EFFICIENT DEPICTION OF VIDEO FOR SEMANTIC RETRIEVAL APPLICATIONS BY DIMENSIONALITY REDUCTION OF VISUAL FEATURE SPACE

Amarakoon Mudiyanselage Randitha Ravimal Bandara

(128029E)

Degree of Doctor of Philosophy

Department of Information Technology

University of Moratuwa

Sri Lanka

March 2021

EFFICIENT DEPICTION OF VIDEO FOR SEMANTIC RETRIEVAL APPLICATIONS BY DIMENSIONALITY REDUCTION OF VISUAL FEATURE SPACE

Amarakoon Mudiyanselage Randitha Ravimal Bandara

(128029E)

Thesis submitted in partial fulfilment of the requirements for the degree Doctor of Philosophy

Department of Information Technology

University of Moratuwa

Sri Lanka

March 2021

DECLARATION

I declare that this is my own work, and this thesis does not incorporate, without acknowledgment, any material previously submitted for a Degree or Diploma in any other University or institute of higher learning, and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgment is made in the text.

Similarly, I hereby grant to University of Moratuwa the non-exclusive right to reproduce and distribute my thesis, in whole or in part in print, electronic, or other media. I retain the right to use this content in whole or part in future works (such as articles or books).

Signature: UOM Verified Signature Date: 2/3/2021

The above candidate has carried out research for the PhD thesis under my supervision.

Name of the supervisor I: Dr. Lochandaka Ranathunga

Signature of the supervisor I: UOM Verified Signature Date: 2/3/2021

Name of the supervisor II: Associate Professor Dr. Nor Aniza Abdullah

Signature of the supervisor II: UOM Verified Signature Date: 2/3/2021

ABSTRACT

The retrieval of temporal digital visual data, either by a text or visual query, requires automatic interpretation, which includes high-level annotation by object detection and recognition for text query-based retrieval and low-level abstraction for visual query-based retrieval. Both the accuracy and the speed of the interpretation become crucial factors in real-world applications, due to the high density of visual data. This study has focused on reducing the complexity of visual data efficiently by dimensionality reduction techniques for the detection and recognition of objects in videos for both textual annotation and visual query-based video frame retrieval. The contribution of the study includes three approaches, i.e., a novel visual feature descriptor based on colour dithering – namely Salient Dither Pattern Feature (SDPF), novel object segmentation method based on the proposed feature descriptor – namely Refining Superpixel and Histogram of oriented optical flow Clustering (RSHC) –, and a novel self-supervised local descriptor – namely Network-in-Network with Restricted Boltzmann Machine (NIN-RBM). The experimental results make it evident that the SDPF is rotation and scale invariant and computationally efficient yet shows similar object recognition accuracy to the state-of-the-art methods with minimum supervision. The results further revealed that RSHC has successfully utilized SDPF for accurately segmenting individual objects by using a very shallow history of motion. Furthermore, according to the results, NIN-RBM has shown the state-of-the-art correspondence matching performance over the existing deep-learned self-supervised binary descriptors, keeping the computation time at the minimum. The overall results support the conclusions that RSHC is capable of accurately segment objects in a video, and then SDPF can be successfully used for recognizing the segmented objects. Moreover, NIN-RBM can be used to reliably and rapidly retrieve video frames related to any visual query. Since NIN-RBM is a local descriptor, it can be further used for locating of high-level objects and estimating their poses precisely, to improve the details of semantics retrieved from video data.

Keywords: dimensionality reduction, colour dithering, deep learning, video segmentation, object recognition, correspondence matching, binary descriptor

ACKNOWLEDGEMENT

First and foremost, I consider it is my bounden duty to record here my sincerest gratitude and appreciation to my supervisor, Dr. Lochandaka Ranathunga for his productive cooperation, guidance, and supervision extended throughout this research project. If not for his encouragement and support, this project would never have been a reality. Similarly, I unreservedly thank my co-supervisor, Associate Professor Dr. N.A. Abdullah from the University of Malaya, Malaysia for guiding me to succeed in this endeavour.

I express my heartfelt thanks to the Vice-Chancellor of the University of Moratuwa, the Dean of the Faculty of Information Technology and the Head of the Department of Information Technology, the non-academic staff members of the Faculty of Information Technology, University of Moratuwa for the opportunity given to commence my research work at the University of Moratuwa and facilitating me to carry out the same successfully. Further, I extend my sincere gratefulness to the Vice-Chancellor of the University of Sri Jayewardenepura, the Dean of the Faculty of Applied Sciences and the Head of the Department of Computer Science, and my colleagues in the academic staff for their assistance in many ways, which was a blessing for completing this research study. Thanks, are also due to the chairman and the staff of the National Research Council, Sri Lanka for granting a research scholarship under grant number NRC/12-017 to financially support this endeavour. Also, I must be thankful to the Senate Research Committee of University of Moratuwa for the financial support provided to carry out this study. Furthermore, I am indebted to the CEO of LK Domain Registry for granting me the Prof. V.K. Samaranayake Research Grant in order to continue my research studies at the University of Malaya, Malaysia.

I declare my salutation and admiration for all the esteemed authors, researchers, and philosophers for their great theories, research, publications, and ideas, which have been an enormous support to enhance this research work.

The support and motivation provided by my postgraduate friends, Mr. K.A.S.H. Kulathilake, Mr. V. Senthooran, Mr. B. Hettige and Mrs. M. Sirisuriya who are

affiliated with the Faculty of Information Technology, University of Moratuwa, too deserve mentioning with a debt of gratitude.

I am grateful to my parents, my sister, and brother, for all their encouragement and support extended right throughout this endeavour. If not for them, this project might not have been a reality. Last but not least, it is the dedication of my loving wife and daughter who were bearing all the burdens without passing them to me throughout the past few years that made me complete this research. Finally, I am grateful to all those who assisted me in numerous ways during the course of this research.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Acronym	Definition
ALOI	Amsterdam Library of Object Images
ANN	Artificial Neural Network
ARM	Advanced RISC (Reduced Instruction Set Computing) Machine
BOW	Bag of Words
BRIEF	Binary Robust Independent Elementary Features
BRISK	Binary Robust Invariant Scalable Keypoints
CCTV	Closed-Circuit Television
CDPC	Compact Dither Pattern Code
CIELAB	International Commission on Illumination. L, A and B are colour components
CIFAR	Canadian Institute for Advanced Research
CKN	Convolutional Kernel Network
CNN	Convolutional Neural Network
CNNH	Convolutional Neural Network with Hashing Layer
CPU	Central Processing Unit
CUDA	Compute Unified Device Architecture
DBD-MQ	Deep Binary Descriptor with Multi-Quantization
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
DDD	Dither Density Descriptor
DNN	Deep Neural Network
DSH	Deep Supervised Hashing

Acronym	Definition
EHD	Edge Histogram Descriptor
FAST	Features from Accelerated Segment Test
FC-GPHOG	Fused Colour-Gabor Pyramidal Histogram of Oriented Gradient
FREAK	Fast Retina Key- point
GAP	Global Average Pooling
GBRBM	Gaussian-Bernoulli Restricted Boltzmann Machine
GPHOG	Gabor Pyramidal Histogram of Oriented Gradient
GPU	Graphic Processing Unit
HOG	Histogram of Oriented Gradients
HOOF	Histogram of Oriented Optical Flow
HSDPF	Hessian based Salient Dither Pattern Feature
HSV	Hue Saturation Value
HTD	Homogeneous Texture Descriptor
IFV	Improved Fisher Vector
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
ITQ	Iterative Quantization
LAP	Local Average Pooling
LDA	Linear Discriminant Analysis
LDB	Local Difference Binary
LPP	Locality Preserving Projection
LSH	Local Sensitivity Hashing
MLP	Multi-Layer Perceptron

