFT descriptors are extracted from the images at feature positions detected by our trial set of detectors; Harris Corners, SIFT, SURF, BRISK, FAST, MSCR and MSER. We apply our combinatorial extension to these SIFT descriptors and measure the F_1 score on our test dataset. A distance ratio filter (Equation (2.17)) is applied to ensure we are comparing the robust matches in all cases. The F_1 scores are then compared with those achieved on the same dataset using state of the art colour SIFT descriptors HUE-SIFT, RGB-SIFT, C-SIFT, HSV-SIFT and RG-SIFT (Figure 4.12) using the implementation of [61]. HUE-SIFT has dimension $D = 165$ and all the others have $D = 384$. Our combined descriptor is $D = 138$, based on $D = 128$ SIFT with a 10 bin colour extension. The F_1 score of the original descriptor is shown alongside the F_1 score of our combined descriptor. In all cases apart from features detected by MSCR, the combined descriptor shows a large increase in F_1 over the original descriptor.

A combinatorial extension to SIFT descriptors of SIFT features shows the largest improvement, and comes close to matching the accuracy of C-SIFT, which is three times slower to compute and nearly three times the size. SIFT descriptors extracted from BRISK

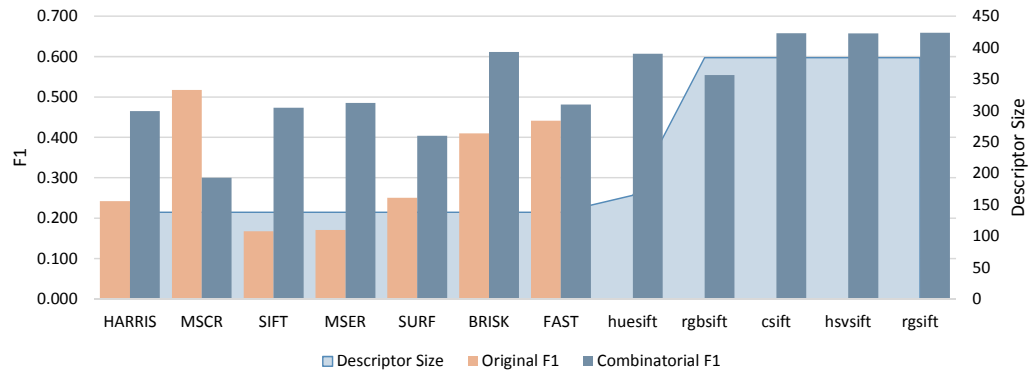


Figure 4.12: Comparing the F_1 score of state-of-the-art colour descriptors with our combinatorial method

features is a significant result for our application as BRISK is a high performance detector and with our combinatorial extension, we achieve an F_1 score that improves on HUE-SIFT and RGB-SIFT and comes close to the others.

Selection of an appropriate feature detector and descriptor is difficult, and the best performing is not universal across all images or all applications. Our method significantly improves the F_1 measurement of accuracy using fast-to-compute detectors to match or exceed state-of-the-art colour descriptors with much lower memory requirements.

4.5.3 Feature matching results

The graphs in Figure 4.13 summarise the results from our test database of 251 images. Each graph shows the F_1 measure of one of our seven selected feature detectors and all four of the feature extractors, comparing the matching performance of four methods of calculating correspondence. The pale blue line shows SIFT features extracted from each of the feature keypoints or region centers, the orange line shows rootSIFT features, SURF is in grey, and colour features of OpponentSIFT and OpponentSURF are in yellow and dark blue respectively. Each of the four methods are represented on the x-axis; the *original* correspondence using Euclidean distance of unmodified feature descriptors is the baseline against which we measure performance improvements. *Blur sensitive* applies the blur sensitive feature detection algorithm using unmodified feature descriptors. *Combinatorial* results are those achieved in using the combinatorial texture and colour feature matching descriptor extensions and matching algorithm, and finally *Combinatorial Blur sensitive* are results from the combined methodology described in this chapter.

The upward left-to-right trend in each of the graphs demonstrates the improvement in matching performance that is achieved with each of our method's components, and the



Figure 4.13: Summary of the results of feature matching with each component of the method, and our combined methodology. Each graph shows results from a different feature detector, and compares results with each descriptors using four methods; *Euclidean* – Euclidean distance of unmodified feature descriptors is the baseline, *Blur sensitive* – blur sensitive feature detection algorithm using unmodified feature descriptors, *Composite* – composite texture and colour feature matching descriptor extensions and matching algorithm, *Composite Blur sensitive* – the combined methodology described in this chapter.

combined methodology. The consistent closeness of the orange and yellow lines in the *Combinatorial Blur sensitive* result is particularly striking. The performance of our method using rootSIFT descriptors (128 + 10 dimensions) closely matches the performance of the much larger OpponentSIFT 384 + 10 dimension descriptor and significantly outperforms state-of-the-art feature matching using the OpponentSIFT 128D descriptors with the Euclidean distance measure.

4.5.4 Storage efficiency vs. matching performance

The choice of feature detector to use in the initial step of the processing pipeline significantly affects the ability to match features across images. The variability of matching accuracy is observable in the results presented in this chapter. Systems attempting to match features across a high volume of images are becoming increasingly common, and a key consideration for such systems is the storage efficiency of the descriptors used and the trade-off between storage and accuracy.

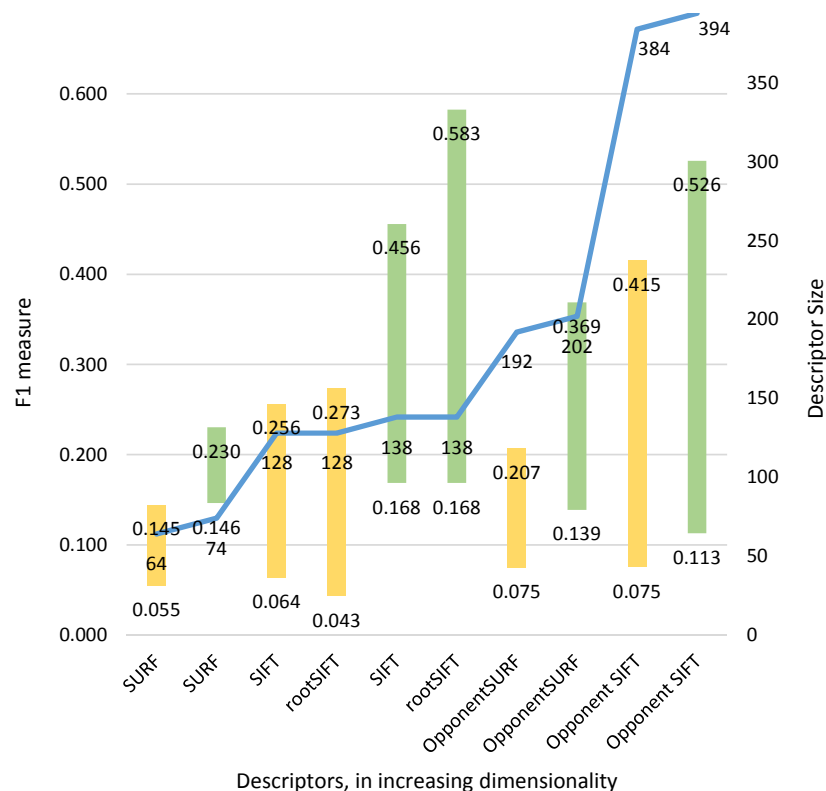


Figure 4.14: The correlation between F_1 matching accuracy (y-axis, left) and descriptor size (y-axis, right). Yellow bars show measures for established descriptors and Green bars are F_1 measures using our method. Using our method with SIFT and RootSIFT 138D combinatorial descriptors out-perform OpponentSIFT and OpponentSurf descriptors of almost 3 times the size.

The accuracy of feature matching using contemporary techniques generally increases in line with the size of the descriptor that is determined by a choice of feature extractor. Figure 4.14 demonstrates this where the tops of the bars represent the peak performance on each descriptor, and the yellow bars of established descriptors are generally higher moving left-to-right. The green bars show the F_1 matching accuracy using our method, where there is a peak in matching accuracy at dimensionality $D = 138$ where our method using SIFT and rootSIFT descriptors outperforms all other state-of-the-art descriptor matching using Euclidean distance measures. The saving in storage using our *Combinatorial rootSIFT* over the performance-comparable *Combinatorial OpponentSIFT* is $394 - 138 = 256$ values per descriptor.

4.6 Conclusion

This chapter has introduced a methodology for improved discriminative feature correspondence in low-quality images, with an emphasis on storage optimisation and runtime complexity. Our efficient and generic extension for feature descriptors improves the performance of feature matching and the blur sensitive feature detection method further enhances feature matching performance. We have shown the flexibility of the approach by applying it to five common keypoint descriptors and two popular region descriptors and we have compared the performance of all of them in matching features between images that vary in quality and appearance. Our experiments have demonstrated that the introduction of colour information to the feature descriptors, a unique feature distance measure and compensating for inter-image blur differences can improve the matching accuracy over the original descriptors in most combinations that were tested, even where the colour detail is visually subtle in poor quality images.

The method provides flexibility as it can be used with any feature descriptor extracted from any keypoint or region detector. Further, evaluation of the method in our problem domain of frame-to-frame feature tracking in low quality videos has demonstrated that smaller descriptors that are computationally lighter can be used to out-perform larger and more expensive feature descriptors. Our experiments have demonstrated an accuracy in matching features that out-performs all state-of-the-art methods using descriptors of less than 36% of size of the nearest performing colour descriptor.

The gains shown with this method are incremental, and while useful in improving correspondence in individual feature sets, do not represent exponential advancement in large scale pattern matching. In the next chapter, we broaden out consideration, and introduce scalable method that shows promise for greater advancement.

Chapter 5

Random Forests for pattern indexing

There has been a lot of research into detecting low level feature interest points, describing those interest points with feature descriptors (§2.1.2), and corresponding similar features from different images (§2.1.4), as we saw in Chapter 4. Finding patterns in a corpus of data extends feature matching to search a database of images – usually represented by their feature descriptors – for correspondences to a previously unseen query image. This indexing is well-researched (§2.1.5), but not a solved problem because of the problems of high-dimensionality of feature descriptors and the volume of images and feature descriptors therein (§2.1.3). In a criminal investigation, distinctive patterns can be identified by investigators which then because a *search query* to identify the same pattern in other video sequences. Some examples of distinctive patterns are tattoos, clothing combinations (Figure 5.1c,e), or baggage (Figure 5.1d).

Inspired by the work of Gall et al. [92], we look to random decision forests to provide a learned method to make sense of the large volume of data. In this chapter, we first investigate the feasibility of using a random forest for pattern detection by training a forest with very few samples of the pattern to be detected, which is representative of our online query-by-example use case. We then move to a novel method of training a decision forest offline to encode the patterns contained within a video which can be searched very quickly for any unseen pattern.

5.1 Minimal training datasets

Machine learning algorithms consist of two phases, 1) training a *model* to represent a complex relationship between the input data and a set of desired results, and 2) applying the trained model to a set of unseen data to predict a result. The training phase is usually an offline task that processes a lot of data and takes many hours. Some applications have constraints that prohibit the use of such a large-scale training scheme, but can benefit from using a learned model. Specifically, our interest in Query-by-Example video search falls into this category, where a user draws a rectangle on the screen to specify an image pattern to be found in a database of videos and images (Figure 5.1). From this rectangle, a set of training data must be found to train a (machine learning) model on demand. The data set of training images is necessarily small and the time to train must be minimised; the application is interactive – *online* – and a user’s expectation is that results will be available within a short time frame. In our use case, the end-user is a part of a police investigation team, and their tolerance to waiting for a result is somewhat greater than a typical user of a web search such as Google. Typically, if a result is returned within a few seconds, then the user will be happy as such a delay doesn’t distract from their workflow.

Random forests are highly configurable A+ predictors, but the mechanism by which they produce a prediction is difficult to understand [207]; “[t]rying to delve into the tangled web that generated a plurality vote from 100 trees is a Herculean task.” Introduced in 2009, Hough Forest – a form of random forests trained on image patches – have gained interest in object detection [208, 209]. Some work has improved the runtime performance

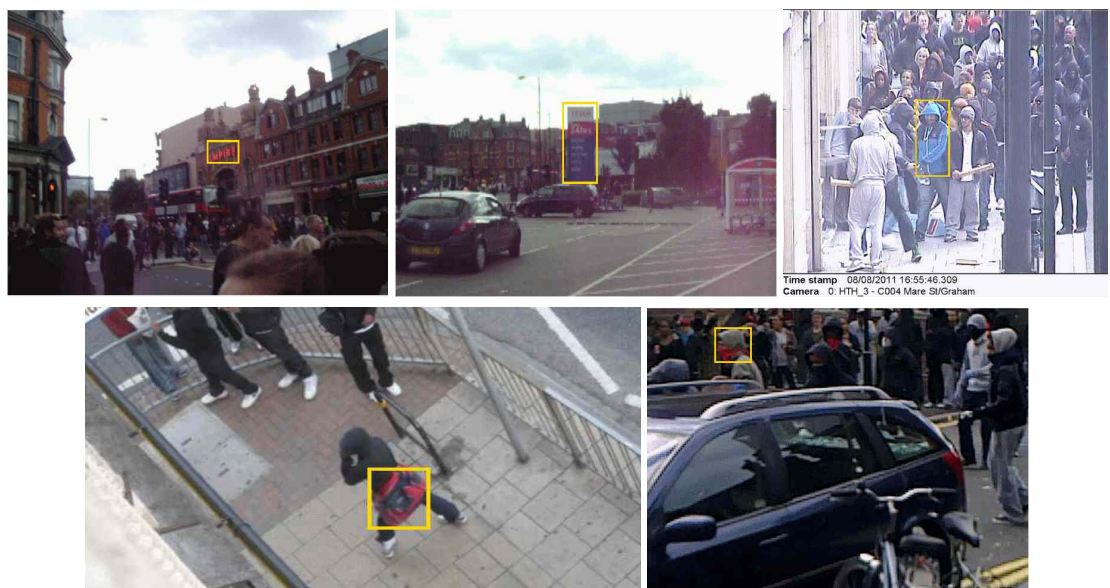


Figure 5.1: (a–e) Pattern queries used in five videos in our experiments

in voting [210], however to the best of our knowledge, no comprehensive study has been made of the impact of parameter tuning on runtime performance or accuracy of pattern detection. We empirically assess the impact of varying a number of configuration parameters (Table 5-A, page 86), measuring the training time, pattern detection time and precision against a manually-created ground truth. We assess the viability of Hough Forests to detection tasks when they are trained with minimal data from a simple initialisation where the user defines a rectangular *query region* as the single input into the system (Figure 5.1).

5.1.1 Discovery of training data through temporal pattern tracking

The query region is tracked through the video using template matching [211], calculating a similarity measure γ that we use as a confidence of tracking accuracy. The template is not updated during the tracking, so avoiding template drift [212] as every frame is matched to the original region. Two thresholds are used during tracking; ε_r to accept or reject the tracked region as a potential training image, and ε_t to terminate the tracking, where $\varepsilon_t < \varepsilon_r$. Let γ_m represent the maximum Normalised Cross Correlation found in the image using a sliding window. If $\varepsilon_t \leq \gamma_m < \varepsilon_r$ then the region is too dissimilar to the query region to be used as a positive training image, but may represent a partial occlusion. In this case, it is rejected but tracking continues so that the correspondence may be re-established in a later frame. If $\gamma_m < \varepsilon_t$, the tracking is terminated because the found region is so dissimilar to the query region that it is unlikely to be a partial-occlusion from which we can recover in a subsequent frame. If $\gamma_m \geq \varepsilon_r$ then the frame image is stored, along with the region bounding box (x_1, y_1, x_2, y_2) and γ_m . The value of ε_t adjusts the system's sensitivity to partial occlusions; a higher value will include more frames in the candidate list that have a lower similarity to the original search pattern, but which will likely be excluded by the next algorithm steps. A higher value of ε_t therefore also increases the runtime complexity and consequently introduces a longer runtime.

When the tracking terminates, we have a list of candidate training images. The process is repeated in a reverse direction to extend the list to include images from before and after the query frame. The candidates are sorted into descending order of γ_m and the n^\oplus images with the highest values of γ_m are selected as positive training images. Figure 5.2 shows an example, with $n^\oplus = 10$ and some of the positive images that were rejected as training images. Increasing n^\oplus selects more images from the left of Figure 5.2b into Figure 5.2a. Each image increases variation from the template which helps to generalise the forest but increases the time required to train.

The tracked regions in the n^\oplus frames are extracted as positive training images $P_i =$



Figure 5.2: Tracked regions are extracted before and after the query frame. (a, Top) The 10 *best matching* regions are selected as positive training images ($n^{\oplus} = 10$). (b, Bottom) Rejected candidate positive training images, based on the values of $\gamma_r = 0.16$ and $\gamma_t = 0.1$. Images are sorted in descending γ_{m_i} ; varying n^{\oplus} will select more or fewer from the left of (b) into (a).

(x_1, y_1, x_2, y_2) . Negative training images are extracted from the rest of the image; the *background*. First, up to four background regions are defined from the background surrounding P_i in the image

$$\begin{aligned}
 B_i^1 &= I_i(0, y_1 - 1, x_1 - 1, y_2 + 1) && \text{the area to the left of } P_i \\
 B_i^2 &= I_i(0, 0, w - 1, y_1 - 1) && \text{the area above } P_i \\
 B_i^3 &= I_i(x_2 + 1, y_1 - 1, w - 1, y_2 + 1) && \text{the area to the right of } P_i \\
 B_i^4 &= I_i(0, y_2 + 1, w - 1, h - 1) && \text{the area below } P_i
 \end{aligned}$$

where w and h represent the width and height of P_i . Up to two of these images may have no area (be empty) if the region is against an edge of the image. The non-empty images $B_i^n, 1 \leq n \leq 4$ are then used as negative training images.

We train a forest using features from our set of training images using code from the original authors [166] available online at https://github.com/cdmh/hough_forests/tree/master.