Acronym	Definition
NIN	Network in Network
NIN-RBM	Network in Network with Restricted Boltzmann Machine
ORB	Oriented FAST and Rotated BRIEF
PCA	Principle Component Analysis
PHOG	Pyramidal Histogram of Oriented Gradient
RAM	Random Access Memory
RBF	Radial Basis Function
RBM	Restricted Boltzmann Machine
ReLU	Rectified Linear Unit
RFD	Receptive Fields Descriptor
RGB	Red, Green, and Blue
ROC	Receiver Operating Characteristics
RPi	Raspberry Pi
RSHC	Refining Superpixels using HOOF and Colour
SDPF	Salient Dither Pattern Feature
SGD	Stochastic Gradient Descent
SIFT	Scale Invariant Feature Transform
SIMD	Single Instruction Multiple Data
SKAR	Smoothed Keypoint-Matching Ratio
SLIC	Simple Linear Iterative Clustering
SURF	Speeded Up Robust Feature
SVM	Support Vector Machine

Acronym	Definition
TBD	Texture Browsing Descriptor
VGG	Visual Geometry Group

CHAPTER 1

INTRODUCTION

Video is one of the multimedia elements frequently used in almost all multimedia applications. The spatial, spectral, and temporal dimensions of video make the data density generally higher than the text, image, and audio elements. The data density of videos continuously increases with the advancement of the technologies related to video, such as video capturing, encoding, decoding, storing, transferring, and streaming. The machine understanding of any multimedia element in digital devices has become an essential requirement for improving the capabilities of searching and indexing of the data. Unlike texts, a video file is challenging to interpret automatically by a digital device without human interaction, due to the complexity of its data. The challenges can be categorized into two classes, first, finding the optimal approach to educate intelligent software to interpret videos and the second, optimizing the software or hardware to interpret within a specified time. The latter is more challenging in instances when the hardware resources are limited, and the video is required to be processed on a real-time basis. Dimensionality reduction is a general technique to extract only the necessary features from the raw data to process them with less complexity. However, such techniques themselves are computationally expensive, and requires satisfying certain strict constraints such as the need for the dataset to be consisting of a meaningful subspace where the original data can be projected. This research study has been carried out for the purpose of finding a novel method that depicts videos efficiently by using a set of computationally cheaper dimensionality reduction techniques. This thesis includes a comprehensively elaborated introduction to the problem, research background, methodology, experimental methods, results, and discussion of the study; the objective of this chapter is to discuss the formation and applications of digital video, video processing, and the problems in the generic process for machine understanding of digital videos.

1.1 Digital video

A digital video is a digital representation of a sequence of interrelated images/frames, which depicts an action or event of visual objects. It can be primarily captured using a video camera that typically converts the illumination variations of an image created by a set of physical optical apparatus to a digital signal, which then arranges them systematically following a video encoding scheme. As the knowledge on digital video is essential to accommodate the understanding and the flow of the upcoming sections of this thesis, the next subsection briefly explain about digital videos.

1.1.1 Background of digital videos

The historical use of conventional forms of video was mainly centred on the filmmaking industry. Later, with the advancement of video technologies, such as more affordable video acquisition devices, cheaper storage, video editing tools, and varieties of communication methods, the use of digital video expanded from commercial to domestic purposes. In the present day, digital videos are being used for recording private events of individuals or a group of people, public and private surveillance systems, medical and scientific fields, television industry, filmmaking industry, and many other multimedia related applications. It is noticeable that the applications mentioned above are mostly the instances where the digital videos are generated or captured. At the same time, these applications facilitate the intended audience to reach their preferred video content by searching, which has made video archiving and streaming have a popular and lucrative endeavour. Currently, many popular video archiving and streaming systems get their archives popular among the general public. The applications mentioned above have been improved over the past decade by introducing many tools that provide some of the necessary facilities not only to make the use of the applications more convenient but also to improve the resource utilization and undergoing business models.

1.1.2 Secondary tools for video applications

The lifecycle of digital data includes data generation or acquisition, information extraction, manipulation, storage, and communication. The tools available for aiding

each of these phases of video data will not be discussed as they are out of the scope of this thesis. However, the storage of data typically involves several other sub-stages, namely indexing and searching. The same applies to video data as it does not self-describe the content. Indexing is required to speed up the searching process of data in extensive archives. It is mandatory to understand the content of videos in order to properly index and retrieve them on relevant search queries. This process is known as "semantic indexing", for it arranges videos according to the semantics of the content. Almost all video archiving and streaming systems use manual, semi-automated, or fully automated video annotation tools to assist semantic indexing. These tools ultimately enable arranging the videos in a way that is convenient to retrieve by providing a related search query. The same searching tools are further extended to find possible violations of copyrights and duplicates of the data in the same or other databases. In addition to improving the video searching experience and the quality of the content, the video annotation and classification contributes to model the individual viewers for the recommendation of videos and commercial advertisements.

Surveillance footages are another source of digital videos which requires numerous tools for the purpose of improving the usability of such systems. There are intelligent tools for real-time detection of abnormal events, which are possible security threats from Closed Circuit Television (CCTV) video footage. In addition to real-time monitoring, searching for a specific event from an extensive archive of surveillance, footages also require specialized tools. A tool that facilitates the automated finding of a known object or an event from an archive of video footage is an intelligent tool that requires a different level of machine understanding of video data.

Many of the tools used with video-related applications require machines to understand the content of videos. Using human interaction in most of these cases has become infeasible due to the complexity and the incredible demand created by the applications. Therefore, it is beneficial to avoid human interaction as much as possible in video understanding by making the machines' involvement stronger.

1.2 Digital video understanding

The objects and their events are considered the content of visual data, which is a complex data type, of which the content is very subjective to describe. A video can be understood in different ways, such as by the objects that appear in the video, their interactions, or both the objects and their interaction in the context of the background in the video. Understanding video data is a hierarchical process that is typically followed in a top-down approach by a human individual but in a bottom-up approach by machines. It is worthwhile to understand how a human viewer understands a video to find about potential techniques of making a machine to understand a video.

1.2.1 Video from the human's point of view

The set of pixels in each frame collectively recreates the appearance of objects in the video for a human viewer, whose visual system does not pay attention to individual pixels unless he/she is force it to do so. However, the pixels can collaborate on creating visible geometries by distributing their intensities and colours. A human can see these geometries and map them with their pre-gained knowledge to segment each of the objects and then ultimately recognizes them [1]. The human visual system attempts recognizing an object by predominantly focusing on the shape, and on failure, by considering the colour (the chromatic attributes i.e. hue and saturation) or the texture of it [2], [3]. The human brain is capable of selecting the best features out of the shape, colour, and texture for processing the visual input, for the purpose of gaining enough knowledge to fill the omitted details of the present context [1]. Video is more complicated as it contains not only the objects but also their movements; hence it is not sufficient to recognize only static objects in order to completely understand a video. Fortunately, the human visual system has a phenomenon called persistence of vision, which enables tracking objects that have already been detected and recognized from a frame over the other consecutive frames [4]. The "persistence of vision" is nothing but a brief chemical memory on the retina, which can keep sending a signal of a static frame to the brain until the next frame has appeared. It helps the brain to find the correspondence between consecutive frames and ultimately build a model of the motion of each object in the shot [4]. The other phenomenon in the human visual

system called “Phi” conceptually completes a motion presented from a set of discrete frames in order to convince a smooth and continuous motion [5]. Phi works with the experience gained from previous visual observations; hence it may be subjective as different people perceive a motion differently.

Detecting, recognizing, and tracking objects are mandatory but not sufficient to develop a complete semantic description of a video. When a set of objects are seen in a video, a human recognizes them and detects their motion at the beginning. Then the arrangement of different objects and their motions map with visual experience gained previously for the purpose of roughly classifying the video into one of the high-level semantics, which is called the context or the topic of the video.

In summary, the human visual system is capable of detecting, segmenting, recognizing, and tracking objects over consecutive frames to build a complete high-level semantic description of a video. However, digital machines always rely on the building block of a digital video, which is a pixel, and the other representations, including the basic geometries that should be derived from the distribution of pixels to understand visual data completely [6]–[8].

1.2.2 Video from the machine’s point of view

A video from a machine’s point of view is nothing but a set of pixels that of which the colours vary over time. Therefore, a machine does not know at least where the meaningful scenes in the time domain of the video are at the beginning; hence the first-ever step of common video understanding of a machine is the structural analysis of the video, which involves shot boundary detection and finding the keyframes. The knowledge about shots and keyframes of a video necessarily makes the process of understanding faster, as not all frames in a shot should be processed. The syntactic features of a video include the shape, colour, texture, and temporal relationships. These features should be extracted in the next step. It is challenging to select the exact set of syntactic features that are optimal for the video. However, a machine cannot directly extract semantics without extracting syntactic features; hence generic syntactic features are used and then optimized for different applications. The next step is extracting the semantic features by classifying the syntactic features, which generally

requires finding five classes of semantics, namely agent, object, place, time, and event, from a video, in order to make a complete description of a video [9]. The agent can be a person or a particular object which dominates the major scene in the video, whereas objects can be any item other than the agent in the scene, which can aid in finding the meaning of the scene. The place is the location on which the video is shot. However, the place can be as general as indoor and outdoor, or it can be very specific based on the requirement. The time is typically not exact but can be morning, early morning, day, evening, and night, etc. The event is the classification of motion of objects and their identities in the video. A machine can finally create a model of a video, which generally represent an ontology consisting of the relationship among the semantics extracted from the video recorded against the playback time. This ontology can be considered as a complete description of a video, which a machine can read and understood. However, different applications require different levels of machine understanding of a video. It is not always required to obtain a complete description, but knowing only certain aspects of the video, such as the metadata or the existence of some objects or low-level concepts in the video may sufficient to an application [10], [11].

1.2.3 Different aspects of machine's understanding

A complete description of a video is required when it is to be stored in a database and providing the users the facility to search for a video by describing its content. This may not be the case if somebody requires obtaining a video, which includes an object provided with a query image. In other words, it is not mandatory to use a textual query, but instead, a video or image query can also be used. It is to be noted that an automatic annotation system can be used to extract the textual representation of the query video or image and use it as an intermediate form to search for data in the database. However, the approach does not support the applications in which the users require finding a specific or the very same object provided in an image within another video. The extraction of textual annotation will not work in this case, as the annotation process over-abstract the object as it cannot completely describe it. The over-abstraction results in robust search; hence video frames with similar objects can be retrieved instead of retrieving only the frames which contain the specific object. This kind of problem

should be overcome without fully recognizing the object but by looking for exact syntactic features. This is a different aspect of machine understanding of visual data, as it really does not dig deep into the real-world meaning of objects, but instead solves the problem at a lower level. This feature is required in many of the applications, such as finding a known instance from a long CCTV footage, object pose estimation, 3D reconstruction and motion to structure, etc.

The field of studying these digital visual understanding is known as “computer vision”. The next subsection briefly describes the field of computer vision, which is the base of this thesis.

1.3 Computer vision

Computer vision can be roughly described as a computer’s understanding of visual data. However, the computer vision field consists of two significant subfields, namely image processing and artificial intelligence.

1.3.1 Image processing

An input image may not be ready for interpretation or analysis right after it has been captured. The image may require some manipulations to achieve a specific output image that is ready for the interpretation or the analysis. The process is generally called as image processing. It is a subfield of signal processing, which includes techniques for noise removal, enhancement, feature extraction, and segmentation, etc. A typical computer vision system includes pre-processing the input, feature extraction, optionally segment, and the interpretation. The pre-processing can be done using image processing. The visual features are required to make the interpretation.

Visual feature: It is not a common practice using the whole set of data for the interpretation directly. Although the visual data is extremely high in density, they contain many redundancies. Practically, a human observer does not pay attention to each pixel, but certain high-level features, such as the colour which dominates out of a collection of pixels and shapes created by the uniform differences between nearby pixels. However, machines work on a bottom-up model as they see the details before the overall idea. Therefore, machines follow a procedure called data abstraction, which

keeps only the necessary information extracted out of the data for an application. The remaining steps of the computer vision system rely on these abstractions instead of the raw data. The representation of the abstract data is called feature descriptor. Conventional visual features can be classified into three categories, i.e., colour, shape, and texture features. The properties of colour distribution, such as the average and standard deviation or the distribution itself, can be considered as visual features [12], [13]. Shape and texture features can also be derived by analysing the neighbouring pixels. Video data consists of a time dimension; hence the extracted features may contain temporal information in addition to the spectral and spatial information. Studies have shown that different applications may require details in different depths with different invariant properties, in order to interpret the visual data accurately.

The interpretation of visual features is not straightforward due to the absence of linear mapping between the different combinations of the feature values and their true real-world meaning. Therefore, it requires machines to be intelligent in interpreting the features. Selected branches of the study of artificial intelligence are utilized to find a reasonable mapping between the different combinations of feature values and their correct interpretation.

1.3.2 Artificial intelligence

Artificial intelligence is a broad topic that studies about developing machines with human-like intelligence. The conventional artificial intelligence focuses on modelling human thinking rationale using computer software. However, modern research concentrates more on modelling the learning ability of the living beings to simulate machines to solve real-world problems; the field is known as machine learning. The interpretation of visual features can also be categorized as a problem that can be solved by learning. The term “learning” describes gaining knowledge by observing examples or gaining experience, by doing a task many times in different ways and getting rewarded based on its success; hence the optimal method can be found. This approach can be used in the interpretation of visual features by providing a sufficiently large number of examples for a well-defined machine learning technique. The studies

referred to in this thesis have used several machine learning techniques combined with novel visual features to overcome several problems related to video data interpretation.

1.4 Handcrafted vs learned features

Modern machine learning algorithms can learn the whole end-to-end process by observing the data. Such mechanisms are generally called as deep learning. Deep learning made the handcrafting features almost obsolete in the present. However, there are several inherent problems of feature learning that make handcrafted features still worthwhile.

Deep learning requires enormous amount of labelled data in terms of what input would give what kind of output covering all the possible scenarios. However, in most of the cases, the required data is not easily available. Transfer learning and data augmentation can alleviate this problem. However, designing the transfer learning or data augmentation scheme requires a significant understanding of the distribution of the data. Poorly designed networks may be successful for the data available at the development but may fail to perform well with simple variations to the stimuli in unexpected ways.

Apart from the cost associated with collecting and augmenting the data, if the problem is not that complicated then having handcrafted features would also reduce learning time and computing resource requirements. Moreover, hand-crafted features can be seen as a particular form of representation that a human designer thinks is the best for solving the problem. The same can be utilized in designing a deep learning model as an improvement.

This thesis describes several innovative handcrafted and deep learned features that can be used in different parts of the process of video data interpretation.