5.1.2 Hough Forest parameters

Many parameters can be changed to tune the time taken to train a forest and apply it (Table 5-A), in our case to pattern detection. Training parameters determine the shape and size of the forest and *how trained* the forest is. Feature channels are created from each training image, and from each feature channel a set of $\varrho_x \times \varrho_y$ patches are extracted from random positions. The number of patches, τ_m , that are extracted often goes unmentioned in the literature; in the implementation from [166], $\tau_m = 50$ patches and the paper describes resizing each image such that the longest spatial dimension is 100 pixels. We propose to optimise this parameter based on the shape and size of the query region,

Table 5-A: Training parameters of a Hough Forest used to balance performance and accuracy

Parameter	Description
τ_t	number of trees
τ_d	maximum depth of each tree
τ_k	stop growing when number of patches in a node $< \tau_k$
n^\oplus, n^\ominus	number of positive and negative images used to train the forest
τ_m	number of patches to extract from each training image
ϱ_x, ϱ_y	dimension of patches extracted from training images

without resizing the images. Given that patches are extracted from random positions within a training image, it follows that the quantity and size of patches as well as the size of the training image determine the coverage. A large number of patches in a small training image will cause a lot of overlap, leading to redundancy in the forest, bloating the trees and reducing runtime performance. An optimum coverage of each positive training image of dimension $w \times h$ by patches $\varrho_x \times \varrho_y$ is achieved with

$$\psi = \left\lfloor \frac{wh}{\varrho_x \varrho_y} \right\rfloor \quad (5.1)$$

non-overlapping patches.

It is our intuition that a correlation exists between the accuracy of a Hough Forest in detection and the shape of the query region. To our knowledge, this consideration has not been reported in previous literature, and we therefore investigate the hypothesis. We introduce a computed dimension for patches based on the aspect ratio of the query region, and consider two alternative means by which to calculate ϱ_x and ϱ_y . First, we scale the patch so that the minimum dimension is 16 [92],

$$\begin{aligned} \varrho_x &= \left\lceil \frac{16w}{\min(w, h)} \right\rceil \\ \varrho_y &= \left\lceil \frac{16h}{\min(w, h)} \right\rceil \end{aligned} \quad (5.2)$$

and second, we restrict the patch size to be within a 16×16 array and adjust the aspect ratio, thus reducing one of the dimensions

$$\begin{aligned} \varrho_x &= \left\lceil \frac{16w}{\max(w, h)} \right\rceil \\ \varrho_y &= \left\lceil \frac{16h}{\max(w, h)} \right\rceil \end{aligned} \quad (5.3)$$

where $\lceil \cdot \rceil$ represents the integer *ceiling*.

5.1.3 Empirical Evaluation

Our experiments use five videos from different scenes (Figure 5.1, page 83) that include footage taken by hand-held cameras – videos 1, 2 and 5, Figure 5.1 a, b and e – and operator-controlled *Pan, Tilt and Zoom* street cameras that contains a lot of blur caused by fast camera movements (Videos 3 and 4, Figure 5.1 c and d). One query region was selected per video, as shown.

Defining ground truth is subjective and somewhat imprecise, so was carried out in a research group with cross-validation and peer-review. We define 16 query patterns as a baseline assessment. Ground truth annotations are defined per-frame to give the most detailed temporal description possible [10], in ViPER format XML files with an xgft file extension. ViPER XML is a common format from the ViPER annotation tool [213] used by Etiseo [214] and PETS¹.

We ran experiments with the parameter values shown in Table 5-B, and Table 5-C shows the calculated number of patches $\tau_m = \psi$ and calculated patch sizes $\varrho_x \times \varrho_y$ for each video. Starting with a baseline experiment, we performed further experiments

Table 5-B: Parameter values used in our experiments (baseline values from [166] are highlighted)

τ_t	$\in \{2, 3, 4, \mathbf{5}, 10, 15, 20\}$
τ_d	$\in \{5, \mathbf{15}, 20, 25\}$
τ_k	$\in \{5, 15, \mathbf{20}, 25\}$
n^{\oplus}	$\in \{\mathbf{10}, 15, 20, 25\}$
n^{\ominus}	$= 5n^{\oplus}$
τ_m	$\in \{\psi, 25, \mathbf{50}, 75\}$
ϱ_x, ϱ_y	$\in (9, 9), (\mathbf{16}, \mathbf{16}), (25, 25), (36, 36), \text{Equation (5.2), Equation (5.3)}\}$
τ_s^n	$\in \{0.5, 1.0\}$
τ_r^n	$\in \{1.0\}$

Table 5-C: Five videos are used in our experiments, with a total of 6 887 frames. Each video has different query region dimensions. The calculated number of training patches, $\tau_m = \psi$ (Equation (5.1)) and adaptive patch dimensions for two constraints Equation (5.2) and Equation (5.3) are shown respectively.

	Dimensions	Number of frames	Query region dimensions $P_x \times P_y$	$\tau_m = \psi$ equation (5.1)	$\varrho_x \times \varrho_y$ equation (5.2)	$\varrho_x \times \varrho_y$ equation (5.3)
Video 1	640 × 480	268	54 × 39	8	22 × 16	16 × 11
Video 2	640 × 480	743	58 × 135	30	16 × 37	6 × 16
Video 3	704 × 625	427	73 × 165	47	16 × 36	7 × 16
Video 4	691 × 360	3 585	74 × 75	21	16 × 16	15 × 16
Video 5	640 × 480	1 864	61 × 58	13	16 × 16	16 × 15
		6 887				

¹<http://www.cvg.reading.ac.uk/PETS2016/>

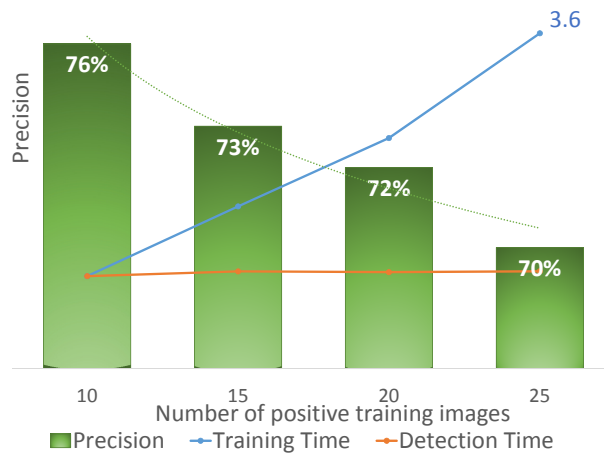


Figure 5.3: Performance and precision change with the number of training images n^{\oplus} . No of Negative training images is $n^{\ominus} = 5n^{\oplus}$. The logarithmic trend of *precision* is shown as a green dotted line, decreasing as n^{\oplus} increases.

varying one parameter value at a time. We measured the time taken to train the forest and to apply pattern detection. Precision is calculated thus: 6 887 frames of video have been manually inspected to create a ground truth of the position or absence of the pattern within each frame. The position is defined by a rectangle of the dimensions of the query region. Detection is applied to each frame in which the ground truth identified the pattern to be present. The intersection area of the detected region and ground truth is divided by the area of the query region, and if the ratio is ≥ 0.8 then a correct detection is recorded.

5.1.4 Results

The number of training images, n^{\oplus} , is an obvious parameter to consider. There is, unsurprisingly, a strong correlation (0.994) between the number of training images and the time taken to train the forest (Figure 5.3, blue curve). The precision drops steeply as n^{\oplus} increased from the baseline 10 to 15 and 20. This can be explained by the *correctness* of images found during tracking. Training images are added in descending order of their cross correlation to the original query region. As more training images are added, these images match less well, and begin to introduce poor images into the training set. Hough Forests are good at handling noisy training data, and training with some noise helps to generalise the classifier, but there comes a point where the training images are too dissimilar to be useful; there is a correlation of -0.993 between the number of trees and the precision achieved. In most of our experiments the detection time (orange curve) is within 10% of the 1 second baseline, therefore the influence of parameter tuning does not materially affect the timing. We do not comment further on the detection performance

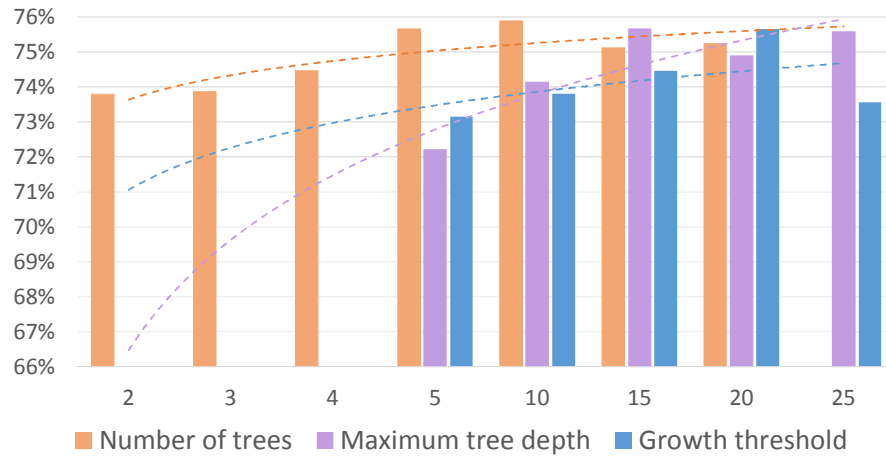


Figure 5.4: Precision measured with each parameter affecting the size and shape of the forest; Number of trees τ_t , Maximum tree depth τ_d , Growth threshold τ_k . Dotted lines show logarithmic trend as each parameter is increases.

for each test unless it is outside of this margin.

Parameters τ_t , τ_d and τ_k affect the size and shape of the forest; the number of trees, the maximum tree depth, and the growth threshold, respectively. Figure 5.4 shows the results of our experiments in changing each of these parameters individually. Time to train a forest is linear with respect to the number of trees in the forest, τ_t . Trees can be trained independently from each other, and so the task is appropriate for parallel processing. The depth of a forest increases memory requirements exponentially; the maximum number of leaves in a tree of depth τ_d is 2^{τ_d} . An example using the code of [166] and a forest with $\tau_d = 25$, consumed 11Gb of memory. The baseline $\tau_d = 15$ produces the highest precision in 4 of 5 videos in our experiments. Precision is increased by 1% for Video 4 with $\tau_d = 20$, but Video 3 decreases to 63% from 68% with the baseline.

Each tree in the forest stops growing when a threshold τ_k number of patches is reached. Varying the value of τ_k from the baseline 20 reduced the precision, or left it unchanged in all of our experiments. τ_m patches are extracted from each training image. Varying τ_m from the baseline $\tau_m = 50$ to other constant values does not improve precision at all.

Our novel method using a dynamic number of patches relative to the size of the query region, Equation (5.1), produces interesting results (Table 5-D). There is a very strong correlation (0.98) between the training time when $\tau_m = \psi$ and the area of the query region ($width \times height$). In 4 of the 5 videos the training time reduced considerably and was unchanged for Video 3 (column 3). The detection time is unchanged or marginally faster (not shown), but the variance in precision is less consistent. For three of the five videos, the precision drops 1 to 4 percentage points, for Video 2 it remains the same and

for one of the most challenging videos, Video 5, the precision increases to 57.9% from 57.1% baseline.

We calculate the patch size $\varrho_x \times \varrho_y$ and the number of patches to extract from each training image, τ_m based on the size of the query region and compare results from combinations of fixed and dynamic values for all of these parameters; $\tau_m = \psi$ from Equation (5.1) and two calculations of ϱ_x and ϱ_y from Equation (5.2) (columns 5 and 7), and Equation (5.3) (columns 4 and 6). Finally, we constrain the patch size to a minimum value of 12 in Equation (5.3) and show the results in column 8. The highlighted values in Table 5-D indicate improvements in training runtime performance and precision across the five videos. Column 4 shows that the best overall performance is achieved with $\tau_m = 50$ using Equation (5.3) to dynamically calculate the patch size, with improved accuracy on 4/5 videos and faster training time in 3/5 videos with one unchanged and only one increasing.

5.1.5 Conclusion

Contemporary literature use a fixed number of 50 square patches of 16×16 for Hough Forest training and its applications, per [166] and our baseline. We have demonstrated that while using a very small training set of images, the accuracy of pattern detection is sensitive to patch size, aspect ratio, and the number of patches that are used. Our experiments have shown that using an adaptive patch size influences both the training time and precision results. Each of our test result sets bar one (Table 5-D column 7) shows increased precision in 3 of 5 videos, and many show reduced training time, too. We consider the variation of patch quantity and dimension to be a interesting area

Table 5-D: Precision results of dynamic patch sizes. Highlighted values indicate improvements in training runtime performance and precision across the five videos. Timings are reported as relative multipliers to the wall clock times of the baseline configuration to ease comparison.

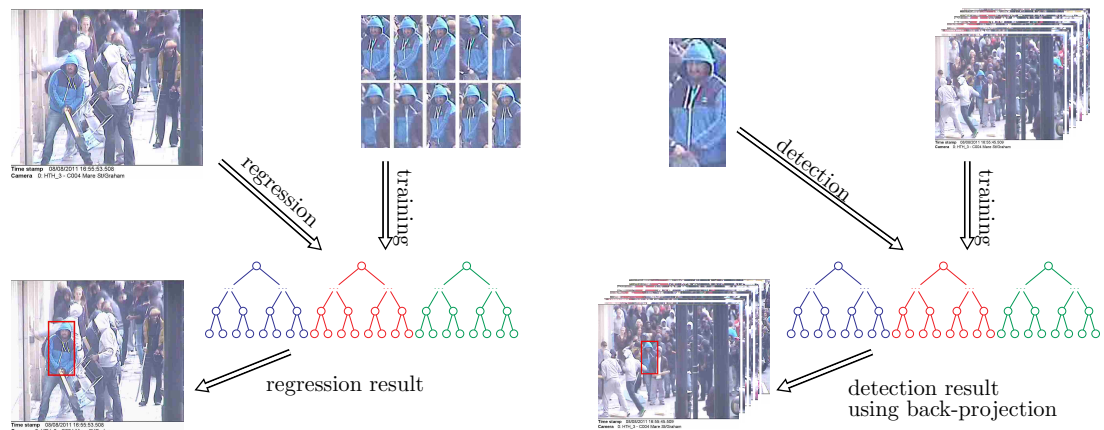
	baseline	$\tau_m = \psi$	$\tau_m = 50$ $\varrho_x \times \varrho_y$ Equation (5.3)	$\tau_m = 50$ $\varrho_x \times \varrho_y$ Equation (5.2)	$\tau_m = \psi$ $\varrho_x \times \varrho_y$ Equation (5.3)	$\tau_m = \psi$ $\varrho_x \times \varrho_y$ Equation (5.2)	$\tau_m = \psi$ $\varrho_x \times \varrho_y$ Equation (5.3) $\min(\varrho_x, \varrho_y) = 12$
Training							
Video 1	1.0	0.1	1.1	1.2	0.1	0.0	0.1
Video 2	1.0	0.4	0.9	1.1	2.5	0.2	0.3
Video 3	1.0	1.0	1.0	1.1	2.6	0.3	0.7
Video 4	1.0	0.3	0.9	0.9	0.3	0.3	0.3
Video 5	1.0	0.2	0.9	0.9	0.2	0.2	0.2
Precision							
Video 1	100.0%	98.7%	100.0%	98.7%	98.7%	96.0%	98.7%
Video 2	98.9%	98.9%	99.5%	99.5%	99.5%	97.3%	99.5%
Video 3	67.9%	67.4%	58.5%	54.8%	64.2%	45.4%	58.5%
Video 4	54.4%	49.9%	58.8%	55.4%	58.4%	49.9%	58.4%
Video 5	57.1%	57.9%	57.9%	57.5%	58.3%	57.9%	58.3%

and encourage further study of how our results can be generalised in detection and classification tasks. However, in a very large corpus, it is impractical to apply forest detection to every image and video frame, even with a fast and lightweight training scheme. The detection process therefore does not scale sufficiently to provide for a user-interactive online search, and we do not pursue this further in our study. Rather, use the knowledge gained to inspire our next contribution.

5.2 Random Forests for a video database index

We use the knowledge from §5.1 that a Hough Forest can be trained effectively for pattern detection with only a small amount of data to further investigate if a forest can discriminate specific patterns that appear in only a few images of a larger set; a typical needle-in-a-haystack search.

Our pattern detection method builds on preceding Hough Forest literature that have demonstrated success in common computer vision tasks such as tracking [215] and action recognition [216], as well as robotics [217], facial expression recognition [218], medical imaging [219] and particularly object detection [92]. Object detection using forests gene-



(a) A typical forest training and regression schema. The forest is trained using random patches from images depicting an object or pattern and a regression is run on a query image to detect the object or pattern.

(b) Our schema inverts the training data and pattern image. The forest is trained using random patches from frames of a video sequence, and testing is run on a query image containing the object or pattern to be searched for. Using back-projection, one pass of the pattern through the forest simultaneously identifies the pattern in all frames of the video.

Figure 5.5: A typical forest training and regression process (left) and our novel process for video indexing (right).

rally, such as in [92, 220, 221], is a supervised learning task based on training using exemplar representations of an object class to be detected in unseen images. However, query-by-example pattern detection in a corpus of images (video frames) does not afford the luxury of prior knowledge of the pattern, and is an inverse problem; the search corpus is known *a priori*, but the pattern to detect is not. To this end, we propose a departure from established methods of forest training, and conceptually invert the use of the forest. We consider the image domain to consist not of instances of an object class or variations of the pattern to be detected, but the corpus *to be searched*. In our case, a forest's image domain is a set of frame images from a single video, or set of related images, within a corpus. The change to conventional use of random forests is illustrated in Figure 5.5. The new schema results in a very fast search of a large set of data whereby a single pass of a small query image through the forest yields probabilistic hypotheses of pattern detection in every image in the training data.

We use a collection of forests to build a complete index of our video and image corpus. Each forest is trained using a single video (Figure 5.6) and can therefore be considered a sub-index of the database relating exclusively to a single video. Each forest $\mathcal{F} = \{\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \dots, \mathcal{T}_N\}$ consists of N trees where each tree \mathcal{T}_i is independent of all other trees for training and detection. Trees are trained per video, using keyframes for both negative and positive training. Each forest therefore represents a pattern index for a video, and the set of forests $F = \{\mathcal{F}_1, \dots, \mathcal{F}_f\}$ is synonymous with a database index file that can be used to search for patterns.