1.5 General process of video data interpretation

The set of states in the process of video data interpretation includes pre-processing, segmentation, feature extraction, object and scene recognition, temporal relationship detection, concept extraction, and building ontology. The image processing removes

noise, makes geometrical adjustments, colours corrections, segmentations and does the other normalizations of the data in the pre-processing state. The segmentation is an optional state, used in some applications prior to object detection. The reason for the segmentation step to be optional is the ability of some object recognition to automatically detect the object first by exhaustively or systematically searching over the frame. The segmentation can sometimes be placed after the feature extraction, as the segmentation technique may utilize the extracted features. The object and scene recognition rely on the extracted features and the intelligent interpretation system used in the application. Once the objects and the scenes are detected, their temporal relationship can be obtained by tracking them over different frames. These temporal relationships can also be recognized as a set of high-level events that are capable of being combined with the objects present in the scene to approximate the high-level conceptual meaning of the video. The final stage is to record the objects, events, and the concept against the time stamp in an ontology for future reference. In addition to this general process, different aspects of visual data understanding require different types of processes. The most related example for this thesis is the correspondence matching for finding a known or a specific object from a set of images or a sequence of frames. It differs from the general process as it does not contain an interpretation stage at which the content is interpreted to a high-level object or a concept. Instead, the output will be the mark of presence of an object, its location, and optionally the estimated pose in the reference frame. However, these information about the objects such as the location and the pose also can be incorporated to the semantics retrieved from a video. There are several stages which demand improvements considering the accuracy of the outputs and the computation cost, and they will be formulated as the problem this thesis discusses in the study.

1.6 Problem in brief

The objects in a video can be recognized using the pixel information in a frame, and the events in the video can be recognized by considering the arrangement of the objects in a frame and their different appearance over the consecutive frames. The task of object recognition is typically accomplished through two main steps, i.e., feature extraction and classification, as mentioned in the above subsection.

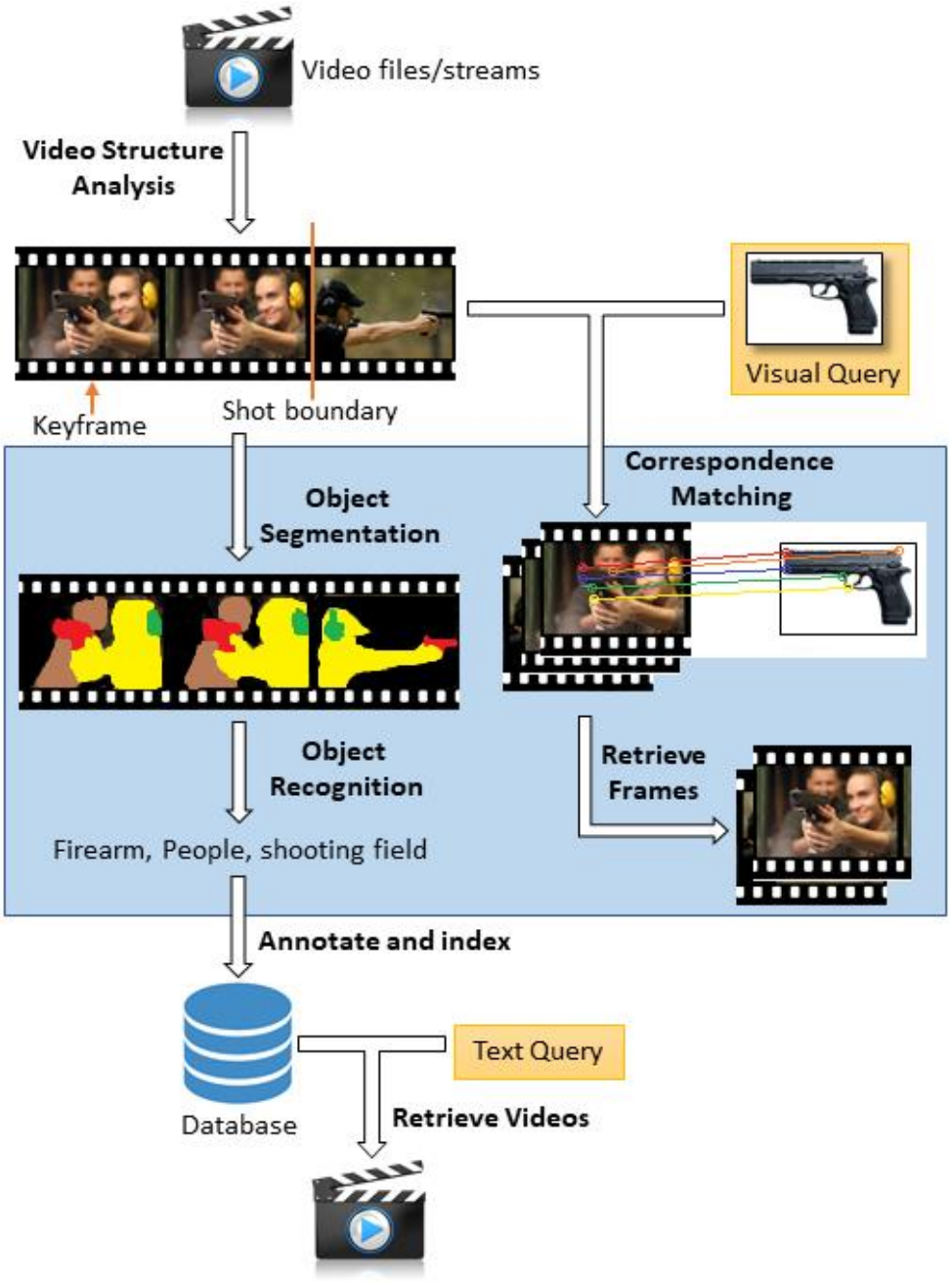


Figure 1.1: Two major sub-processes used in video retrieval applications, namely (1) text query-based retrieval by annotation and indexing, and (2) visual query based retrieval by correspondence matching. The focus of the study is shown inside the blue frame.

The literature and empirical studies make it evident that both the steps are computationally expensive and time-consuming processes, due to the high data density [14]–[16]. Offline video content analysis can be done efficiently due to the feasibility of the prior analysis of the video structure, which reduces the number of frames to be analysed with computationally expensive algorithms. It is impossible to analyse the video structure using a real-time video stream; hence every frame should be analysed with computationally heavy algorithms, in order to analyse the content of the video [17], [18]. However, the real-time video content analysis is a high demanding requirement due to several popular applications such as autonomous vehicles and smart CCTV surveillance systems [17]. Therefore, this study has focused on reducing the computational overhead of the object detection and recognition algorithms, in order to be used in real-time video content analysis applications.

Figure 1.1 illustrates a simplified illustration of two major sub processes of video retrieval applications. The main process starts with video structure analysis and then divides into two different paths: one for text query-based retrieval and the other, for visual query-based retrieval.

1.6.1 Text query based retrieval

In the text query-based retrieval, each of the individual objects in the frames are detected and recognized with an optional segmentation step. Then, the labels of the recognized objects are used to create an ontology with the support of other metadata retrieved from the video file, and finally they are stored in a database with respect to the link of the video file. When a textual query is arrived at, it is systematically matched with the records in the database and then similar video files are retrieved.

1.6.2 Visual query based retrieval

In contrast to the text query-based retrieval, the visual query-based path simply uses either the raw frames or the key frames to carry out correspondence matching with the query image directly and retrieve the frames that includes an instance of the queried image. Although there are many variants of visual query-based retrieval techniques, the correspondence matching technique is more robust due to operates without

requiring any knowledge of the visual content. Moreover, it enables to search for video frames that contains a known object which has not been seen by the retrieval system before. However, the correspondence matching requires to obtain key-points and local key-point descriptors from both the query and each frame in the video.