Forests are trained using a novel scheme to label training data as positive and negative samples (§5.2.1) which reduces the unsupervised learning task to a supervised learning problem without manual intervention (such as manually labelled training data) and

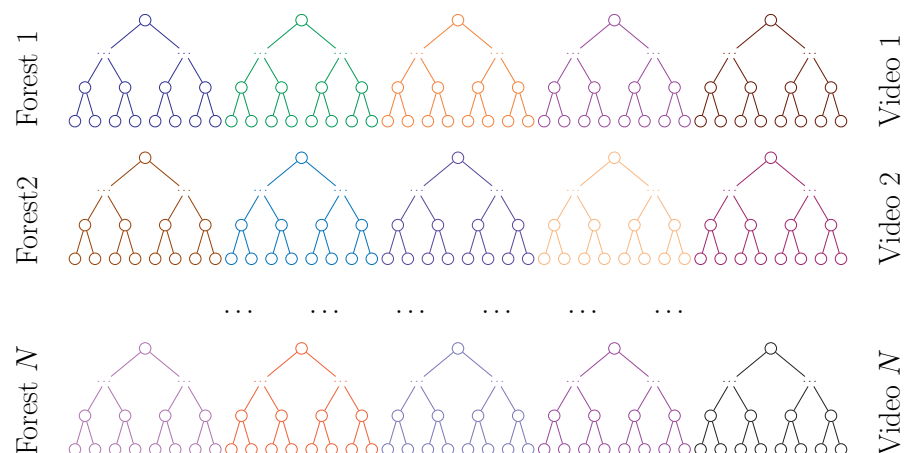


Figure 5.6: The video database index is a collection of independent forests. Each row represents a forest of five independent trees, trained using frames from single video sequence.

removes the need for an explicit set of negative training images altogether. Training is the most time-consuming function – consuming several hours of processing for each video – and can be done as an offline activity for each video, independently of all other videos in the index. Training each forest is therefore consistent with building a video index in a more conventional retrieval system. A trained forest provides fast access to the patterns contained within the video such that it is searchable on-demand for unseen patterns of any size and dimension.

We draw inspiration from Razavi et al. and their use of *back-projection* to recover information regarding the origins of the votes in Hough space to determine a bounding box of an object in the image domain [222]. To perform a search, patches are extracted from a query image (or sub-image defined by a query region) using a sliding window (a common technique in object detection [223, 224]) and filtered through each tree of the forest. At *arriving* at a leaf node, all patches held at the leaf are used to back-project a weighted vote into a high-dimensional accumulator space. The *support* of the leaf is used to trace the contributing vote back to the source frames, and the vote is accumulate in each of them (see §5.2.3 on page 100).

The independence of the components within a collection of forests is important for large-scale searching, providing scalability and flexibility.

Scalability To support large video database search, the index must be highly scalable. The independence of components in the forest collection means that algorithms to train and detect can easily be paralleled to extend processing across many cores, processors and machines as there is no dependence between individual trees. Pattern search is fast, but is easily scalable to gain potential increases in performance – each forest can pass the query image patches through all of its trees simultaneously and accumulate the results as they complete.

Flexibility New videos can be added easily without need for any re-training of existing forests. A new forest is simply created, trained with the new data and then added to the collection. If a video is no longer required to be searched, then the relevant forest can simply be removed from the collection and will no longer be included in future searches. No re-training is necessary. Where available, the date of the video can be added as a property of the forest to further increase performance and search results relevance. A user can specify a time frame and forests containing videos from outside of the range can easily be excluded from the search.

5.2.1 Training

There is enormous redundancy in videos when treated simply as a collection of individual frames [168]. To reduce data volumes, each forest is trained with keyframes of a video. In line with other researchers, e.g. [225–227], we sample keyframes at a fixed interval through each video. However rather than fixing a pre-defined constant interval, we calculate the interval according to the length of the video,

$$\text{interval} = \min \left(\frac{\# \text{ frames}}{100}, 100 \right) \quad (5.4)$$

This dynamic selection of an interval ensures that enough frames are included in the index without including too many keyframes so as to increase the size of the forest's trees and consequently increase processing time and storage requirements unreasonable.

Each tree is trained with a set of feature patch data $\mathcal{S} = \mathcal{S}^{\oplus} \cup \mathcal{S}^{\ominus}$, where $\mathcal{S}^{\oplus} = \{P_i(J_i^{a_i})\}$ and $\mathcal{S}^{\ominus} = \{P_k(J_k^{a_k})\}$ are sets of positive and negative training patches selected from a random choice of feature channel J^a from \mathbf{v} (Equation (2.20), page 38).

Extracting patches

A large number of patches, N , is selected, where N is derived from the dimensions of the images w, h and the patch size w_p, h_p thus

$$N = 2 \left\lceil \frac{w}{w_p} \right\rceil \left\lceil \frac{h}{h_p} \right\rceil \quad (5.5)$$

where $\lceil \cdot \rceil$ is the integer ceiling. Such a large number of patches ensures dense placement with a good coverage of the image area so that patterns from across each frame will be contained in the index. Selection of the patch position is random in three dimensions; (x, y, a) where x, y is the image plane co-ordinates and a is the feature channel.

Automatic patch labelling

To reduce the unsupervised learning problem to one of supervised learning, each patch is algorithmically assigned a label for positive or negative training, $c_i \in \{\oplus, \ominus\}$, determined by a texture saliency metric based upon triangulation derived from [228]. Each training image is padded to square dimensions and divided into two right-angled isosceles triangles which are then recursively subdivided half way along the hypotenuse, through the right-angle. Let $M(x, y)$ be the measured intensity value at co-ordinate (x, y) , and given a triangle with vertices $(x_1, y_1), (x_2, y_2), (x_3, y_3)$, let $\gamma_1 = M(x_1, y_1), \gamma_2 = M(x_2, y_2)$,



Figure 5.7: Triangular decomposition (column 2) searching for high-saliency areas. Negative training patches will be extracted from red regions in column 3, and positive training patches will be extracted from the unmarked regions. A randomly offset grid (column 4) of 16×16 patches placed over the image are labelled green squares as positive training patch positions and red are used as negative training positions (based on intersection of the mask in column 3).

and $\gamma_3 = M(x_3, y_3)$. A linear-interpolated approximation of the intensity of a point inside the triangle is then defined

$$R(x, y) = \gamma_1 + \alpha(\gamma_2 - \gamma_1) + \beta(\gamma_3 - \gamma_1) \quad (5.6)$$

where α and β are the barycentric coordinates, defined by

$$\alpha = \frac{(x - x_1)(y_3 - y_1) - (y - y_1)(x_3 - x_1)}{(x_2 - x_1)(y_3 - y_1) - (y_2 - y_1)(x_3 - x_1)} \quad (5.7)$$

$$\beta = \frac{(x_2 - x_1)(y - y_1) - (y_2 - y_1)(x - x_1)}{(x_2 - x_1)(y_3 - y_1) - (y_2 - y_1)(x_3 - x_1)} \quad (5.8)$$

The accuracy of the approximation is given by comparison with $M(x, y)$

$$|M(x, y) - R(x, y)| \leq \epsilon, \epsilon > 0 \quad (5.9)$$

if this condition is satisfied for every point (x, y) within the triangle, then the region is marked as low-saliency, and recursion is terminated. Figure 5.7 shows example frames of videos (column 1), the triangulation (column 2), labelling mask (column 3), and labelled patches (column 4). Randomly positioned patches that overlap the red area are labelled for negative training, $c_i = \ominus$, and all others are labelled for positive training, $c_i = \oplus$.

Node split functions

At each non-leaf node j in the tree, a *split function* binary test is performed on each patch appearance by comparison of some offset τ with θ parameters $h(\mathbf{v}; \theta) \rightarrow \{0, 1\}$ to determine whether to direct incoming patches to the left or right subtree, ${}^L\mathcal{S}_j$ or ${}^R\mathcal{S}_j$, such that $\mathcal{S}_j = {}^L\mathcal{S}_j \cup {}^R\mathcal{S}_j$. The goal of splitting the data is to increase the similarity of patches within a node as the depth of the tree increases. Therefore a constraint is place that ${}^L\mathcal{S}_j \neq \emptyset$ and ${}^R\mathcal{S}_j \neq \emptyset$. Candidate functions that do not satisfy this constraint are discarded.

A very simple pixel pairwise-comparison test is used in [166], consisting two feature values at locations $\mathbf{p} \in \mathbb{R}^2$ and $\mathbf{q} \in \mathbb{R}^2$ in feature channel a ,

$$h(\mathbf{v}; \theta) = \begin{cases} 0 & \text{if } I(\mathbf{p}) < I(\mathbf{q}) + \tau \\ 1 & \text{otherwise} \end{cases} \quad \theta = (\mathbf{p}, \mathbf{q}, a) \quad (5.10)$$

where $I(\cdot)$ is the pixel intensity at the given position. Such a simple test has the advantage of being very fast to execute, but many disadvantages. The comparison of patches

containing 256 values is reduced to the comparison of two randomly positioned pixels within the patches. By using such little information from the patch and ignoring 254 of the 256 values, the comparison is sensitive to salt-and-pepper noise and rescale interpolation artefacts. We choose a more complex method of patch comparison than that of Gall and Lempitsky [92], and accept an inevitable increase in runtime complexity in return for a more robust test function.

We use an *entropy* calculation of the *grey-scale co-occurrence matrix* [229] (GLCM) that is naturally aligned with the optimisation goal of maximising the *information gain* achieved in each subtree. The split node function is parameterised with θ consisting a patch bounding box $\mathbf{b} = (\mathbf{p}, \mathbf{q})$, $\mathbf{p} \in \mathbb{R}^2$ and $\mathbf{q} \in \mathbb{R}^2$ in feature channel a ,

$$h(\mathbf{v}; \theta) = \begin{cases} 0 & \text{if } H(\mathcal{C}(\mathbf{b})) < \tau \\ 1 & \text{otherwise} \end{cases} \quad \theta = (\mathbf{b}, a) \quad (5.11)$$

where $\mathcal{C}(\mathbf{b})$ is the GLCM matrix of the patch image at \mathbf{b} in feature channel a , and $H(\cdot)$ is the entropy,

$$H(X) = - \sum_{i=1}^{256} \sum_{j=1}^{256} p(x_{ij}) \log p(x_{ij}) \quad X = \mathcal{C}(\mathbf{b}) \quad (5.12)$$

Optimising the split function

A set of random split functions is generated, and the best is selected according to the greatest information gain ΔE , i.e. the reduction in uncertainty achieved by splitting the data at a node into child subsets [230]

$$h = \arg \max \Delta E \quad (5.13)$$

$$\Delta E = H(S) - \sum_{d \in \{L, R\}} \frac{|^d S|}{|S|} H(^d S) \quad (5.14)$$

$$H(S) = - \sum_{c \in \{\oplus, \ominus\}} p_c \log(p_c), \quad p_c = \frac{|S^c|}{|S|} \quad (5.15)$$

where H denotes class entropy and p_c is the fraction of S_j belonging to class c .

An exhaustive search for the maximum information gain is undefined in the context of using randomised tests for the node split function, and some stop condition is required for the search. The stop condition may be an absolute threshold of the minimum acceptable measured gain, a relative value to counter diminishing returns on effort and/or time, or simply a number of iterations (random tests) performed during a search.

Choosing an absolute threshold is difficult; it must be high enough so as to determine a good – or at least reasonable – split of data, and yet not so high that it is unachievable in a reasonable processing time. In the worst case, a too-high threshold may cause a deadlock of training where a suitable split function cannot be found to satisfy the condition of the threshold. A relative threshold – one that determines a search to continue until a measured gain is within a proportion of the gain at the parent node, can suffer similar dangers of impossibility. Commonly, therefore, the number of iterations of a search are limited to balance complexity and performance.

Most naïvely, a fixed number of iterations can be set in the algorithm, and at each node said number of random tests are performed and the test that results in the greatest information gain is selected. The published implementation¹ from [166] uses this technique. Random trees are hierarchical, with node splits being independently optimised at each level using data progressively more similar as the depth increases. Lepetit and Fua observed that by optimising the split less well at the root node of each tree, the correlation between trees is reduced [231]. Reflecting the hierarchical nature of trees, the authors further suggest a variable iteration limit, n , based on the current level of the tree, d ;

$$n = \begin{cases} 10, & \text{if } d = 1 \text{ (root node)} \\ 100d, & \text{if otherwise} \end{cases} \quad (5.16)$$

Randomising on tests is far more powerful than randomising on data [48], and we randomise on both.

Information Gain The information gain ΔE at each node j is calculated from the negative entropy $H(\cdot)$ and the cardinality $|\cdot|$ of the training sets at the node. Observe that each subtree consists of two subsets of data, labelled for positive and negative training. Let L_j represent the left subtree $L_j = {}^L S_j$ and $L_j^c \subset L_j$ be the subset for a given class label, thus $L_j^c = {}^L S_j^c$, $c \in \{\oplus, \ominus\}$ such that $L_j = L_j^\oplus \cup L_j^\ominus$ and $L_j^\oplus \cap L_j^\ominus = \emptyset$. L_j^\ominus may be the empty set. Where $|L_j^\oplus| < \varepsilon$, typically $\varepsilon = 5$ in our application, there is insignificant gain to be made by further division of a node. Further, let R_j represents the right subtree $R_j = {}^R S_j$ with all the same properties as L_j . Expanding Equation (5.14) and substituting Equation (5.15) for node j gives,

$$\Delta E(S_j) = H(S_0) + \frac{1}{|S_j|} \left(|{}^L S_j| H({}^L S_j) + |{}^R S_j| H({}^R S_j) \right) \quad (5.17)$$

¹<http://www.iai.uni-bonn.de/~gall/projects/houghforest/houghforest.html>

where S_0 is the full training data set. Given $S_0 = S_0^\oplus \cup S_0^\ominus$, then $H(S_0) = \log 1 = 0$,






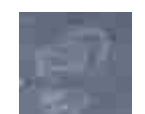




$$\Delta E(S_j) = \frac{1}{|S_j|} \left(|{}^L S_j| H({}^L S_j) + |{}^R S_j| H({}^R S_j) \right) \quad (5.18)$$

5.2.2 Reflection and Rotation Invariance

In Chapter 3, we discussed the importance of invariance to pattern appearances, specifically with respect to *reflection invariance* which is currently ignored in contemporary literature. Our choice of node split function using the GLCM entropy is a deliberate consideration of this requirement, and achieves invariance to rotation in 90° , 180° and 270° [232], and in reflection of the pattern.

The GLCM is a statistical method of examining second-order texture that considers the spatial relationship of pixels. GLCM texture is a two-dimensional histogram in which the relative frequencies $P(i, j, d, \Omega)$ of two pixels are recorded. The pixels are separated by distance d , occur in a direction specified by the angle Ω and have grey level intensities

Table 5-E: Examples of GLCM Entropy of the Nike logo seen in Figure 3.1. We show the GLCM Entropy value of the extracted logo pattern as it appears (“Original”), and in reflection and rotation, with the value relative to the original.

	Pattern	GLCM Entropy	Pattern	GLCM Entropy
Original		-53 867.1		-75 476.5
Mirrored		-53 867.1		-75 476.5
Rotated 90°		-53 867.1		-75 476.5
Rotated 180°		-53 867.1		-75 476.5
Rotated 270°		-53 867.1		-75 476.5

i and j [233]. We set $d = 1$, $\Omega \in \{0, 90, 180, 270\}$, tallying the co-occurrence of each pixel intensity and its neighbour in each direction up, down, left and right. The co-occurrence matrix is therefore invariant to rotation and reflection. We then calculate the entropy of the matrix using Equation (5.12), which produces a rotation- and reflection-invariant measure of randomness in the image.

We demonstrate the invariance in Table 5-E with the entire pattern, although in the forest trees, the GLCM entropy is used with individual patches of 16×16 pixels. The *Original* image in row 1 shows the pattern extracted from two original frames, as we described in Figure 3.1 on page 42. Row 2 shows the reflected image and Rows 3 – 5 show the original image rotated by multiples of 90° . For each of the 10 images, the GLCM Entropy is shown, and the entropy is equal for all variations of each original image. The consistency in these figures demonstrate the invariance to reflection and rotation such that the patches extracted from any of the variations will group into the same tree leaf, and yield a positive vote.

5.2.3 Pattern Detection

The trees of a trained forest represent the codebook of visual appearance, where each leaf node contains a set of patches¹, and a probability for those patches representing a foreground patch; positive classification, $c_k = \oplus$. During detection, a sliding window of 16×16 is passed over the query image, extracting patches at each position. The patch is passed through each tree of the forest, and a weighted vote is calculated for each patch;

$$w_i = \sum_{l \in \mathcal{L}} \frac{p_f}{|\mathcal{S}_l^\oplus| \cdot |\mathcal{L}|} \quad (5.19)$$

where p_f is the probability that a patch belongs to the foreground, shown in Equation (2.21) on page 39, and \mathcal{L} is the set of leaves at which the patch *arrives* within each tree, after being passed through the tree from the root node.

Votes w_i from the trees are back-propagated to the original position of the source patch using a Hough accumulation space, \mathcal{H} . \mathcal{H} is a high dimensional space of pattern hypotheses $\mathbf{h}(c, \mathbf{r}, f, \sigma)$, for class $c \in \{\oplus, \ominus\}$, originating from a patch at position (bounding box) $\mathbf{r} = (x_1, y_1, x_2, y_2)$, in frame f of the training dataset at scale σ . For each patch in the leaf, votes are cast into \mathcal{H} using parameters $(c = \oplus, x, y, f, \sigma)$. Rather than a single vote, multiple votes are cast over the original patch area, \mathbf{r} . For example, if a patch at $\mathbf{r} = (150, 100, 165, 115)$ in frame 6 at scale 1.0 has $w = 0.2$, then every 2D position

¹ patches with similar texture properties, independent of the feature channel, will share a leaf node, for example a patch from channel a and a patch from channel b that have similar GLCM entropies

$(150, 100, 6, 1.0) - (165, 115, 6, 1.0)$ is incremented by 0.2, therefore casting $16 \times 16 = 256$ votes. This is repeated for every patch from the sliding-window, accumulating a high-dimensional voting landscape of pattern position likelihoods.

Finding Hypotheses

Given the automatic labelling has produced $c = 1$ for foreground patches of high saliency, the subspace $P_f \subseteq \mathcal{H}$ at $\theta = (c = \oplus, f = f^*, \sigma = \sigma^*)$ represents likelihoods of pattern occurrences in each frame f at scale σ (Figure 5.8). All subspaces $\mathcal{P}_{f_i} \in \mathcal{H}$ within the constraints of θ can then be assessed for peak values of maximum likelihood of pattern occurrences in each frame f at each scale σ . Common approaches that search for peak likelihoods smooth the voting landscape with a large Gaussian kernel (perhaps 5×5) and identify the location of the maximum value as a detection hit. As our votes are cast in 16×16 patch-size regions, we take a different approach. No Gaussian smoothing is required, and we find a region the size of the query image at scale σ with the largest score consisting the sum of votes contained within the region.

Relative hypothesis scoring

Each image hypothesis score is independent. To produce a relative ranking for the pattern hypotheses across all images, we calculate a *Standardised score*, z , such that the absolute value of z represents the distance between the independent score x and the population

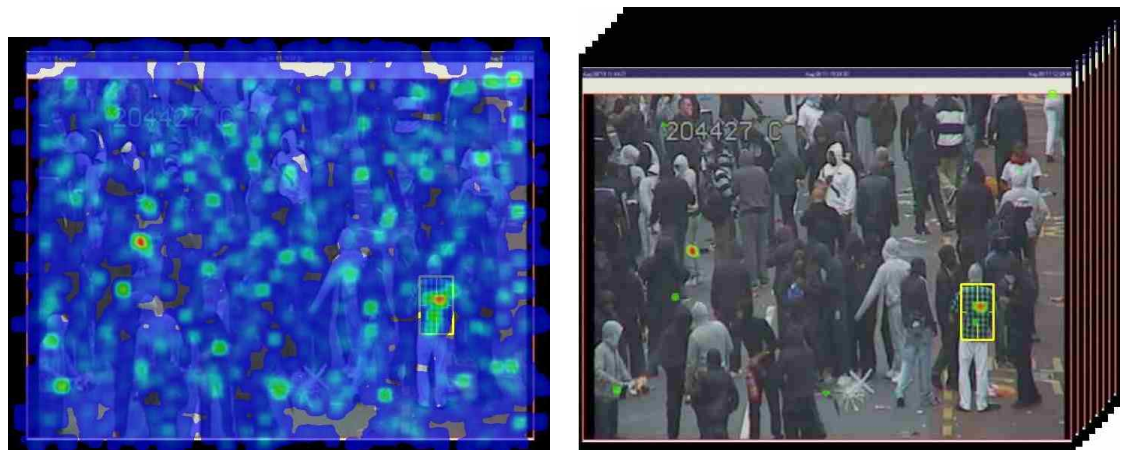


Figure 5.8: Back-projection of votes from the high-dimension voting space to the original video frames. Left, the subspace $P_f \subseteq \mathcal{H}$ at $\theta = (c = \oplus, f = f^*, \sigma = \sigma^*)$ represents likelihoods of pattern occurrences in each frame f at scale σ . Right, a peak is found in the votes that correctly identifies the pattern being searched.

mean μ in units of the standard deviation σ ;

$$z = \frac{x - \mu}{\sigma} \quad (5.20)$$

where the population is the set of strongest hypothesis scores from each image in the forest's dataset.

The Standard scores can be ordered in descending numerical order to produce a ranking result of pattern appearances in the video or image set. A further extension enables the standardisation to inter-forest ranking, providing corpus-wide relative ranking. Corpus-wide relative ranking is somewhat outside the scope of this thesis. While we inevitably consider ranking to some degree for use in the precision/recall analysis (§5.2.5), we recognise the need for further research to improve ranking results (§6.1.1), and we establish our method with good detection precision results.

Eliminating frame dimension bias

Votes are accumulated from random patches in the image space (§2.4.2) and the quantity of patches is a function of the constant dimension of images in the forest's dataset (§5.2.1). The dynamic calculation of the number of patches relative to the size of the image (Equation (5.5)) ensures that random patch placement provides good coverage over the image, but inadvertently introduces a bias. Larger images contain a greater number of patches, which can increase the strength of the accumulated vote not because the pattern is more likely, but because there are more votes to be cast from patches that overlap the pattern. This bias is, however, easily removed by dividing the unstandardised hypothesis score by the area (number of pixels) of the images in the dataset $z' = \frac{z}{wh}$.

5.2.4 Evaluation

We evaluate our method on a variety of videos that are representative of footage from fixed-camera street-scene CCTV cameras prevalent in the UK, Table 5-F. Ground truth was created by manual inspection, defining a tight bounding-box around the pattern in each keyframe in which it appeared. Ground truth was then verified by at least one other member of our research team.

Feature matching accuracy is typically measured using Precision-Recall [130]. *Precision* is the ability of the system to retrieve only relevant documents and *Recall* is the ability of the system to retrieve all relevant documents [234]. Precision is also known as *Confidence* or *True Positive Accuracy*, and Recall is also known as *Sensitivity* or *True Positive Rate* [235]. Receiver Operator Characteristic (ROC) curves are commonly used to present

results for binary decision problems, but can present an overly optimistic view of an algorithm's performance if there is a large skew in the class distribution [236], which can be misleading [237]. Precision-Recall (PR) curves [238] give a more informative picture of an algorithm's performance [236], and we therefore use these in our assessment.

Two precision and recall evaluations are used; for pattern detection and for retrieval. In the case of pattern detection – our primary focus in this thesis – accuracy is assessed by an overlap of bounding boxes of the detected region, A_d , and the ground truth region, A_{gt} , thus

$$recall = \frac{A_d \cap A_{gt}}{A_{gt}}, \quad precision = \frac{A_d \cap A_{gt}}{A_d} \quad (5.21)$$

In the case of retrieval, precision and recall are calculated with respect to True Positive (TP), True Negative (FN), and False Positive (FP) results ranked in the Top n results where $n \in \{5, 10, 20, 50, 100, 1000, 2000, 3000, 4000, 5000, 6000, 6554\}$ in a dataset of 6554 keyframe images. Defining Y number of *GT frames* to be the set of frames that are included in the ground truth as containing a given pattern, then

$$\begin{aligned} TP &= \# \text{ observed GT frames in the top } n \\ FP &= n - TP \\ FN &= \min(n, Y) - TP \end{aligned} \quad (5.22)$$

and

$$recall = \frac{TP}{TP + FN}, \quad precision = \frac{TP}{TP + FP} \quad (5.23)$$

Videos from the wild

The videos in our dataset were provided to our research group by the Metropolitan Police in London, and suffer all of the impurities that we described in our introduction (§1.1). The videos contain images of very busy street scenes (Figure 5.9) from which a number of query patterns can be selected. We assess our detection method using a baseline of 16 query patterns shown in Figure 5.10, with a description and key provided in Table 5-G and a detailed cross-reference in Table 5-F. The query patterns have been selected as a representative set of distinctive patterns that appear in a varying number of videos, which is designed to test pattern detection and retrieval accuracies in recall and precision. The videos are challenging for detection tasks for a number of reasons as we've discussed, including the variation of scale and occlusions that occur in nature scenes. This is effectively demonstrated in Table 5-G where we show, for each pattern, the dimensions of the defined ground truth and the minimum and maximum size (in pixel area) at which the pattern is observed along with the range of the observed pattern scales.

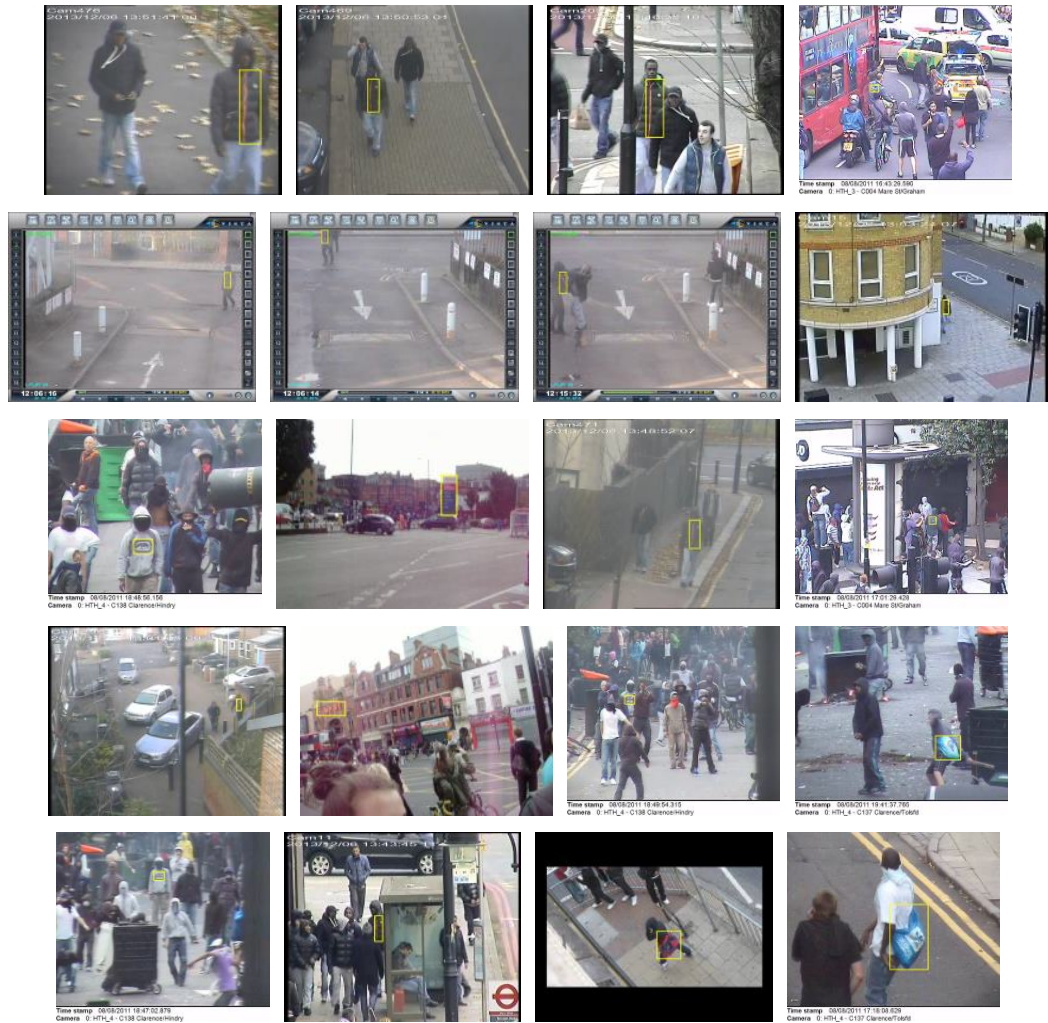


Figure 5.9: Representative thumbnails of videos in the dataset

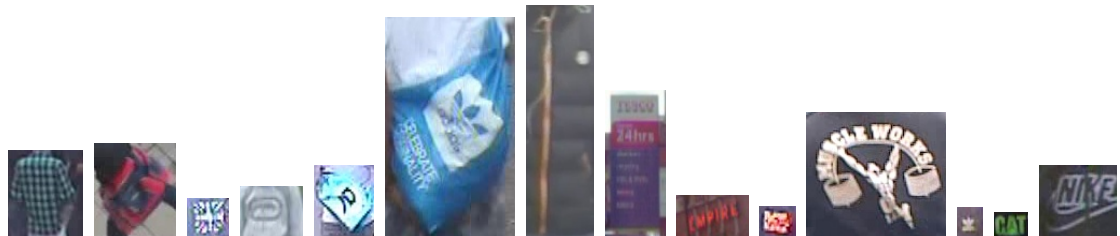


Figure 5.10: Query patterns A-N (see Table 5-G for descriptions) for which ground truth annotations are provided. The images are shown in proportional size to each other

Table 5-G: Key to the query patterns (Figure 5.10) in the dataset. See Table 5-F for detailed video attributes.

Key	Pattern	Query Image Size	Smallest (min area)	Largest (max area)	Scale range
A	Green check shirt	70 × 80	11 × 22	87 × 170	0.04 – 2.64
B	Red and blue bag	76 × 88	18 × 31	94 × 139	0.08 – 1.95
C	Karrimor logo	39 × 36	5 × 15	85 × 63	0.05 – 3.81
D	Grey Rhino hoodie	58 × 48	24 × 13	70 × 59	0.11 – 1.48
E	Blue JD sports bag	56 × 67	10 × 37	96 × 80	0.11 – 2.05
F	Blue Adidas sports bag	121 × 206	19 × 20	210 × 349	0.02 – 2.94
G	Black hoodie with orange zip	63 × 217	8 × 22	83 × 272	0.01 – 1.65
H	Tesco sign	56 × 138	11 × 109	57 × 147	0.16 – 1.08
I	Empire theatre sign	67 × 39	24 × 23	70 × 43	0.21 – 1.15
J	Orange SuperDry (very small)	33 × 29	10 × 10	35 × 31	0.10 – 1.13
K	Muscle gym logo	130 × 117	19 × 10	126 × 136	0.01 – 1.13
L	White Adidas logo (very small)	24 × 28	8 × 8	33 × 46	0.10 – 2.28
M	CAT logo	31 × 23	8 × 14	49 × 33	0.16 – 2.27
N	Nike logo (also seen in reverse)	72 × 67	16 × 17	109 × 78	0.06 – 1.76

We draw attention to some specific complexities contained within the dataset;

- the two blue sports bags, patterns E and F, appear identical apart from the logo on the bag; one is a white JD Sports logo and the other is a white Adidas logo. There is a high chance of confusion in detecting one bag over the other
- the dataset contains multiple copies of the same logo in different colour and/or with different coloured backgrounds, which can confuse algorithms in identifying the “correct” instance
- some videos are (partial) subsets of another; these are included to test the effectiveness of indexing short and long videos
- some query patterns are extremely small (e.g. pattern K is 67 × 39 pixels) which reduces the effectiveness of many detection and retrieval algorithms
- the Nike logo (pattern N) appears in reverse in some sections of video, which can be invisible to many detection algorithms

5.2.5 Results

Precision/Recall curves for the evaluations are shown in Figure 5.11 on the following pages. Pattern detection is the primary focus of the thesis, and we measure the accuracy of our method with respect to pattern detection (**red** curve) using precision and recall defined in Equation (5.21) on page 104. For completeness, we also discuss retrieval in two scales; *video retrieval* (**blue** curve) measures the retrieval accuracy ranking each of the

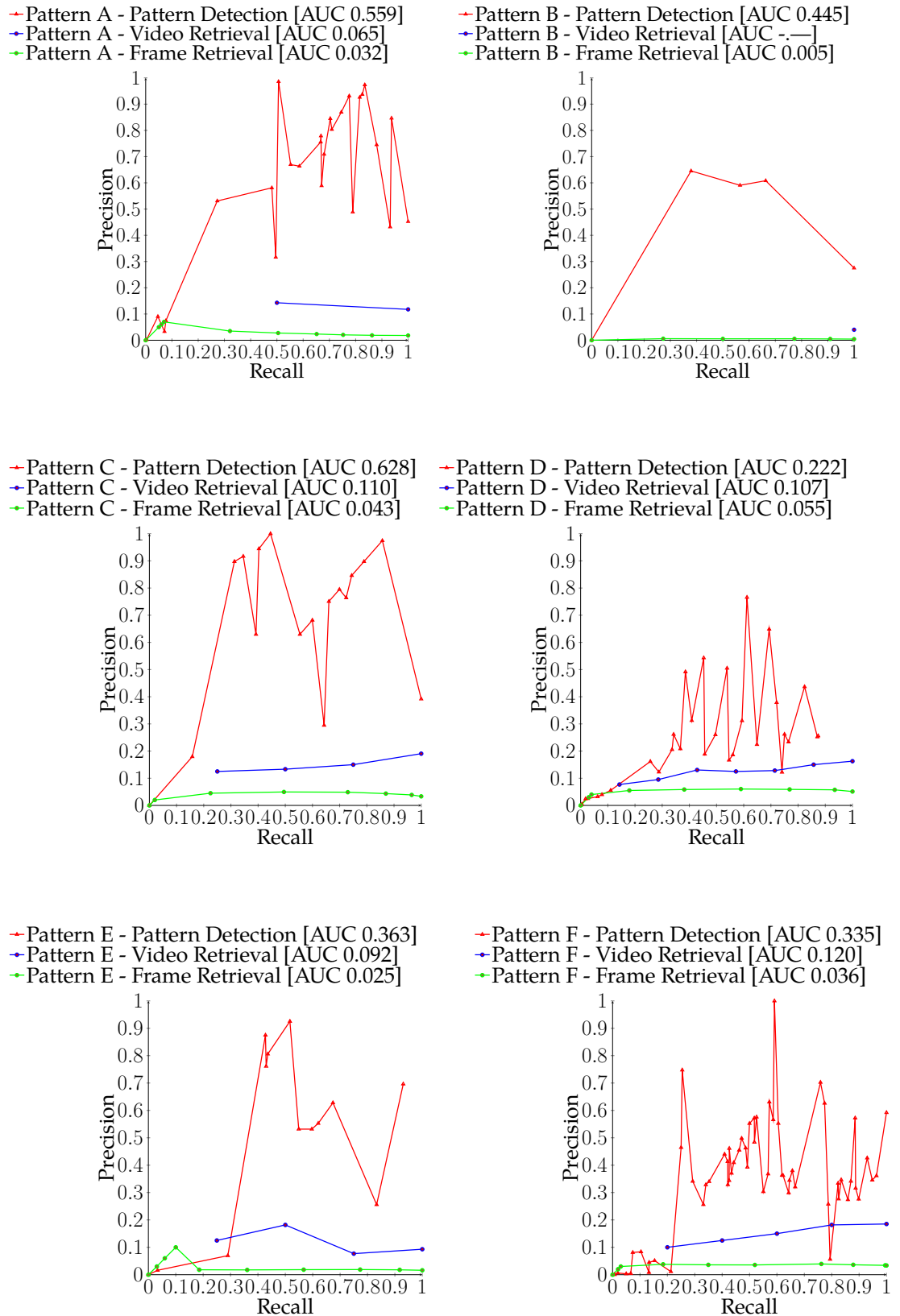


Figure 5.11a: Precision/Recall curves for search patterns A–F in our video corpus