1.6.3 Scope of the study

The blue frame in Figure 1.1 surrounds the components that are focused on in the study unravelled this thesis. In summary, there are three specific subsidiary problems addressed in the study, namely,

- Video frame segmentation for detecting individual objects
- real-time object recognition in frames without the support of accelerated hardware
- known or specific object detection in real-time using correspondence matching.

All the above problems are related to video and image retrieval. However, when it comes to video annotation the first two are mainly required for retrieval applications. The third must be solved for instance retrieval applications, such as searching for an object from a CCTV footage.

Presently, objects in videos can be recognized in real-time on high-performance computer systems. However, there are several drawbacks of the existing algorithms; they do not work with average performance hardware and they are not robust to many geometrical variances. The lower robustness causes when treating similar objects in different orientations as different objects. When the process of object recognition is considered, the feature extraction and description can be identified as one of the computationally critical tasks.

In addition to the general object recognition, known or specific object detection is also required to be carried out, most of the times, on a real-time basis, and the existing methods struggle against several problems, namely, the computational efficiency, robustness, and discriminative power. The problem of known object detection differs from a simple object recognition, due to the unavailability of many samples of the

images to train a machine learning model. Moreover, it is not practical to train or to fine-tune a machine-learning based object recognition model in real-time, due to the enormous amount of time requirement. Therefore, known object detection is typically performed by correspondence matching, which matches similar local patches between a pair of frames. However, there might be several hundreds of local patches that need to be described with a discriminative yet robust descriptor with lower dimensionality. Hence the large quantity makes the process highly time consuming, even for a single pair of frames. The lower dimensionality is a must for real-time correspondence matching, due to the mandatory exhaustive matching phase. Finding the optimal local patch descriptor is not straightforward; hence many have been handcrafted and evaluated throughout past research that have been discussed in Chapter 2. The state-of-the-art techniques use learning techniques, such as deep learning, in order to find an optimal feature descriptor. However, learning optimal features through state-of-the-art techniques requires an extensive dataset with the ground truths.

1.7 Study aim and objectives

The above brief problem discussion reveals the necessity of improving object detection and recognition to meet the strict constrains of various applications. In this study, several critical points that carry the potential of being improved, such as the efficiency of feature extraction, robustness, and dimensionality of feature descriptor, and the supervision of feature learning, have been identified. Therefore, this research focusses mainly on improving the feature extraction and description by means of computational efficiency and invariant properties for object recognition and improving local patch description for known or specific object detection.

The aim of the study is to devise efficient algorithms to detect and recognize objects in a video frame by minimizing the hardware resource requirement using a dimensionality reduction technique.

The study focuses mainly on achieving the following objectives to accomplish the above aim.

- Critically review the existing systems, concepts, and tools for the depiction of video for semantic retrieval

  It is intended to thoroughly examine the existing approaches for depicting videos in order to find the critical steps that mostly affect the applications negatively. The study objective was set to be achieved through a thorough literature review and experimenting by implementing a significant method on a single test platform. The experimental setup and the evaluation metrics are explained in Chapter 6. The evaluation results of the novel methods are discussed in Chapter 7 including an extensive comparison with the existing works.

- Devise a novel visual descriptor with low dimensionality while maintaining a state-of-art retrieval accuracy

  The intention of this objective is to develop a new algorithm to achieve a computationally cheaper visual feature descriptor to recognize objects in each frame of a video with one of predefined set of labels. This objective was to be achieved by exploring a novel dimensionality reduction technique that can be utilized to both extract and describe visual data by including the most critical information for the recognition.

- Devise a segmentation algorithm to locate multiple objects in video frames

  The previous objective has limitations, as achieving it makes a simple object that appear in a video frame to be recognized through a single feature descriptor. However, there might be multiple dominating objects in a single frame. Therefore, it is intended to segment the objects by utilizing all three types, namely the spatial, spectral, and temporal information in a single algorithm.

- Devise an efficient unsupervised technique to detect a known or specific object from video frames

  This objective was set up with the intention of developing an algorithm to locally describe visual objects by learning an efficient local feature descriptor that can be used to detect a specific object by applying correspondence matching. Its focus is on the computation efficiency, mode of supervision, dimensionality, robustness, and discriminative power.

1.8 Organization of the thesis

This thesis is presented through eight chapters to specify the significant aspects and outcomes of this research study. Some of the essential aspects of global visual features for object recognition, object segmentation, and correspondence matching have been discussed in Chapter 2. Several significant handcrafted and deep learned global features, segmentation methods, and local features, are broadly discussed in this chapter. Moreover, the problems in object recognition, video segmentation, and correspondence matching have been elaborated using literature evidence. Finally, the latest research attempts to enhance the above-mentioned tasks have been expounded in the literature review.

Chapter 3 presents the organization of the overall research, emphasizing the proposed method for object recognition. The theoretical background of the proposed work has been elaborated by providing the details of the source of inspiration. Moreover, the use of colour dithering technique as a dimensionality reduction technique, and the ultimate utilization of it to create a complete and invariant feature descriptor have been explained. It is noteworthy that this chapter reports several versions of the proposed work as the significant milestones in the journey of improvement and evolvement throughout the study.

Chapter 4 presents the proposed object segmentation technique with the details of the theoretical concepts behind it. The relationship between the proposed feature descriptor with the proposed segmentation technique has been elaborated in this chapter. Moreover, the ability of utilization of spectral, spatial, and temporal

information available in a video, combined with an existing clustering algorithm to accurately segment multiple objects, has been described.

Presenting the proposed local feature descriptor, Chapter 5 elaborates on the correspondence matching. The importance of dimensionality and the real and binary types of local descriptors have been discussed. Moreover, the deep learning of local binary descriptors and their use in correspondence matching have been discussed in this chapter.

Chapter 6 explains the experimental setup and validation methods, whereas Chapter 7 reports the results of the experiments with a critical and a constructive discussion. Finally, the recommendations and future extensions of this study have been presented in Chapter 8.

1.9 Summary

This chapter provides a comprehensive overview of the research. Firstly, it covers the formation of digital video by explaining the concept of pixel, frame, shot, and shot boundaries. Secondly, it explains the importance of an autonomous understanding of video for applications. It further elucidates how the human interprets visual data in a top-down approach, while machines interpret the same in a bottom-up approach. Then it covers the necessary background of video understanding, namely image processing and artificial intelligence. At the core of the Chapter, the specific problem, namely the computational expensiveness of the object recognition and known object detection, is explained, after which the intention of the study, i.e., aiming at devising efficient algorithms to detect and recognize objects by minimizing the hardware resource requirement followed by four significant objectives, is presented. Finally, the organization of this thesis is introduced to navigate the relevant information and its presentation.

CHAPTER 2

RESEARCH BACKGROUND

The focus of the study is developing a set of algorithms for object recognition, segmentation, and detection, with minimizing hardware resource requirements. This chapter presents the background of the study by enforcing the critical problems that this study addresses, referring to the evidence from the extant literature.

2.1 Scene and object recognition

Scene and object recognition is one of the most challenging tasks in computer vision, due to the complexity of the problem [19]–[21]. The improvement of object recognition can benefit many other applications, such as visual content search and retrieval [6], [22]–[24]. Due to the increasing amount of visual content on the internet, video productions, visual-based security systems, and medical applications, not only the classification accuracy but also computational efficiency of the concept classification problem is found essential [7]. Visual concept classification generally includes three steps, namely, feature extraction, description, and classification. It is challenging to select a practical visual feature that is robust to geometrical variances, keeping the low dimensionality of the feature space and less computation time, without sacrificing the discriminative power of the descriptor [25], [26].