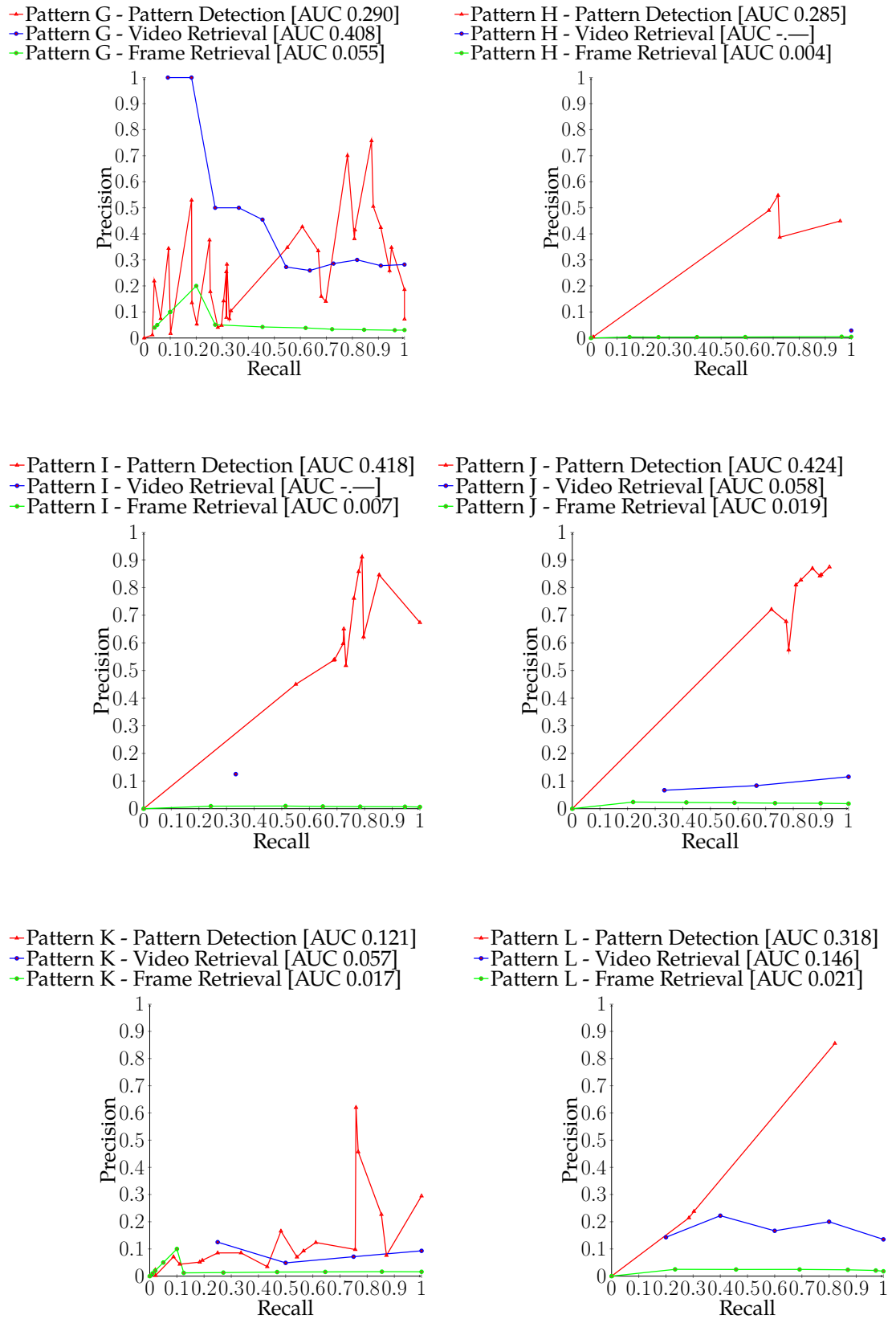


Figure 5.11b: Precision/Recall curves for search patterns G–L in our video corpus

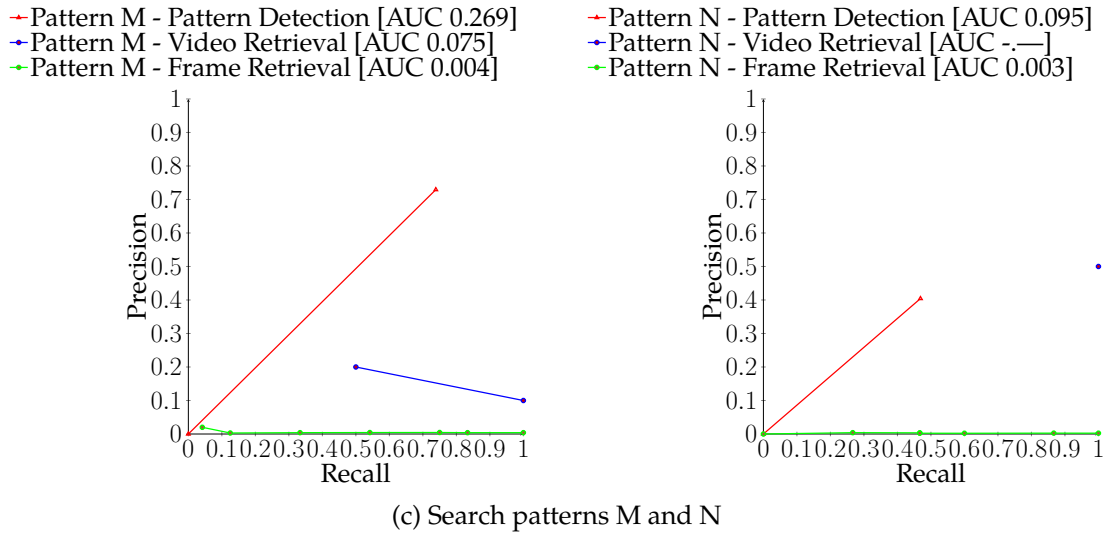


Figure 5.11: Precision/Recall curves for search patterns in our video corpus

forty three videos based on the likelihood of them containing the pattern in any frame, and more challenging, *frame retrieval* (green curve) measures the retrieval accuracy of individual frames based on their rank position in the 6,117 frame images. Each chart represents results for a single pattern (Table 5-G, Figure 5.10), and the *area under the curve* (AUC) measurement is calculated and shown in the legend. For patterns B, H, I and N, the video retrieval curve has only single data point, and therefore no AUC.

Pattern C has the largest AUC at 0.628 with 200 frame images in the index that contain the pattern, and ranks fourth highest in video retrieval and frame retrieval. The best retrieval result is seen for pattern G, with a video retrieval AUC of 0.408, but a frame retrieval AUC of only 0.055 which is matched by pattern D. This can be explained because pattern G appears in 11 of the 43 videos. Pattern N has only a single data point for video retrieval, however its position is highest on all the precision/recall charts with precision of 0.5 at recall 1.

5.2.6 Visual inspection of the pattern detection

While precision/recall curves show a full assessment of the method's accuracy, it is useful to inspect some examples closer to understand the effectiveness of the feature detection. In Figure 5.12 we illustrate a typical example of matching SIFT features from a query image to frames from our database. In Figure 5.12a, 16 features are accurately matched to their corresponding position in the frame from which the query image is extracted. In a different scene containing a clear shot of the shirt (query pattern) there is only one correspondence found (Figure 5.12b). In a third frame (Figure 5.12c), with a lot of visual



Figure 5.12: Examples of matching SIFT features in a query region to low quality images; (a) 16 features are correctly matched to the original query frame; (b) a single feature match from the query image to a subsequent frame of the video; (c) a single false match in a low contrast frame, and (d) a single false match in the presence of significant occlusion



Figure 5.13: Visual comparison of pattern detection results from the same query as Figure 5.12, using the Hough-forest-based pattern index method described in this chapter. Green rectangles are ground truth, yellow rectangles show the detected area and white rectangles show lower-ranked detection positions, eliminated in preference for the yellow.

interference, a correspondence is found to an incorrect position, and no features in the correct position are identified. Finally, Figure 5.12d shows a frame with $\sim 75\%$ occlusion of the pattern, and that also yields a single correspondence to an incorrect position. The same frames of videos are shown in Figure 5.13 with pattern detections shown using our method. The green rectangles show the ground truth bounding box of the pattern and the yellow rectangles show the detected area. The less dominant white rectangles show lower-ranked detection positions which were eliminated in preference for the yellow. In all example frames, the detected region heavily overlaps the ground truth and is classified as a positive detection. Figure 5.13c and Figure 5.13d are particularly noteworthy not only because both frames generated false positive correspondence using SIFT feature matching, but because of the *cause* of failure in matching SIFT descriptors. These frames demonstrate the ability of our method to detect patterns in very low contrast images and in the presence of very heavy occlusions.

5.2.7 Complexity

The runtime complexity of our systems is measure on a desktop PC with 32Gb RAM and an Intel[®] Core[™] i7-4790 processor running at 3.6GHz with 8 logical processors on 4 cores. Forests were trained with 30 trees, each with a maximum depth of 30, thus a potential tree size of 2^{30} ($\approx 10^9$) nodes. Our implementation¹ is multi-threaded to make maximum utilisation of the available CPU resource; in training each tree is trained using a CPU core and in detection, 8 sets of trees vote into independent accumulators and the accumulators are summed when all voting is complete (this follows the well-known *map-reduce* idiom of functional programming and popularised in [239]).

Training complexity The runtime complexity of building the index is not critical as it is not interacting with a user, but is important for scaling a solution. Building a search index is a one-off task to summarise the corpus into an easily searchable form, and is performed offline. The complexity of training using the videos in our test dataset ranged from 12 to 25 minutes per tree. Forests of 30 trees therefore required between 360 and 750 minutes of training time, which can be run in parallel depending of the machine capacity. In our environment, we have 8 CPU cores, and therefore train 8 trees simultaneously, requiring an elapsed time of 45 to 94 minutes per forest. The search index is built independently from user access to the system and can be made available – *brought online* – when the process completes. New videos can be added to the corpus at any time by training a new forest \mathcal{F}' independently of the working system, and adding it to F at any time without any impact to existing trained forests. Similarly, videos can be deleted immediately from

¹https://github.com/cdmh/hough_forests/tree/track_contributors

Table 5-H: Detection complexity for the patterns in Figure 5.10, Table 5-G (page 106)

Pattern	Query pattern size (pixels)	Number of patches	Average Detection time
A	70 × 80	3 575	5.143 secs
B	76 × 88	4 453	6.629 secs
C	39 × 36	504	2.085 secs
D	58 × 48	1 419	3.087 secs
E	56 × 67	2 132	3.661 secs
F	121 × 206	20 246	20.819 secs
G	63 × 217	9 696	14.495 secs
H	56 × 138	5 043	6.601 secs
I	67 × 39	1 248	2.729 secs
J	33 × 29	252	1.778 secs
K	130 × 117	11 730	13.334 secs
L	24 × 28	117	1.761 secs
M	31 × 23	128	1.680 secs
N	72 × 67	2 964	4.642 secs

the corpus simply by removing its forest.

Detection complexity is dependent on the size of forest and the size of the query pattern. Average times per forest are shown in Table 5-H. The smallest pattern, L, is 24×28 pixels and is detected in 1.761 seconds. The largest pattern, F, is 121×206 pixels and takes 20.819 seconds. The difference is explained because the sliding window passed over the query pattern creates a far greater number of patches which are filtered through the trees of the forest to cast votes. The query patterns contain 672 and 24,926 pixels, respectively, yielding $(24 - 15)(28 - 15) = 117$ and $(121 - 15)(206 - 15) = 20\,246$ patches. The bottleneck in our current implementation is the GLCM entropy calculation, which takes 94% of the CPU time for detection.

5.2.8 Scalability

To reduce data volumes, each forest is trained using data extracted from keyframes sampled from the videos (§5.2.1). Analysis of the most appropriate method of keyframe detection for pattern detection is an open research question beyond the scope of this thesis (§6.1.2), however the number of frames in the index directly affect the training time, and to a lesser extent the detection time, and is therefore a critical factor in scalability.

The method can be easily parallelised and the complexity can be easily handled with state-of-the-art computational tools and hardware. Each forest indexes a separate video and is independent of all other forests in off-line training and on-line user interaction.

As such, the limit of scalability is in the availability of hardware and the associated costs, rather than any physical limit.

5.2.9 Multi-scale detection

As people and objects move around a scene, or as a camera operator zooms in or out, the pattern for which we are searching can change size significantly, as was evident in Table 5-G. Detecting a pattern from a given example without consideration for multiple scales will perform much worse for a pattern of a dissimilar size to the query image. Consequently, multi-scale detection is performed to increase the likelihood of successful detection. A popular approach uniformly rescales the query image to different sizes, and multiple detections are performed, one at each scale. In each keyframe, the scale producing the strongest response (normalised with the scaled query frame $height \times width$) can be selected as the pattern hypothesis.

This multi-scale detection technique is commonplace but is not without its fragility. In rescaling the query image, blurring and other artefacts can be introduced, which affects pixel values. In up-scaling, missing information is filled in through various means of interpolation, and in downscaling, pixel values can be altered to improve visual appearance at the expense of distortion, which can affect our algorithm. The pixel pairwise-comparison from [92] is sensitive to pixel value interference such as that introduced by rescaling for multi-scale detection, and although the nature of random forests is that they are insensitive to noise through the accumulation of many thousands of votes from uncorrelated trees, this co-occurrence sensitivity is undesirable. Our use of the GLCM entropy of a patch in the split node helps with multi-scale detection because the entropy is minimally affected by relative effect of blurring during rescaling.

5.2.10 Multi-pattern search

Each forest is trained with keyframes of a video without regard of a query pattern. Consequently, a single forest forms a generic index that can be used to search for any number of patterns. This is demonstrated in Figure 5.14 where one forest is used to locate an *adidas* logo and a *marble hill* logo on different clothing. No additional training was performed, and a single forest can accurately detect the two distinct patterns shown (yellow boxes).

5.3 Conclusion

This chapter investigated the feasibility of training random decision forests for pattern detection using very few samples, and described a new technique for applying an esta-



Figure 5.14: Result of searching for multiple patterns in one video using a single forest. Left, the adidas logo is correctly found, and Right, the marble hill logo is detected, shown with a yellow rectangle.

blished framework of randomised Hough Forests to large-scale pattern detection. By re-thinking how a forest is trained and used in pattern detection, we have demonstrated the scalability and performance of pattern detection on a large scale.

Training the forest effectively is key to the performance of the method, and inevitably remains somewhat of an open research area with respect to the selection of keyframe images. This, and other future work, is discussed in more detail in §6.1. The use of negative patches from low-contrast areas of the image to train the forest works well in our experiments to date and has a good basis in theory. Experience from textual retrieval systems suggest this may not always hold, and perhaps is another area for future investigation.

Studies of [textual] retrieval effectiveness show that all terms should be indexed ... any visible component of a page might reasonably be used as a query term ... Even stopwords – which are of questionable value for bag-of-words queries – have an important role in phrase queries [135].

Relating this experience to the image search domain may suggest that low-contrast, and even background patches, should be included in the searchable forest and therefore used as positive patches for training.

Chapter 6

Conclusion

Pattern Detection has been a well-studied area in computer vision for many years, and yet current systems fall short in their capability to succeed in robust and consistent detection of patterns. This is particularly the case in low quality images, such as those studied in this thesis. We have observed the causes of the low quality that result from video recorded in natural environments (§1.1), often with low-grade and poorly maintained equipment, and we have described the practical issues of acquisition and re-capturing (§1.1.4) from the experience of the Metropolitan Police's VIIDO unit dealing with CCTV videos.

It is common knowledge that subjects in CCTV images who know they are being recorded try to disguise their identity and appearance. We described a particular example in Figure 3.1 which identified a need for algorithms to be effective in detecting patterns not only in their given orientation, but also in the mirror reflected orientation. To that end, Chapter 3 looked at the robustness of contemporary methods regarding *reflection invariance*. For the first time in the literature we analysed popular low-level feature detectors for consistency in images in either mirrored orientation. Assessment of ten popular image feature detectors showed that variance exists to some extent in most detectors (Table 3-F on page 57), although some are implementation flaws – for example, numerical rounding precision – and some are fundamental in the design of the detector. Our findings conclude that FAST and CenSurE detectors are the only two that are perfectly invariant. GFTT and the Harris Corner detector are invariant after feature matching and filtering algorithms are applied to find the correct correspondences in uneven sized sets of detected interest points. BRISK, ORB, SIFT and SURF cannot be considered invariant to bilateral symmetry, and SIFT is the least invariant of all the detectors that we have experimented with. Region-based detectors MSER and MSCR were assessed based on a common approach of defining a keypoint at the centre of the detected region. In

these cases, MSER is somewhat invariant, and MSCR is largely invariant, indicating that colour plays an important role in the reflection invariance of the maximally stable region algorithm. In modern research projects of scene classification [193, 197], object detection [193] and age-guessing [198], we demonstrated that when systems are presented with images and their horizontal reflections, they have produced results that are different for each reflected image orientation. In depth analysis of these systems and the cause of their variance is, however, outside the scope of this thesis.

Low level features are a fundamental working unit for most computer vision tasks, and matching features from two or more images is commonplace for many tasks including detection and retrieval. In Chapter 4 our research investigated improvements in corresponding features in low-quality images. Feature detectors work using the image texture, and are therefore sensitive variances caused by blurring and illumination changes. Under different image conditions, the features are less well defined and the feature detector can highlight different interest points. When this occurs, correspondence between feature descriptors extracted from the interest points is very difficult. Our novel blur-sensitive feature detection technique employed an adaptive approach to discovering matching features in pairs of images and can be simply extended to longer sequences of frame images by considering each consecutive pair in the sequence in turn and matching the blur parameters in each pair. Further, we used the texture and colour attributes in a composite descriptor with a unique distance calculation for feature matching that combined the attributes to improve matching effectiveness.

Our pattern detection scheme in Chapter 5 is robust to reflection invariance (§5.2.2), motivated by our findings in Chapter 3. We described a novel method of pattern indexing for large-scale video search using random forests in place of conventional *tf-idf* reverse indexes. By inverting the use of the forest to encode the patterns contained within video keyframes rather than the more usual model trained to encode representations of an a priori pattern or object, we demonstrated using a real-world work-flow of query-by-example, searching for distinctive patterns in videos from the wild. The automatic labelling of image training data from the same video is convenient for discovering negative test data and filtering low saliency regions of the image from the index. The node split function described results in a search query that is invariant to mirror reflection and rotation through 90° angles. The achieved performance shows results that are useful in a practical sense, and capable of detecting even small or occluded patterns in poor quality videos.