2.1.1 Handcrafted feature descriptors

Selecting the optimal visual feature for an object detection application is not a straightforward method; hence, many different features have been handcrafted and evaluated for achieving different aspects of applications. The term “handcrafting” emphasizes human involvement in the selecting and validating of features without using an autonomous optimization technique. The basic visual features are colour, texture, and shape. When a set of features are described numerically, it is called as a feature descriptor. It is not necessary for a descriptor to rely only on one feature. Several features can be concatenated to form a single descriptor [22]. Histogram [13] and colour correlogram [27] are popular colour feature descriptors. Colour correlograms solve the issue of the absence of spatial details in colour histograms [27].

However, colour correlogram suffers from high computational complexity [28]. Spatial-chromatic hybrid techniques are also available in image retrieval researches [12], [29]. Conventional colour-spatial integrated methods are known to have either no correlation between the colour and spatial information or yield poor results over some types of images [12]. One of the major texture feature descriptors is Homogeneous Texture Descriptor (HTD)[30]. HTD concerns about the homogeneity of the illumination variations over the region of interest and then measure and describe numerically in order to discriminate the texture. The coarseness (how rough the texture), directionality (the dominant direction of the texture) and the regularity are concerned in Texture Browsing Descriptor (TBD) [31]. Occasionally the textures appeared on some images may not pose any homogeneity. In these occasions, the edge distribution can be used. In addition to the HTD and TBD which are provided in MPEG-7 standard, another descriptor called as Edge Histogram Descriptor (EHD) [32] has been proposed for describing non-homogeneous texture. Nevertheless, these texture descriptors can be employed strictly for texture classification, hence, for general cases, it may require a combination of several descriptors, which is also problematic due to high computational cost. Conventional shape descriptors [33] require well-segmented objects, and they also impose to narrow the domain of usage, compared to colour and texture information based descriptors [34]. The Histogram of Oriented Gradients (HOG) [35] has been widely used for describing the object appearance and shape by utilizing the distribution of edge orientations. Improvements made, inspired by HOG, have been identified in many recent studies, such as Pyramidal HOG (PHOG) [36] and Gabor-PHOG (GPHOG) [21]. All these improvements adversely affected the dimension of the final descriptor. When comparing with the dimension of HOG descriptor, the dimensionalities of these new variants are greater by an order of magnitude. Although the HOG based descriptors are very capable of discriminating local shapes. It can abstract a shape by the distribution of gradients with their orientations. However, this rigid description limits the degree of robustness to geometrical transformations, such as translations and rotations, though it can be adjusted by varying the block size and orientation bin size [35].

Throughout the evolution of local features and descriptors, researchers have invented several significant feature descriptors. Scale Invariant Feature Transform (SIFT) [37] can be attributed as the mostly used local descriptor for vast array of applications including, image registration, 3D reconstruction, augmented reality and panoramic stitching[38], [39]. SIFT uses difference of Gaussian operation to detect keypoints and HOG for the description. Speeded Up Robust Feature (SURF) [40] is another local feature descriptor that is faster than SIFT. Although the SURF is faster, it is a frequently discussed problem that both SIFT and SURF experience an extreme cost of computation that is unacceptable for many of the real-time applications [41]. A set of local feature vectors can be described as a global feature vector by using a Bag of Feature (BoF) model [42], [43], which has demonstrated a reasonably high performance in image classification and visual tracking applications [42], [44]. However, the high dimensionality of local feature descriptors adversely affects the computation time of the BoF histogram. The facts prove that dimensionality has a great correlation with the computation cost of visual description. Moreover, it correlates with the time taken for the classification as well. "Curse of dimensionality" is a popular term to describe this adverse effect of the dimensionality [45].

2.1.2 Dimensionality reduction

Dimensionality reduction is the general term used for the process of reducing the number of random features by obtaining a set of principal features or transforming them to a lower-dimensional space[46]. It is one of the mandatory steps in the feature selection and feature extraction. Principal Component Analysis (PCA) [41] is a widely used dimensionality reduction technique that is conducted by a statistical procedure. The significant use of PCA can be found in face detection and recognition related research. As an alternative to PCA another method has been suggested that preserves the local structure of the dataset. The method is named as Locality Preserving Projections (LPP) [47] and it has also been used for several different problems that require to keep the neighbourhood structure after projecting to a low-dimensional space. fuzzy lattices technique[48] on the other hand is a space transformation method that is functionally similar to both PCA and LPP. It also has been used in various image processing applications. Any of these techniques explore the correspondences among

the data points given to seek the transformation function. Lack of good correspondence results in the inability of finding a reasonably lower dimensional space to get transformed. This is a limitation of any of the said methods as they fail to find an expressive subspace with non-favourable datapoints[49]. Use of the said dimensional reduction methods, integrated to the preliminary feature extraction techniques, generally affect the cost of computation adversely [50].

The human visual system has an inferior sensitivity to the spatial resolution, yet it has the capability of exchanging spectral resolution with spatial resolution in the discrete representation of a picture. These two properties have been well utilized to create an illusion of the presence of colours, which cannot be reproduced in reflection-based applications such as printing and some emission-based application such as display devices and multimedia projectors. The technique which utilizes this feature for the aforementioned applications is known as dithering [51]. A recent study has developed Compacted Dither Pattern Code (CDPC) [52], that uses an approximated implementation of a dithering technique. Its primary attempt is to reduce the number of elements in the feature vector. It ultimately produces a descriptor with a significantly lower dimension compared to several leading visual data descriptors. CDPC descriptor includes colour information because it primarily converts different colours to set of indices. Moreover, it consists of texture like details, due to the use if micro-patterns which consist of several coloured pixels. The base of CDPC is its approximation of colour dithering. Since it overly abstracts the distribution of dither patterns, it performed better in discriminating irregular shapes such as landscapes. It has also been proven that the proposed technique in the studies about CDPC can dramatically reduce the dimensionality with a very low computational overhead, compared to the traditional dimensionality reduction techniques, while achieving competitive classification performance compared to the state-of-the-art [52], [53].

2.1.3 Deep learned features

Besides the handcrafted features based approaches, the feature learning approaches, such as convolution neural network(CNN) and deep Autoencoder, have become the

most promising technique in raw data classification [54]. These are commonly called as Deep Neural Networks (DNN). CNN is a special kind of artificial neural network (ANN) that consists of a set of learnable convolutional kernels with different sizes attached to a fully connected conventional neural network. When an image or a video frame is fed, the CNN convolves the input with all the kernels, and the resultant convolutional response is fed to the fully connected ANN for the purpose of classification. In this context, the convolutional response can be considered as a visual feature. CNN learns the weights of the convolutional kernels, which provides the best average classification performance by observing a vast dataset. Several successful deep learning-based object recognizers have been introduced throughout the past decade [54]–[56]. DNN is known to be an approach that requires significantly high performable computer resources at the training phase; hence, it typically is used when a high-performance computer or a suitable accelerated hardware platform is available. Moreover, a typical DNN is almost infeasible to execute in real-time on a generic hardware platform without a thorough simplification of the model. In general, a CNN requires lots of annotated data that samples all possible variations of input data, in its training. This is a major problem of any DNN today. It has been alleviated by using the data augmentation and transfer learning. Data augmentation tends to make CNN geometrically invariant. However, the degree of geometrical invariance provided by any of CNN model is very low, compared to many of the common handcrafted features, due to the features which are limited to convolutional responses [57].

2.2 Video segmentation

It is impractical to supervise an object segmentation method for the purpose of reliably segmenting of all the objects independent from any domain [58]. However, such applications that require to be generic have become a common demand. Hence the problem has been approached with several potential self-supervised, semi-supervised, or fully unsupervised segmentation methods [58], [59]. Moreover, the low consistency of existing unsupervised spatial segmentation approaches, are computationally impractical for the real-world use [60]. In the context of videos, there are very limited studies available for unsupervised real-time object segmentation that utilize motion

cues[61], [62]. Hence, it requires extensive works to solve this problem up to a usable state.