6.1 Future work

In respect of the pattern indexing presented in Chapter 5, there are a number of areas that interesting research exists to improve the performance in detection and retrieval. We take an opportunity to outline here a few ideas from where we think the greatest gains may come.

6.1.1 Retrieval & ranking

As we stated in §5.2.3 (page 102), “[c]orpus-wide relative ranking is somewhat outside the scope of this thesis.” If our method is to be usable in a content-based retrieval system, then retrieval and ranking need to be addressed. Other improvements discussed in this chapter should also improve retrieval performance, and further research is required to elevate it to an acceptable level for end-users.

Query Expansion Identifying objects in videos using object-based features that use dominant colour, texture, size, etc. is difficult and time-consuming [155]. Textual information retrieval systems use *Query Expansion* [234] to improve recall rates by expanding a search term which may consist of only one or two words to find related and relevant documents on the search subject. Chum et al. successfully used a query expansion technique [137, 244] to improve the recall rate from that achieved with a descriptor quantisation method, which can restrict the search horizon too much. Others have experimented with query expansion in multimedia retrieval [245, 246] and search re-ranking to improve the precision of search results [247]. It seems reasonable that a query expansion method could be applied in our system to improve and re-rank results for greater accuracy.

6.1.2 Keyframe selection

How to select the most effective keyframes for our method is an open research question, and more generally an active research area in the literature, e.g. [240, 241]. An assessment of the *best* method is local to the problem domain, and commonly addressed in related areas such as video summarisation [157, 242, 243]. In §5.2.1, consistent with other researchers [225–227], we sampled keyframes at a fixed interval through each video, but dynamically calculated the frame interval based on a formulation designed to capture a reasonable quantity of data while preventing an overload of keyframes on long videos. While this method produces an acceptable level of frames that contained patterns that we subsequently searched for, we recognise that there is room for further development to improve the technique. We need to balance the quantity of resulting keyframes with a

representative sample of the video. Our goal is to build an index for subsequent search for patterns of distinct visual appearance. Rather than sampling at fixed intervals, we pose a research question for further exploration:

Can the precision of our method be improved by using the local or global entropy properties to select keyframes from each video?

If keyframes are better selected to increase the likelihood of the pattern occurring, then the accuracy will be improved. More occurrences of the pattern in the tree will yield a stronger accumulated vote and therefore more accurate detection precision. More frames in the index containing the pattern – and perhaps more importantly, fewer frames not containing the pattern – will improve the retrieval recall, as we saw with pattern G in Figure 5.11 (blue curve) on page 109. Both these improvements are resulting in reduced noise and increasing distinct pattern inclusion in the index. This was the motivation behind our pattern labelling technique in §5.2.1 (page 94), and it seems a reasonable choice to use a similar measure in the selection of keyframes.

6.1.3 Feature Channels

We have seen throughout this thesis that the choice of image features greatly affects the performance of computer vision algorithms. This is no less true for the feature channels (§2.1.2) used in the pattern detection forests of Chapter 5 than those assessed for reflection invariance in §3.2 of Chapter 3.

Investigation of the choice of feature channels fell outside of the scope of our study, and consequently we chose to use the feature channels described by the original authors of the Hough Forests, Gall et al. [166]. However, some cursory experiments have showed that some feature channels contribute more votes into the ground truth area than others (see Figure 6.1; the channels are numbered 0 – 31 and ordered as described in §2.4.2).

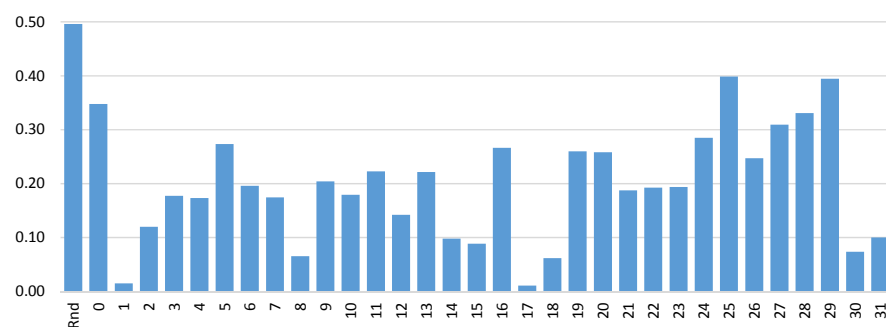


Figure 6.1: Distribution of feature channel contributions to correct votes in a sample of pattern searches

A small collection of pattern searches was run using a random selection of feature channel (per our method in Chapter 5), shown in §2.4.2 as *Rnd*. The same collection of searches was then repeated, limiting votes from each of the 32 feature channels in turn. The average precision of each collection is plotted in the figure. It is clear that some channels are more contributory than others in casting accurate votes for the ground truth area, which suggests that careful selection of the set of feature channels could improve the overall accuracy. Note, however, that the random selection out-performs all single-channel voting. It seems reasonable random selection of channels from a set of channels that each perform to greater accuracy than the current set would combine to an overall superior accuracy.

6.1.4 Online growth of trained random trees

Random forest methods, including ours, are typically trained to a fixed dimension of number of trees and the maximum depth of the forest. The choice of each of these parameters, along with others, affect the performance, complexity and memory requirements of the system, as we investigated in §5.1. Online random forests have been explored [248, 249], growing the trees further while the system is in use, rather than limiting training to a pre-user offline activity. On-the-fly learning offers a way to overcome the ‘closed world’ problem in computer vision, where object category recognition systems are restricted to only these pre-defined categories [250]. We have experimented with some online training in combination with our described method, but the the runtime complexity is currently too high to make a worthwhile evaluation.

We believe that a hybrid system can perform well; we suggest that each pattern search begins with the forests trained as described in Chapter 5, and with each patch extracted from the query image, the tree leaves are further split based on the GLCM entropy of the pattern’s patch. Votes are then cast from these new leaves, which provide a more discriminative vote for the specific pattern being search for.

To accommodate the online training, trees are trained offline to a shallower depth, keeping the leaves with larger numbers of patches. Leaf nodes are then subdivided using the online patches extracted from the search pattern. This reduction in tree depth reduces the complexity of the offline training, speeding up the training process. Conversely, the online detection process is significantly increased, and in the current implementation yields the system inappropriate for online searching. Some accuracy performance success has been observed in cursory experiments, however, and the hybrid offline/online training scheme could be very useful if the complexity is overcome.

Consummatum est

The research contained in this thesis will, I hope, live on and inspire future work in some of the areas described, and many more besides. As Charles Darwin wrote in 1869, towards the end of his life, in a letter to J.D. Hooker – botanists, explorer, founder of geographical botany, and Darwin’s closest friend – *“well it is a beginning, and that is something.”*¹

■

¹<http://harvardmagazine.com/2005/11/intelligent-evolution.html>

References

- [1] A. M. Treisman, G. Gelade, A feature-integration theory of attention, *Cognitive Psychology* 12 (1) (1980) pp. 97–136.
- [2] A. Anjulan, N. Canagarajah, A unified framework for object retrieval and mining, *IEEE Transactions on Circuits and Systems for Video Technology* 19 (1) (2009) pp. 63–76.
- [3] J. Sivic, A. Zisserman, Video Google: a text retrieval approach to object matching in videos, in: *Proceedings Ninth IEEE International Conference on Computer Vision*, Vol. 2, 2003, pp. 1470–1477.
- [4] J. Sivic, A. Zisserman, Efficient visual search of videos cast as text retrieval, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31 (4) (2009) pp. 591–606.
- [5] D. Bordwell, *The Way Hollywood Tells It: Story and Style in Modern Movies*, University of California Press, 2006.
- [6] J. Yuan, H. Wang, L. Xiao, W. Zheng, J. Li, F. Lin, B. Zhang, A Formal Study of Shot Boundary Detection, *IEEE Transactions on Circuits and Systems for Video Technology* 17 (2) (2007) pp. 168–186.
- [7] M. M. N. Asghar, F. Hussain, R. Manton, Video Indexing: A Survey, *International Journal of Computer and Information Technology* 3 (1) (2014) pp. 148–169.
- [8] G. Edelman, J. Bijhold, Tracking people and cars using 3D modeling and CCTV., *Forensic science international* 202 (1-3) (2010) pp. 26–35.
- [9] C. Henderson, E. Izquierdo, Robust Feature Matching in the Wild, in: *Science and Information Conference*, IEEE, London, 2015.
- [10] C. Schmid, R. Mohr, Local grayvalue invariants for image retrieval, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 19 (5) (1997) pp. 530–535.
- [11] F. Schaffalitzky, A. Zisserman, Automated Scene Matching in Movies, in: *Conference on Image and Video Retrieval*, 2002, pp. 186–197.
- [12] C. Henderson, S. Blasi, F. Sobhani, E. Izquierdo, On the impurity of street-scene video footage, in: *6th International Conference on Imaging for Crime Prevention and Detection (ICDP-15)*, IET, London, 2015.
- [13] M. Doring, Requirements for the ideal CCTV video codec, Technical report, Geutebruck Technical Report (2006).
- [14] C. Henderson, E. Izquierdo, Multi-scale reflection invariance, in: *SAI Computing*, London, 2016, pp. 420–425.

- [15] C. Henderson, E. Izquierdo, Symmetric stability of low level feature detectors, *Pattern Recognition Letters* 78 (2016) pp. 36–40.
- [16] C. Henderson, E. Izquierdo, Robust Feature Matching in Long-Running Poor-Quality Videos, *IEEE Transactions on Circuits and Systems for Video Technology* 26 (6) (2016) pp. 1161–1174.
- [17] C. Henderson, E. Izquierdo, Feature Correspondence in Low Quality CCTV Videos, in: L. Chen, S. Kapoor, R. Bhatia (Eds.), *Emerging Trends and Advanced Technologies for Computational Intelligence*, Vol. 647, Springer International Publishing, 2016, pp. 261–281.
- [18] C. Henderson, E. Izquierdo, Minimal Hough Forest training for pattern detection in Query by Example video search in: *International Conference on Systems, Signals and Image Processing (IWSSIP)*, London. (2015).
- [19] C. Henderson, E. Izquierdo, Rethinking random Hough Forests for video database indexing and pattern search, *Computational Visual Media* 2 (2) (2016) pp. 143–152.
- [20] C. Henderson, E. Izquierdo, Scalable pattern retrieval from videos using a Random Forest index, in: *International Conference on Internet of Things, Data and Cloud Computing (ICC 2017)*, Cambridge, UK, 2017.
- [21] M. J. Swain, D. H. Ballard, Color indexing, *International Journal of Computer Vision* 7 (1) (1991) pp. 11–32.
- [22] K. Konstantinidis, a. Gasteratos, I. Andreadis, Image retrieval based on fuzzy color histogram processing, *Optics Communications* 248 (4-6) (2005) pp. 375–386.
- [23] Y. Rubner, C. Tomasi, L. J. Guibas, Earth mover’s distance as a metric for image retrieval, *International Journal of Computer Vision* 40 (2) (2000) pp. 99–121.
- [24] M. A. Stricker, M. Orengo, Similarity of color images, in: W. Niblack, R. C. Jain (Eds.), *SPIE 2420, Storage and Retrieval for Image and Video Databases III*, 1995, pp. 381.
- [25] N. Dalal, B. Triggs, Histograms of Oriented Gradients for Human Detection, in: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*, Vol. 1, IEEE, 2005, pp. 886–893.
- [26] C. Papageorgiou, T. Poggio, A Trainable System for Object Detection, *International Journal of Computer Vision* 38 (1) (2000) pp. 15–33.
- [27] C. Harris, M. Stephens, A Combined Corner and Edge Detector, in: *Proceedings of the Alvey Vision Conference 1988*, Alvey Vision Club, 1988, pp. 147–151.
- [28] H. P. Moravec, *Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover.*, Technical Report, Stanford University, Dept. of Computer Science, California (1980).
- [29] J. Shi, C. Tomasi, Good features to track, in: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition CVPR-94*, IEEE Comput. Soc. Press, 1994, pp. 593–600.
- [30] K. Mikolajczyk, C. Schmid, An affine invariant interest point detector, in: *European Conference on Computer Vision (ECCV’02)*, Vol. 2350, 2002, pp. 128–142.
- [31] D. G. Lowe, Distinctive image features from scale-invariant keypoints, *International Journal of Computer Vision* 60 (2) (2004) pp. 91–110.
- [32] H. Bay, A. Ess, T. Tuytelaars, L. Van Gool, Speeded-Up Robust Features (SURF), *Computer Vision and Image Understanding* 110 (3) (2008) pp. 346–359.
- [33] P. Viola, M. Jones, Rapid object detection using a boosted cascade of simple features, *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and*

- Pattern Recognition. CVPR 2001
- [34] H. Bay, T. Tuytelaars, L. Van Gool, SURF: Speeded Up Robust Features, in: European Conference on Computer Vision (ECCV'06), 2006, pp. 404–417.
 - [35] E. Rosten, T. Drummond, Machine Learning for High-Speed Corner Detection, in: European Conference on Computer Vision (ECCV'06), Vol. 3951, Springer, 2006, pp. 430–443.
 - [36] S. Leutenegger, M. Chli, R. Y. Siegwart, BRISK: Binary Robust invariant scalable keypoints, in: 2011 International Conference on Computer Vision, IEEE, 2011, pp. 2548–2555.
 - [37] E. Rublee, V. Rabaud, K. Konolige, G. R. Bradski, ORB: An efficient alternative to SIFT or SURF, in: Proceedings of the IEEE International Conference on Computer Vision, 2011, pp. 2564–2571.
 - [38] M. Calonder, V. Lepetit, C. Strecha, P. Fua, BRIEF: Binary robust independent elementary features, in: European Conference on Computer Vision (ECCV'10), Vol. 6314, 2010, pp. 778–792.
 - [39] M. Agrawal, K. Konolige, M. R. Blas, CenSurE: Center Surround Extremas for Realtime Feature Detection and Matching, in: Computer Vision – ECCV 2008, Vol. 5305, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 102–115.
 - [40] P.-E. Forssén, D. G. Lowe, Shape descriptors for maximally stable extremal regions, in: Proceedings of the IEEE International Conference on Computer Vision, 2007, pp. 1–8.
 - [41] D. Nistér, H. Stewénus, Linear time maximally stable extremal regions, in: European Conference on Computer Vision (ECCV'08), 2008, pp. 183–196.
 - [42] P. M. Roth, M. Donoser, H. Bischof, Tracking for learning an object representation from unlabeled data, Proceedings of the Computer Vision Winter Workshop (CVWW) (2006) pp. 46–51.
 - [43] P.-E. Forssén, Maximally stable colour regions for recognition and matching, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 2007, pp. 1–8.
 - [44] D. S. Cheng, M. Cristani, M. Stoppa, L. Bazzani, V. Murino, Custom Pictorial Structures for Re-identification, in: Proceedings of the British Machine Vision Conference (BMVC) 2011, British Machine Vision Association, 2011 pp. 68.1–68.11.
 - [45] A. Bhattacharyya, On a measure of divergence between two statistical populations defined by their probability distributions, Bulletin of the Calcutta Mathematical Society 35 (1943) pp. 99–109.
 - [46] T. Tuytelaars, K. Mikolajczyk, Local Invariant Feature Detectors: A Survey, Foundations and Trends in Computer Graphics and Vision 3 (3) (2008) pp. 177–280.
 - [47] H. Yang, I. Patras, Mirror, mirror on the wall, tell me, is the error small?, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2015, pp. 4685–4693.
 - [48] A. Baumberg, Reliable feature matching across widely separated views, in: Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2000 2000.
 - [49] J.-M. Morel, G. Yu, ASIFT: A New Framework for Fully Affine Invariant Image Comparison, SIAM Journal on Imaging Sciences 2 (2) (2009) pp. 438–469.
 - [50] Y. Ke, R. Sukthankar, PCA-SIFT: a more distinctive representation for local image descriptors, in: Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision

- and Pattern Recognition(CVPR), 2004. Vol. 2, IEEE, 2004, pp. 506–513.
- [51] K. Mikolajczyk, C. Schmid, A performance evaluation of local descriptors, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27 (10) (2005) pp.1615–1630.
- [52] S. Saleem, R. Sablatnig, A Modified SIFT Descriptor for Image Matching under Spectral Variations, in: *Image Analysis and Processing – ICIAP 2013*, pp. 652–661.
- [53] Jie Chen, Shiguang Shan, Chu He, Guoying Zhao, M. Pietikäinen, Xilin Chen, Wen Gao, WLD: A Robust Local Image Descriptor, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32 (9) (2010) pp. 1705–1720.
- [54] M. F. Vural, Y. Yardimci, A. Tamlizel, Registration of multispectral satellite images with Orientation-Restricted SIFT, in: *IEEE International Geoscience and Remote Sensing Symposium, IEEE, 2009*, pp. III-243–III-246.
- [55] Z. Yi, C. Zhiguo, X. Yang, Multi-spectral remote image registration based on SIFT, *Electronics Letters* 44 (2) (2008) pp. 107–108.
- [56] S. Saleem, R. Sablatnig, A Robust SIFT Descriptor for Multispectral Images, *IEEE Signal Processing Letters* 21 (4) (2014) pp. 400–403.
- [57] S. Saleem, A. Bais, R. Sablatnig, Towards feature points based image matching between satellite imagery and aerial photographs of agriculture land, *Computers and Electronics in Agriculture* 126 (2016) pp. 12–20.
- [58] M. Heikkilä, M. Pietikäinen, C. Schmid, Description of interest regions with local binary patterns, *Pattern Recognition* 42 (3) (2009) pp. 425–436.
- [59] S. Saleem, R. Sablatnig, Interest Region Description Using Local Binary Pattern of Gradients, in: *Image Analysis, 2013*, pp. 468–477.
- [60] R. Arandjelović, A. Zisserman, Three things everyone should know to improve object retrieval, in: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2012*, pp. 2911–2918.
- [61] K. E. A. van de Sande, T. Gevers, C. G. Snoek, Evaluating color descriptors for object and scene recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32 (9) (2010) pp. 1582–1596.
- [62] Hurvich, Leo M., and Dorothea Jameson. An opponent-process theory of color vision, *Psychological review* 64 (6 pt1) (1957), pp. 384–404.
- [63] J. Zhang, Y. Barhomi, T. Serre, A new biologically inspired color image descriptor, in: *European Conference on Computer Vision (ECCV'12), Vol. 7576, Springer, 2012*, pp. 312–324.
- [64] A. Bosch, A. Zisserman, X. Muñoz, Scene classification using a hybrid generative/discriminative approach, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30 (4) (2008) pp. 712–727.
- [65] A. E. Abdel-Hakim, A. A. Farag, CSIFT: A SIFT descriptor with color invariant characteristics, in: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), Vol. 2, IEEE Computer Society, New York, 2006*, pp. 1978–1983.
- [66] J. van de Weijer, C. Schmid, Coloring Local Feature Extraction, in: *European Conference on Computer Vision (ECCV'06), Vol. 3952, 2006*, pp. 334–348.
- [67] C. Wengert, M. Douze, H. Jégou, Bag-of-colors for improved image search, in: *Proceedings of the 19th ACM International Conference on Multimedia - MM'11, ACM Press, New York*,

- New York, USA, 2011, pp. 1437-1440.
- [68] H. Abdi, L. J. Williams, Principal component analysis, *Wiley Interdisciplinary Reviews: Computational Statistics* 2 (4) (2010) pp. 433–459.
- [69] A. Sharma, K. K. Paliwal, Fast principal component analysis using fixed-point algorithm, *Pattern Recognition Letters* 28 (10) (2007) pp. 1151–1155.
- [70] D.-c. Shi, L. Xu, L.-y. Han, Image retrieval using both color and texture features, *The Journal of China Universities of Posts and Telecommunications* 14 (2007) pp. 94–99.
- [71] D.-c. Shi, G.-q. Yan, Combining MSCR detector and PCA-SIFT descriptor for scene recognition, in: *2nd International Conference on Advanced Computer Control (ICACC)*, vol. 2, IEEE, 2010, pp. 136–141.
- [72] A. Sinha, S. Banerji, C. Liu, New color GPHOG descriptors for object and scene image classification, *Machine Vision and Applications* 25 (2) (2013) pp. 361–375.
- [73] A. Bosch, A. Zisserman, X. Muñoz, Representing shape with a spatial pyramid kernel, in: *Proceedings of the 6th ACM International Conference on Image and video retrieval – CIVR’07*, ACM Press, New York, New York, USA, 2007, pp. 401–408.
- [74] A. Bosch, A. Zisserman, X. Muñoz, Image Classification using Random Forests and Ferns, in: *2007 IEEE 11th International Conference on Computer Vision*, IEEE, 2007, pp. 1–8.
- [75] A. Bosch, A. Zisserman, X. Muñoz, Scene classification via pLSA, in: *European Conference on Computer Vision (ECCV’06)*, Vol. 3954, 2006, pp. 517–530.
- [76] A. Oliva, W. Hospital, L. Ave, Modeling the Shape of the Scene : A Holistic Representation of the Spatial Envelope, *International Journal of Computer Vision* 42 (3) (2001) pp. 145–175.
- [77] C. Liu, V. K. Mago, Cross Disciplinary Biometric Systems, in: *15th International Conference on Image Processing, Computer Vision, and Pattern Recognition (CVPR)*, Vol. 37 of Intelligent Systems Reference Library, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 205–225.
- [78] X. Song, D. Muselet, A. Trémeau, Affine transforms between image space and color space for invariant local descriptors, *Pattern Recognition* 46 (8) (2013) pp. 2376–2389.
- [79] X. Tian, L. Jiao, X. Liu, X. Zhang, Feature integration of EODH and Color-SIFT: Application to image retrieval based on codebook, *Signal Processing: Image Communication* 29 (4) (2014) pp. 530–545.
- [80] M. Brown, G. Hua, S. Winder, Discriminative learning of local image descriptors., *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33 (1) (2011) pp. 43–57.
- [81] K. Simonyan, A. Vedaldi, A. Zisserman, Learning Local Feature Descriptors Using Convex Optimisation, *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2014) pp. 1573–1585.
- [82] S. A. Fezza, M.-C. Larabi, K. M. Faraoun, Feature-based Color Correction of Multi-View Video for Coding and Rendering Enhancement, *IEEE Transactions on Circuits and Systems for Video Technology* (2014) pp. 1486 – 1498.
- [83] K. E. A. van de Sande, T. Gevers, C. G. Snoek, Color Descriptors for Object Category Recognition, in: *Conference on Colour in Graphics, Imaging, and Vision (CGIV)*, Society for Imaging Science and Technology, 2008, pp. 378–381.
- [84] T. H. Rassem, B. E. Khoo, Object class recognition using combination of color SIFT descriptors, in: *2011 IEEE International Conference on Imaging Systems and Techniques*, IEEE,

- 2011, pp. 290–295.
- [85] J. R. Smith, S.-F. Chang, Tools and techniques for color image retrieval, in: *Storage and Retrieval for Image and Video Databases (SPIE)*, vol. 2670, 1996, pp. 2–7.
- [86] A. Vedaldi, H. Ling, S. Soatto, Knowing a good feature when you see it: Ground truth and methodology to evaluate local features for recognition, *Computer Vision*, vol. 285, Springer Berlin Heidelberg, pp. 27–49.
- [87] I. Dimitrovski, D. Kocev, S. Loskovska, S. Džeroski, Improving bag-of-visual-words image retrieval with predictive clustering trees, *Information Sciences*, vol. 329 (Feb 2016) pp. 851–865.
- [88] U. Park, A. Jain, I. Kitahara, K. Kogure, N. Hagita, ViSE: Visual Search Engine Using Multiple Networked Cameras, in: *18th International Conference on Pattern Recognition (ICPR'06)*, vol. 3, IEEE, 2006, pp. 1204–1207.
- [89] D. Gray, H. Tao, Viewpoint Invariant Pedestrian Recognition with an Ensemble of Localized Features, in: *European Conference on Computer Vision (ECCV'08)*, Springer, 2008, pp. 262–275.
- [90] B. Prosser, W.-S. Zheng, S. Gong, T. Xiang, Person Re-Identification by Support Vector Ranking, in: *Proceedings of the British Machine Vision Conference 2010*, British Machine Vision Association, 2010, pp. 21.1–21.11.
- [91] C. Liu, S. Gong, C. C. Loy, X. Lin, Person Re-identification: What Features Are Important?, in: A. Fusiello, V. Murino, R. Cucchiara (Eds.), *European Conference on Computer Vision (ECCV'12) Workshops and Demonstrations*, vol. 7583 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 391–401.
- [92] J. Gall, V. Lempitsky, Class-specific hough forests for object detection, in: *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2009*, pp. 1022–1029.
- [93] J. Sivic, F. Schaffalitzky, A. Zisserman, Efficient object retrieval from videos, in: *Proceedings of the 12th European Signal Processing Conference EUSIPCO 04 Vienna Austria, 2004*, pp. 36–39.
- [94] J. Yang, Y.-G. Jiang, A. G. Hauptmann, C.-W. Ngo, Evaluating bag-of-visual-words representations in scene classification, *Proceedings of the international workshop on Workshop on multimedia information retrieval (MIR '07) (2007)* pp. 197–206.
- [95] R. Shekhar, C. Jawahar, Word Image Retrieval Using Bag of Visual Words, in: *2012 10th IAPR International Workshop on Document Analysis Systems, IEEE, 2012*, pp. 297–301.
- [96] S. O'Hara, B. A. Draper, Introduction to the Bag of Features Paradigm for Image Classification and Retrieval (jan 2011).
- [97] Y.-G. Jiang, C.-W. Ngo, J. Yang, Towards optimal bag-of-features for object categorization and semantic video retrieval, *Proceedings of the 6th ACM International Conference on Image and Video Retrieval - CIVR '07 (2007)* pp. 494–501.
- [98] F. Schaffalitzky, A. Zisserman, Multi-view matching for unordered image sets, or “How do I organize my holiday snaps?”, in: *European Conference on Computer Vision (ECCV'02)*, vol. 2350, 2002, pp. 414–431.
- [99] J. Matas, O. Chum, M. Urban, T. Pajdla, Robust wide-baseline stereo from maximally stable extremal regions, in: *Image and Vision Computing*, 22 (10), 2004, pp. 761–767.

- [100] K. Mikolajczyk, C. Schmid, Indexing based on scale invariant interest points, in: Proceedings 8th IEEE International Conference on Computer Vision. ICCV 2001, vol. 1, IEEE Computer Society, 2001, pp. 525–531.
- [101] A. Alzu'bi, A. Amira, N. Ramzan, Semantic content-based image retrieval: A comprehensive study, *Journal of Visual Communication and Image Representation* 32 (2015) pp. 20–54.
- [102] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, C. Schmid, Aggregating Local Image Descriptors into Compact Codes, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34 (9) (2012) pp. 1704–1716.
- [103] J. Philbin, O. Chum, M. Isard, J. Sivic, A. Zisserman, Object retrieval with large vocabularies and fast spatial matching, in: 2007 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2007.
- [104] J. Philbin, O. Chum, M. Isard, J. Sivic, A. Zisserman, Lost in quantization: Improving particular object retrieval in large scale image databases, in: 2008 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2008.
- [105] F. Perronnin, C. Dance, Fisher Kernels on Visual Vocabularies for Image Categorization, in: 2007 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2007.
- [106] F. Perronnin, J. Sánchez, T. Mensink, Improving the fisher kernel for large-scale image classification, in: European Conference on Computer Vision (ECCV'10), Springer, 2010, pp. 143–156.
- [107] F. Perronnin, Y. Liu, J. Sanchez, H. Poirier, Large-scale image retrieval with compressed Fisher vectors, in: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2010, pp. 3384–3391.
- [108] R. Tao, E. Gavves, C. G. Snoek, A. W. Smeulders, Locality in Generic Instance Search from One Example, in: 2014 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2014, pp. 2099–2106.
- [109] B. Ma, Y. Su, F. Jurie, Local Descriptors Encoded by Fisher Vectors for Person Re-identification, in: European Conference on Computer Vision (ECCV'12). Workshops and Demonstrations, Springer, 2012, pp. 413–422.
- [110] C. G. Snoek, K. E. A. van de Sande, D. Fontijne, A. Habibian, M. Jain, MediaMill at TRECVID 2013: Searching Concepts, Objects, Instances and Events in Video, in: NIST TRECVID Workshop, 2013.
- [111] N. Keen, Color moments, School Of Informatics, University Of Edinburgh (2015).
- [112] H. Liu, R. Setiono, Chi2: feature selection and discretization of numeric attributes, in: Proceedings of 7th IEEE International Conference on Tools with Artificial Intelligence, 1995, pp. 388 – 391.
- [113] H. Jégou, M. Douze, C. Schmid, P. Pérez, Aggregating local descriptors into a compact image representation, in: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE, 2010, pp. 3304–3311.
- [114] R. Arandjelović, A. Zisserman, All About VLAD, in: 2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2013, pp. 1578–1585.
- [115] T. Liu, Z. Yuan, J. Sun, J. Wang, N. Zheng, X. Tang, H.-Y. Shum, Learning to detect a salient object., *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33 (2) (2011)

- pp. 353–67.
- [116] T. Kadir, J. Brady, Saliency, scale and image description, *International Journal of Computer Vision* 45 (2) (2001) pp. 83–105.
- [117] I. Guyon, A. Elisseeff, An introduction to variable and feature selection, *The Journal of Machine Learning Research*, 3 pp. 1157–1182.
- [118] H. Jégou, M. Douze, C. Schmid, On the burstiness of visual elements, in: *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2009*, pp. 1169–1176.
- [119] M. Shi, Y. Avrithis, H. Jégou, Early burst detection for memory-efficient image retrieval, in: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2015, pp. 605–613.
- [120] J. Meng, J. Yuan, Y.-P. Tan, G. Wang, Fast object instance search in videos from one example, in: *2015 IEEE International Conference on Image Processing (ICIP)*, IEEE, 2015, pp. 4381–4385.
- [121] D. Nistér, H. Stewénius, Scalable Recognition with a Vocabulary Tree, in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '06) 2* (2006) pp. 2161–2168.
- [122] P. Turcot, D. G. Lowe, Better matching with fewer features: The selection of useful features in large database recognition problems, in: *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, IEEE, 2009, pp. 2109–2116.
- [123] L. Sun, G. Liu, Visual Object Tracking Based on Combination of Local Description and Global Representation, *IEEE Transactions on Circuits and Systems for Video Technology* 21 (4) (2011) pp. 408–420.
- [124] E. Rosten, R. Porter, T. Drummond, Faster and better: a machine learning approach to corner detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32 (1) (2010) pp. 105–19.
- [125] G. Nebehay, R. Pflugfelder, Consensus-based matching and tracking of keypoints for object tracking, in: *2014 IEEE Winter Conference on Applications of Computer Vision (WACV) 2014*, IEEE, 2014, pp. 862–869.
- [126] M. Chli, A. Davison, Active matching, in: *European Conference on Computer Vision (ECCV'08)*, vol. 5302, 2008, pp. 72–85.
- [127] M. Muja, D. G. Lowe, Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration, in: *Proc. VISAPP International Conference on Computer Vision Theory and Applications*, 2009, pp. 331–340.
- [128] P. Mahalanobis, On the generalized distance in statistics, *Proceedings of the National Institute of Sciences (Calcutta)* 2 (1936) pp. 49–55.
- [129] R. De Maesschalck, D. Jouan-Rimbaud, D. Massart, The Mahalanobis distance, *Chemometrics and Intelligent Laboratory Systems* 50 (1) (2000) pp. 1–18.
- [130] M. Buckland, F. Gey, The relationship between Recall and Precision, *Journal of the American Society for Information Science* 45 (1) (1994) pp. 12–19.
- [131] W. T. Chu, T. C. Lin, Logo recognition and localization in real-world images by using visual patterns, in: *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2012, pp. 973–976.

- [132] S. Romberg, L. G. Pueyo, R. Lienhart, R. van Zwol, Scalable logo recognition in real-world images, *ACM International Conference on Multimedia Retrieval - ICMR* (2011)
- [133] R. Baeza-Yates, B. Ribeiro-Neto, *Modern information retrieval*, ACM Press, New York, 1999.
- [134] J. Sivic, A. Zisserman, Efficient Visual Search for Objects in Videos, in: *Proceedings of the IEEE* 96 (4) (2008) pp. 548–566.
- [135] J. Zobel, A. Moffat, Inverted files for text search engines, *ACM Computing Surveys* 38 (2) 2006.
- [136] Z. Liu, H. Li, W. Zhou, R. Zhao, Q. Tian, Contextual hashing for large-scale image search, *IEEE Transactions on Image Processing* 23 (4) (2014) pp. 1606–1614.
- [137] O. Chum, A. Mikulik, M. Perd’och, J. Matas, Total recall II: Query expansion revisited, in: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Prague, IEEE, 2011, pp. 889–896.
- [138] J. M. J. Kanter, K. Veeramachaneni, Deep Feature Synthesis: Towards Automating Data Science Endeavors, in: *International Conference on Data Science and Advanced Analytics (DSAA)*, IEEE, 2015.
- [139] M. Korytkowski, R. Scherer, P. Staszewski, P. Woldan, Bag-of-features image indexing and classification in Microsoft SQL Server relational database, in: *2015 IEEE 2nd International Conference on Cybernetics (CYBCONF)*, IEEE, 2015, pp. 478–482.
- [140] G. Yu, J. Yuan, Z. Liu, Unsupervised random forest indexing for fast action search, in: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2011, pp. 865–872.
- [141] A. Hietanen, J. Lankinen, J.-K. Kämäräinen, A. G. Buch, N. Krüger, A Comparison of Feature Detectors and Descriptors for Object Class Matching, *Neurocomputing* 184 (5), pp. 3–12.
- [142] J. Meng, J. Yuan, Gang Wang, Jianbo Xu, Object instance search in videos, in: *2013 9th International Conference on Information, Communications & Signal Processing (ICICS)*, IEEE, 2013.
- [143] K. Fukunaga, L. Hostetler, The estimation of the gradient of a density function, with applications in pattern recognition, *IEEE Transactions on Information Theory* 21 (1) (1975) pp. 32–40.
- [144] Y. Cheng, Mean shift, mode seeking, and clustering, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 17 (8) (1995) pp. 790–799.
- [145] Z. Kalal, K. Mikolajczyk, J. Matas, Tracking-learning-detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34 (7) (2012) pp. 1409–1422.
- [146] Y. Jiang, J. Meng, J. Yuan, J. Luo, Randomized Spatial Context for Object Search, *IEEE Transactions on Image Processing* 24 (6) (2015) pp. 1748–1762.
- [147] C. Cotsaces, N. Nikolaidis, I. Pitas, Video shot detection and condensed representation. a review, *IEEE Signal Processing Magazine* 23 (2).
- [148] Z. Wang, F. Z. Qureshi, I Remember Seeing This Video: Image Driven Search in Video Collections, in: *2013 International Conference on Computer and Robot Vision*, IEEE, 2013, pp. 326–331.
- [149] L. Rabiner, B. Juang, An introduction to hidden Markov models, *IEEE ASSP Magazine* 3 (1) (1986) pp. 4–16.