Visual objects are inherently complex due high density of data. These data typically expressed many different features. It is not a straightforward process to find a reliable relationship among the different features and the semantics. Applying a clustering technique for cluster multiple features is a well-known technique for visual segmentation. The techniques presented in [63], [64] have introduced several segmentation techniques that use pixel locations and their variation over different frames in videos. However, all these methods cannot segment visual data to conceptually meaningful objects, in the absence of support from a supervised method.

2.2.1 Semantic segmentation

Semantic segmentation means separation of different objects appeared in a single or a sequence of images. The low-level segmentation explores the similar properties of adjacent regions. However, semantic segmentation requires the knowledge about each object in order to classify each pixel to one of the objects in the scene. Moreover, a single object may consist of several homogeneous regions, which will be treated as multiple objects by any low-level segmentation algorithm. Semantic segmentation is typically associated with supervised learning. The studies in [65], [66] utilize a huge annotated dataset to learn to classify each pixel in an image or a video frame to one of the semantic objects which have been seen previously.

Unsupervised segmentation methods are essential when the annotated data is not available, or some foreign objects which were not present in the dataset should appear in the input frame.

2.2.2 Supervised and unsupervised segmentation

The process of supervised segmentation learns about many different properties of object regions by observing many samples of annotated objects. Unsupervised segmentation techniques work with syntactical features and other available clues to segment, without any predefined knowledge of the objects. In most of the studies, colour and texture-based syntactic features have been used for unsupervised object

segmentation. It is noticeable that the techniques can results in inaccurate segmentation results, as there may be multiple objects with the same syntactic features, and on the other hand, there may be objects which contain multiple syntactic features. However, it is advantageous, as the method that will work universally, though the segmentation accuracy is comparably lower than most of the supervised methods. The studies in [58], [60] use unsupervised techniques to segment objects and have achieved a consistent performance.

Once the data is being abstracted by expressing them with low-level features, it can be clustered for purpose of segmenting objects. Clustering is more efficient than pixel-wise object segmentation methods due to the lower density of data. Clustering can be conducted in both supervised and unsupervised manners; however, the main concern of this study is unsupervised object segmentation, due to its universal applicability.

2.2.3 Data clustering

Data clustering groups any given set of data points within the space by considering one or more attributes. When this grouping is done solely based on the similarity of the attributes that are being considered, it is called as unsupervised clustering technique. Some of the datasets should be grouped together by considering different combination or values in different attributes. Since it is difficult to handcraft such mapping function, it is learnt from a set of sample groups. This kind of grouping is called as supervised clustering. The most common clustering techniques are centre based. These techniques first form a set of cluster centres and the iteratively tuning them with grouping the data points until satisfying certain conditions such as minimizing a clustering error. K-means is one of the mostly cited centre-based clustering methods. If it is assumed that the data is generated from a model, then a model-based clustering technique are used. The purpose of these techniques is to recover the model that has been used to generate the data. Expectation-Maximization (EM) is one of the mostly cited model-based clustering techniques. Occasionally some data points show different densities over different regions of its distribution. If it is assumed that the proximity of these datapoints in the space quantifies their similarity, then density-based clustering can be a better option. Any of density-based clustering

operates by grouping datapoints based on their proximity and some predefined cluster separation parameters. Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is an example for density-based clustering techniques.

K-means may cause several problems when it is applied for a set of data. The most significant problem is that it requires the number of clusters that the data is being clustered. This limitation prevents using K-means for data points where their behaviour is completely unknown. Moreover, this limitation suggests using K-means as a secondary clustering algorithm on ground that some heuristics of the dataset exist. Furthermore, K-means is not capable of avoiding outliers. The algorithm attaches all the data points to some intermediate clusters formed. At the end of the algorithm there are no datapoints that do not belong to any of the K clusters.

Unlike K-means, DBSCAN avoids forming clusters with noisy data points. The core assumption of DBSCAN is that if data points are measured to be nearby according to the given metric, then those points should belong to a single cluster. If there is any point which does not overcome the distance threshold to any of the data point that has been already in a cluster, then those data points will not be clustered and considered to be outliers. This is an advantage over typical K-means clustering. Moreover, DBSCAN does not require to manually define the number of clusters. Instead, it has two other parameters which are less specific to the data compared to the number of clusters. The first parameter is the minimum neighbour detachment that specifies the threshold for the distance between two neighbours to determine whether they belong to the same cluster or not. If the distance does not win over the threshold, then DBSCAN algorithm forms a new temporary cluster with the detached neighbour. This temporary cluster will be kept further only if that cluster can collect a set of data points in which the number of data points is higher than another threshold. In summary, DBSCAN request two thresholds to initialize the clustering algorithm. However, these two parameters can be fine-tuned by doing several trials.

High dimensionality of data adversely affects the any of the clustering algorithms. However, EM based clustering, have been used with extremely high dimensional data in several studies [67]. Moreover, it is known to robust for noisy data. EM techniques

assume that the data is generated from a mode and then attempt to recover it in terms of the parameters of the distributions. It typically obtain the parameters of not just a single distribution but a mixture of several different distributions that are generally falls into multivariate category [68]. Hence, EM is capable of clustering data that is generated from very complex models that follows multiple distributions together. Better distinguishing of feature points by using a suitable clustering technique can help to segment objects with reasonable accuracy, which is the reason why this study relates to feature clustering-based segmentation.

In addition to the object recognition and segmentation, the study works on detecting known or specific objects in video, using correspondence matching.

2.3 Known object detection

Visual abstraction through a global feature descriptor can be used to recognize an object and then label it using one or several classes used for building the recognition model. However, a global feature descriptor cannot be used to identify a specific object given as a visual query from an image or a video frame. Objects in a frame can appear in different orientations, scale, positions, and illuminations; hence it is not very fortunate to find a given object by pixel-to-pixel matching. Correspondence matching is one successful method capable of achieving the task of known object detection.

2.3.1 Correspondence matching

The primary use of a low-level visual feature is to digest the high-density visual data, keeping sufficiently discriminative details that can be used for reliably computing the similarities. In the recent past, the main concern in the context of local feature descriptors has been the accuracy of matching, and as a result, they have been evolved from handpicked feature descriptors to deep learned features without concerning the cost of computation. Scale Invariant Feature Transform (SIFT) [37] is one of the mostly used handcrafted feature descriptor. The deep learned local feature descriptors include CKN-grad [55], MatchNet [69] and DeepCompare [70]. These three descriptors have been widely used mainly because they can be trained with unsupervised or self-supervised learning algorithms.

Efficient Descriptors: With the advancement of both hardware and software in the field of multimedia, many real-time audio/video applications have been introduced. Some of the video or visual analysis related applications required real-time feature extraction and description techniques. Although the existing descriptors such as HOG [21] are very discriminative, they have not been optimized for real time applications. With reference to the aforementioned incapability and the history of improvements that focused on efficiency, there have been several studies available, including SURF [40], ORB [25] and CDPC [53]. SURF follows an approach that speedup the extraction process by using a different representation called as integral image. Furthermore, it uses a lower dimensional descriptor that is based on Haar response [40]. ORB use a significantly different approach compared to SURF, where it pushes the descriptor to an extreme end by limiting it to contain binary string. Making the descriptor having binary values significantly beneficial in the feature matching phase in terms of computation time. CDPC also utilized an unconventional dimensionality reduction technique for obtain a low dimensional descriptor [53]. It can be seen that all three descriptors have an intersection in their objectives, namely the reduction of dimension. Both the SURF and CDPC produce real-valued descriptors where ORB produces binary valued descriptor. Binary value descriptors are generally preferred for real-time applications due to the computational benefits that can be seen in the feature descriptor matching phase [25], [57].