- [150] J. Bilmes, A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models, *International Computer Science Institute* 4 (510) (1998).
- [151] Yun Zhai, M. Shah, Video scene segmentation using Markov chain Monte Carlo, *IEEE Transactions on Multimedia* 8 (4) (2006) pp. 686–697.
- [152] C. Geyer, Practical Markov Chain Monte Carlo, *Statistical Science* 7 (4) (1992) pp. 473–483.
- [153] C.-R. Huang, H.-P. Lee, C.-S. Chen, Shot Change Detection via Local Keypoint Matching, *IEEE Transactions on Multimedia* 10 (6) (2008) pp. 1097–1108.
- [154] S. Porter, M. Mirmehdi, B. Thomas, A shortest path representation for video summarisation, in: *Proceedings of 12th International Conference on Image Analysis and Processing*, 2003. IEEE Computer Society, 2003, pp. 460–465.
- [155] W. Hu, N. Xie, L. Li, X. Zeng, S. Maybank, A Survey on Visual Content-Based Video Indexing and Retrieval, *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 41 (6) (2011) pp. 797–819.
- [156] A. Girgensohn, J. Boreczky, Time-constrained keyframe selection technique, *Proceedings IEEE International Conference on Multimedia Computing and Systems* (1999) pp. 756–761.
- [157] X.-D. Yu, L. Wang, Q. Tian, P. Xue, Multilevel video representation with application to keyframe extraction, *Proceedings of 10th International Multimedia Modelling Conference*, 2004.
- [158] D. Gibson, N. Campbell, B. Thomas, Visual abstraction of wildlife footage using Gaussian mixture models and the minimum description length criterion, in: *Object recognition supported by user interaction for service robots*, Vol. 2, IEEE Computer Society, 2002, pp. 814–817.
- [159] D. Wang, Unsupervised video segmentation based on watersheds and temporal tracking, *IEEE Transactions on Circuits and Systems for Video Technology* 8 (5) (1998) pp. 539–546.
- [160] L. Breiman, Random forests, *Machine Learning*, 45 (1) (2001) pp. 5–32.
- [161] T. K. Ho, Random decision forests, in: *Proceedings of 3rd International Conference on Document Analysis and Recognition*, vol. 1, IEEE Computer Society Press, 1995, pp. 278–282.
- [162] T. K. Ho, The random subspace method for constructing decision forests, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20 (8) (1998) pp. 832–844.
- [163] Y. Amit, D. Geman, Shape Quantization and Recognition with Randomized Trees, *Neural Computation* 9 (7) (1997) pp. 1545–1588.
- [164] T. Hastie, R. Tibshirani, J. Friedman, *The Elements of Statistical Learning*, Springer-Verlag New York, 2009.
- [165] J. A. Aslam, R. A. Popa, R. L. Rivest, On Estimating the Size and Confidence of a Statistical Audit, in: *Proceedings of the USENIX Workshop on Accurate Electronic Voting Technology*, 2007, p. 8.
- [166] J. Gall, A. Yao, N. Razavi, L. Van Gool, V. Lempitsky, Hough forests for object detection, tracking, and action recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33 (11) (2011) pp. 2188–2202.
- [167] J. Sivic, F. Schaffalitzky, A. Zisserman, Object Level Grouping for Video Shots, *International Journal of Computer Vision* 67 (2) (2006) pp. 189–210.

- [168] A. Araujo, M. Makar, V. Chandrasekhar, D. Chen, S. Tsai, H. Chen, R. Angst, B. Girod, Efficient video search using image queries, in: 2014 IEEE International Conference on Image Processing (ICIP), IEEE, 2014, pp. 3082–3086.
- [169] M. Makar, S. S. Tsai, V. Chandrasekhar, D. Chen, B. Girod, Interframe Coding of Canonical Patches for Low Bit-Rate Mobile Augmented Reality, *International Journal of Semantic Computing* 7 (1) (2013) pp. 5–24.
- [170] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, T. Darrell, DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition, in: 31st International Conference on Machine Learning, Beijing, China, 2013. pp. 647–655
- [171] M. Oquab, L. Bottou, I. Laptev, J. Sivic, Learning and Transferring Mid-level Image Representations Using Convolutional Neural Networks, in: 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2014, pp. 1717–1724.
- [172] M. Zeiler, R. Fergus, Visualizing and understanding convolutional networks, in: European Conference on Computer Vision (ECCV'14), vol. 8689 2014, pp. 818–833.
- [173] G. Loy, J.-O. Eklundh, Detecting Symmetry and Symmetric Constellations of Features, in: European Conference on Computer Vision (ECCV'06), 2006, pp. 508–521.
- [174] E. Michaelsen, D. Muench, M. Arens, Recognition of Symmetry Structure by Use of Gestalt Algebra, in: 2013 IEEE Conference on Computer Vision and Pattern Recognition Workshops, IEEE, 2013, pp. 206–210.
- [175] V. Patraucean, R. G. von Gioi, M. Ovsjanikov, Detection of Mirror-Symmetric Image Patches, in: 2013 IEEE Conference on Computer Vision and Pattern Recognition Workshops, IEEE, 2013, pp. 211–216.
- [176] Z. Wang, Z. Tang, X. Zhang, Reflection Symmetry Detection Using Locally Affine Invariant Edge Correspondence, *IEEE Transactions on Image Processing* 24 (4) (2015) pp. 1297–1301.
- [177] X. Guo, X. Cao, J. Zhang, X. Li, MIFT: A mirror reflection invariant feature descriptor, in: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 5995 LNCS, 2010, pp. 536–545.
- [178] X. Guo, X. Cao, MIFT: A framework for feature descriptors to be mirror reflection invariant, *Image and Vision Computing* 30 (8) (2012) pp. 546–556.
- [179] R. Ma, J. Chen, Z. Su, MI-SIFT: Mirror and Inversion Invariant Generalization for SIFT Descriptor, in: *Proceedings of the ACM International Conference on Image and Video Retrieval - CIVR '10*, ACM Press, New York, New York, USA, 2010, pp. 228–235.
- [180] W. L. Zhao, C. W. Ngo, Flip-invariant SIFT for copy and object detection, *IEEE Transactions on Image Processing* 22 (3) (2013) pp. 980–991.
- [181] X. Guo, X. Cao, FIND: A neat flip invariant descriptor, in: *Proceedings of International Conference on Pattern Recognition*, 2010, pp. 515–518.
- [182] H. Zhang, X. Guo, X. Cao, Water reflection detection using a flip invariant shape detector, in: *Proceedings of International Conference on Pattern Recognition*, Vol. 20, IEEE, 2010, pp. 633–636.
- [183] L. Xie, Q. Tian, B. Zhang, Max-SIFT: Flipping invariant descriptors for Web logo search, in: 2014 IEEE International Conference on Image Processing (ICIP), IEEE, 2014, pp. 5716–5720.
- [184] A. Kanazaki, Y. Mukuta, T. Harada, Mirror reflection invariant HOG descriptors for object detection, in: *Proceedings of International Conference on Image Processing (ICIP)*, IEEE,

- 2014, pp. 1594–1598.
- [185] M. Agarwal, V. Venkatraghavan, C. Chakraborty, A. K. Ray, A mirror reflection and aspect ratio invariant approach to object recognition using Fourier descriptor, *Applied Soft Computing Journal* 11 (5) (2011) pp. 3910–3915.
- [186] L. Fei-Fei, R. Fergus, P. Perona, Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories, *Computer Vision and Image Understanding* 106 (1) (2007) pp. 59–70.
- [187] S. Lazebnik, C. Schmid, J. Ponce, A sparse texture representation using local affine regions, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27 (8) (2005) pp. 1265–1278.
- [188] P. L. Rosin, Measuring Corner Properties, *Computer Vision and Image Understanding* 73 (2) (1999) pp. 291–307.
- [189] G. R. Bradski, The OpenCV Library, *Dr. Dobb’s Journal of Software Tools*. Available online <http://opencv.org>
- [190] M.-M. Cheng, G.-X. Zhang, N. J. Mitra, X. Huang, S.-M. Hu, Global contrast based salient region detection, in: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2011, pp. 409–416.
- [191] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, R. Fergus, Intriguing properties of neural networks, *arXiv preprint arXiv:1312.6199* (2013).
- [192] A. Nguyen, J. Yosinski, J. Clune, Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images, in: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. pp. 427–436
- [193] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, A. Oliva, Learning Deep Features for Scene Recognition using Places Database, in: Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems* 27, Curran Associates, Inc., 2014, pp. 487–495.
- [194] J. Xiao, K. A. Ehinger, J. Hays, A. Torralba, A. Oliva, SUN Database: Exploring a Large Collection of Scene Categories, *International Journal of Computer Vision* 119 (1), pp. 3–22.
- [195] via Google Image Search, Palace of Westminster (2016).
- [196] Wikipedia, City of London skyline (2016).
- [197] Wolfram, The Wolfram Language Image Identification Project (2016).
- [198] Microsoft, How Old Do I Look? (2016).
- [199] Wikipedia, Photograph of Alan Turing (2016).
- [200] The Telegraph, Photograph of Prince Charles (2016).
- [201] R. Dash, B. Majhi, Motion blur parameters estimation for image restoration, *Optik - International Journal for Light and Electron Optics* 125 (5) (2014) pp. 1634–1640.
- [202] C. Schuler, M. Hirsch, Learning to Deblur, in: *NIPS 2014 Deep Learning and Representation Learning Workshop*, Montreal, 2014.
- [203] J. Canny, A Computational Approach to Edge Detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 8 (6) (1986) pp. 679–698.
- [204] B. D. Lucas, T. Kanade, An iterative image registration technique with an application to stereo vision, in: *Proceedings of the 7th International Joint Conference on Artificial Intelligence, IJCAI ’81*, pp. 674–679.

- [205] I. Goodfellow, Y. Bengio, A. Courville *Deep Learning* MIT Press, Cambridge, MA ISBN 9780262035613
- [206] S. Lazebnik, C. Schmid, J. Ponce, Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories, in: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, (CVPR'06), vol. 2, IEEE, 2006, pp. 2169–2178.
- [207] L. Breiman, Statistical Modeling: The Two Cultures, *Statistical Science* 16 (3) 2001 pp. 199–231
- [208] O. Barinova, V. Lempitsky, P. Kholi, On Detection of Multiple Object Instances Using Hough Transforms, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34 (9) (2012) pp. 1773–1784.
- [209] N. Yokoya, A. Iwasaki, Object Detection Based on Sparse Representation and Hough Voting for Optical Remote Sensing Imagery, *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 8 (5) (2015) pp. 2053–2062.
- [210] K. Rematas, B. Leibe, Efficient object detection and segmentation with a cascaded Hough Forest ISM, in: 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops), IEEE, 2011, pp. 966–973.
- [211] K. Briechle, U. Hanebeck, Template matching using fast normalized cross correlation, in: *Proceedings of SPIE* 4387 (2001) p. 95–102.
- [212] I. Matthews, T. Ishikawa, S. Baker, The template update problem, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26 (6) (2004) pp. 810–815.
- [213] D. Doermann, D. Mihalcik, Tools and techniques for video performance evaluation, *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000* pp. 167–170
- [214] A. T. Nghiem, F. Bremond, M. Thonnat, V. Valentin, ETISEO, performance evaluation for video surveillance systems 2007 IEEE Conference on Advanced Video and Signal Based Surveillance, IEEE 2007, pp. 476–481
- [215] M. Godec, P. M. Roth, H. Bischof, Hough-based tracking of non-rigid objects, *Computer Vision and Image Understanding* 117 (10) (2013) pp. 1245–1256.
- [216] A. Yao, J. Gall, L. Van Gool, A hough transform-based voting framework for action recognition, in: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010, pp. 2061–2068.
- [217] M. Sun, G. Bradski, B.-X. Xu, S. Savarese, Depth-Encoded Hough Voting for Joint Object Detection and Shape Recovery, in: *European Conference on Computer Vision (ECCV'10)*, 2010, pp. 658–671.
- [218] G. Fanelli, A. Yao, P.-L. Noel, S. H. Gage, L. V. Gool, Hough forest-based facial expression recognition from video sequences, in: K. N. Kutulakos (Ed.), *European Conference on Computer Vision (ECCV'12) Workshops*, vol. 6553 of *Trends and Topics in Computer Vision*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 195–206.
- [219] A. Criminisi, D. Robertson, E. Konukoglu, J. Shotton, S. Pathak, S. White, K. Siddiqui, Regression forests for efficient anatomy detection and localization in computed tomography scans, *Medical Image Analysis* 17 (8) (2013) pp. 1293–1303.
- [220] J. Gall, N. Razavi, L. Van Gool, An Introduction to Random Forests for Multi-class Object Detection, in: *15th International Workshop on Theoretical Foundations of Computer Vision*, 2012, pp. 243–263.

- [221] K. Ma, J. Ben-Arie, Compound Exemplar Based Object Detection by Incremental Random Forest, in: 22nd International Conference on Pattern Recognition, IEEE (2014), pp. 2407–2412.
- [222] N. Razavi, S. H. Gage, L. van Gool, Backprojection Revisited: Scalable Multi-view Object Detection and Similarity Metrics for Detections, in: K. Daniilidis, P. Maragos, N. Paragios (Eds.), 11th European Conference on Computer Vision (ECCV'10), Vol. 6311 LNCS, Springer Berlin Heidelberg, Crete, Greece, 2010, pp. 620–633.
- [223] P. Viola, M. J. Jones, Robust Real-Time Face Detection, *International Journal of Computer Vision* 57 (2) (2004) pp. 137–154.
- [224] C. H. Lampert, M. B. Blaschko, T. Hofmann, Beyond sliding windows: Object localization by efficient subwindow search, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2008, pp. 1–8.
- [225] H. Grabner, P. M. Roth, H. Bischof, Is pedestrian detection really a hard task, in: IEEE International Workshop on Performance Evaluation of Tracking and Surveillance, 2007.
- [226] S. Oh, A. Hoogs, A. Perera, N. Cuntoor, C.-C. Chen, J. T. Lee, S. Mukherjee, J. K. Aggarwal, H. Lee, L. Davis, E. Swears, X. Wang, Q. Ji, K. Reddy, M. Shah, C. Vondrick, H. Pirsiavash, D. Ramanan, J. Yuen, A. Torralba, B. Song, A. Fong, A. Roy-Chowdhury, M. Desai, A large-scale benchmark dataset for event recognition in surveillance video, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2011, pp. 3153–3160.
- [227] P. Over, J. Fiscus, G. Sanders, D. Joy, TRECVID 2014 – An Overview of the Goals, Tasks, Data, Evaluation Mechanisms and Metrics, in: Proceedings of TRECVID 2014, Orlando, Florida, 2014.
- [228] W. Kazmi, H. J. Andersen, A comparison of interest point and region detectors on structured, range and texture images, *Journal of Visual Communication and Image Representation* 32 (2015) pp. 156–169.
- [229] R. M. Haralick, K. Shanmugam, I. Dinstein, Textural Features for Image Classification, *IEEE Transactions on Systems, Man, and Cybernetics* 3 (6) (1973) pp. 610–621.
- [230] A. Criminisi, J. Shotton (Eds.), *Decision Forests for Computer Vision and Medical Image Analysis*, Springer Science & Business Media, 2013.
- [231] V. Lepetit, P. Fua, Keypoint recognition using randomized trees, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28 (9) (2006) pp. 1465–1479.
- [232] P.C. Chen, T. Pavlidis, Segmentation by texture using a co-occurrence matrix and a split-and-merge algorithm, *Computer Graphics and Image Processing* 10(2) June 1979, pp. 172–182
- [233] M. Yazdi, K. Gheysari A new approach for the fingerprint classification based on gray-level co-occurrence matrix, *World Academy of Science, Engineering and Technology*, vol. 47, pp. 313–316
- [234] C. Carpineto, G. Romano, A Survey of Automatic Query Expansion in Information Retrieval, *ACM Computing Surveys* 44 (1) (2012) pp. 1–50.
- [235] D. Powers, Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation, *Journal of Machine Learning Technologies* 2 (1) (2011) pp. 37–63.
- [236] J. Davis, M. Goadrich, The relationship between Precision-Recall and ROC curves, in: Proceedings of the 23rd international conference on Machine learning (ICML '06), ACM

- Press, New York, New York, USA, 2006, pp. 233–240.
- [237] F. Provost, T. Fawcett, R. Kohavi, The case against accuracy estimation for comparing induction algorithms, in: *Proceedings of the 15th International Conference on Machine Learning*, 1998.
- [238] C. Manning, H. Schütze, *Foundations of statistical natural language processing*, 2nd Edition, MIT Press, 1999.
- [239] J. Dean, S. Ghemawat, MapReduce: Simplified Data Processing on Large Clusters, in: *Proceedings of 6th Symposium on Operating Systems Design and Implementation (OSDI'04)*, 2004, pp. 137–149.
- [240] I. Chugh, R. Gupta, R. Kumar, P. Sahay, Techniques for key frame extraction: Shot segmentation and feature trajectory computation, in: *Proceedings of 6th International Conference – Cloud System and Big Data Engineering (Confluence)*, IEEE, 2016, pp. 463–466.
- [241] R. Hannane, A. Elboushaki, K. Afdel, P. Naghabhushan, M. Javed, An efficient method for video shot boundary detection and keyframe extraction using SIFT-point distribution histogram, *International Journal of Multimedia Information Retrieval* 5 (2) (2016) pp. 89–104.
- [242] J. Peng, Q. Xiaolin, Keyframe-based Video Summary using Visual Attention Clues, *IEEE Multimedia* 17 (2) (2009) pp. 64–73.
- [243] M. Hudelist, K. Schoeffmann, Q. Xu, Improving interactive known-item search in video with the keyframe navigation tree, *MultiMedia Modeling* vol. 8935 (2015) pp. 306–317.
- [244] O. Chum, J. Philbin, J. Sivic, M. Isard, A. Zisserman, Total Recall: Automatic Query Expansion with a Generative Feature Model for Object Retrieval, *IEEE 11th International Conference on Computer Vision (ICCV)*, 2007.
- [245] A. P. Natsev, A. Haubold, J. Tešić, L. Xie, R. Yan, Semantic concept-based query expansion and re-ranking for multimedia retrieval, in: *Proceedings of the 15th international conference on Multimedia*, 2007, pp. 991–1000.
- [246] R. Arandjelović, *Advancing Large Scale Object Retrieval*, PhD thesis, University of Oxford (2013).
- [247] T. Mei, Y. Rui, S. Li, Q. Tian, Multimedia search reranking: A literature survey, *ACM Computing Surveys* 46 (3) (2014) pp. 1–38.
- [248] A. Saffari, C. Leistner, J. Santner, M. Godec, H. Bischof, On-line Random Forests, in: *Proceedings of 12th International Conference on Computer Vision Workshops, ICCV Workshops*, IEEE, 2009, pp. 1393–1400.
- [249] S. Schuler, C. Leistner, P. Roth, On-line Hough Forests, in: *British Machine Vision Conference (BMVC) 2011*, British Machine Vision Association, 2011.
- [250] K. Chatfield, R. Arandjelović, O. Parkhi, A. Zisserman, On-the-fly learning for visual search of large-scale image and video datasets, *International Journal of Multimedia Information Retrieval* 4 (2) (2015) pp. 75–93.