2.4 Binary-valued local feature descriptors

Local feature descriptors are widely used for key-point to key-point matching purpose, where it is necessary to get the correspondence between two images. In general situation, the matching must be done exhaustively where each feature descriptor from the query image will be compared with all the feature descriptors in all the images in a database or frames in a video. Hence the number of comparisons is extremely huge. The comparison can be done with different similarity measurement techniques. Euclidean distance is one of the mostly used similarity measurement techniques for real-valued local descriptors [25], [57]. It is noteworthy that the calculation of Euclidean distance requires many subtraction, multiplication, addition, and square root operation per pair of descriptors. This cost of computation will be multiplied by the

number of comparisons made in the application and ultimately, it may become less usable. This problem can be overcome by using binary-valued descriptors because the similarity can be evaluated using Hamming distance, which is significantly cheaper in computation. BRIEF [71] is one of the preliminary study that contributed to the fringe of local binary descriptors. However, it fails to detect matching pairs with slight orientation difference hence ORB [25] is invented. Moreover, the ways of describing visual features through a binary string has been further explored and resulted in several other binary descriptors such as BRISK [72] and FREAK [73]. Although these descriptors have different capabilities, internally, they use pair-wise pixel comparisons in order to get the binary values. This similarity measurement does not concern about illumination differences, scale of the content and other high-level features. Therefore these binary descriptors are less robust for frequent visual deformations [74]. Studies have proven that this problem can be sufficiently solved in many different applications by forming the binarizing technique that has been optimized by observing large number of samples. One famous approach is to use machine learning to obtain data to binary descriptor mapping function directly by inspecting samples.

There are two main steps in any of the binary descriptors namely, the projection and binarization. The projection step projects the data to more discriminable space and then the binarization step quantize the projected data to a binary string. LDAHash [75] is a binary description method that can learn such a projection method just by observing the given data. It optimizes the projection by minimizing the intra-class covariance together with maximizing the inter-class covariance. Moreover, it binarize the projected data by using a threshold that minimizes the quantization error. The strengths of some existing handcrafted local binary descriptors have been utilized in recent machine-learned binary descriptors. For example D-BRIEF [76] is an extension of BRIEF [71] descriptor. In this study, the same original BRIEF has been employed on a more discriminative subspace. This subspace has been learned by using Linear Discriminant Analysis (LDA) [75] over the given dataset. The study reports that D-BRIEF outperforms the original BRIEF [76]. In D-BRIEF, an integral image representation is used along with box filters to speed up the computation. However, the use of box filters has been identified to be efficient in computing yet demonstrates

less robustness to rotation. Several recent studies has suggested that AdaBoost [77] can be used to learn binary-valued local descriptors. Receptive Field Descriptor (RFD) [77] is an example binary descriptor that uses AdaBoost in the learning stage of the algorithm. Local Difference Binary (LDB) [78] is another example that is keen on selecting optimal pair of samples to compare and binarization. However, BinBoost [79], another method that utilize AdaBoost in its coding system use the boosting method to optimally weight the features and then to pick the best set of bits from a pool of bits. The purpose of using this technique is mainly targeted at obtaining an extremely compact and precise binary descriptor. RFD differs from Bin-Boost only by the receptive field selection step that is employed in the RFD. Literature suggests that RFD and BinBoost outperform many of the previously invented binary descriptors. However, the large number of weak classifiers that are used in the AdaBoost require excessive computation time to extract the features and to describe them. BinBoost require to employ AdaBoost once per each bit, which results in even more computationally expensive process [79]. This excessive computation time prevents the AdaBoost based local binary descriptors used in real-time applications regardless of the benefits received from a binary-valued descriptor.

Self-supervised hashing techniques obtain hash codes by minimizing the reconstruction error of the input image by backpropagating the learned binary descriptor [74]. Binary hash codes can be used to search for a specific or relate visual content with a constant complexity. Locality Sensitive Hashing (LSH) [80] is an important milestone of self-supervised hashing that outputs a locality sensitive binary hash code. Another related method named as Semantic hashing (SH) [81] utilizes a similar method with the objective of hashing more high level semantics to binary hash codes. SH is initially used for hashing text documents by obtaining binary codes. It learns the binary hashing through a stack of Restricted Boltzmann Machines (RBM). Further, its RBM not only extract features, but also function as the feature to the binary mapping system. DeepBit also output a binary hash code but without using the reconstruction error minimization approach. Instead it uses set of objective functions with custom loss functions to train the binarization part [74]. It utilizes features extracted from VGG16 [56] that have been pre-trained to learn binary codes. The

learning phase involve in optimizing three objectives, i.e., the descriptor to be transformation invariant, evenly distributed, and minimum in quantization loss. These unsupervised descriptors are excellent in case of an unavailability of an annotated dataset. However, it is noteworthy to mention that this method does not outperform any of the existing supervised methods in terms of classification accuracy.

Deep learning: Learning features by observing a sample dataset with a clearly defined objective has been identified as the reason behind the many of succeeded visual feature descriptors. The term "deep learning" specifies that it learns both feature and classification together. It permits learning a set of powerful features from a broad set of images. The primary application of deep learning in multimedia is to classify a given input such as an image or a video frame by assigning a label. Conventional Neural Networks (CNN) consists of a set of convolution filters in which the filter weights can be learnt from a set of annotated data. The classification of CNN based features is carried out by using a stack of dense layers where all the neurons in a layer connected with all the neurons in the next layer. Many of the well performed binary descriptors comprising CNNH [82], Binary L2-Net [83], DSH [84], DeepBit [74], and CNNH+ [85] are learned over convolutional features that are obtained from models such as VGG16. Those methods typically substitute a custom layer that generates binary codes, to the fully connected layers attached to the end of the convolutional layers. However, CNN based models are known for less robust to rotation and also hard to avoid over-fitting [74], [86].

According to this background study, it can be seen that the existing method suffers from one or more limitations such as being real-valued, less discriminative, higher dimension, require a large, labelled dataset for supervision, less robustness and computational expensiveness. Hence, it requires enormous effort on further exploring on local descriptors, preferably having all the features such as unsupervised, binary-valued, robust, less requirement of computing resources and higher discriminative power.

2.5 Summary

This chapter has emphasized the background of the study. Initially, the literature on object recognition and the related concepts, such as handcrafting and deep learning of visual features with their limitations, have been discussed, followed by discussion on the background of object segmentation, emphasizing the importance of visual feature clustering for efficient object segmentation. Finally, the background and the existing methods for known object detection, with correspondence matching, have been discussed, clearly elaborating the limitations that exist with the past works. The next chapter presents and examines the proposed low dimensional feature for object recognition.

CHAPTER 3

LOW-DIMENSIONAL FEATURE FOR OBJECT RECOGNITION

This chapter presents a novel visual descriptor that can rapidly and effectively encode both chromatic and geometric information of visual contents in a low dimensional space using dithering techniques. It elaborates how the three different versions of the proposed method have been evolved throughout the study by overcoming the limitations encountered.

3.1 Overview

The objective of the proposed method is to efficiently describe visual objects through a low dimensional and geometrically invariant feature descriptor. The core hypothesis of the proposed method is that it can use colour dithering, which is used in colour printing, in order to reconstruct many colours from a smaller number of colour inks, as a dimensionality reduction technique of visual feature descriptor. The proposed method has been improved over three major versions, and in the initial version, the descriptor was created by applying a colour quantization technique as an approximation to colour dithering. The initial version is invariant to many geometrical transformations, and it is extremely fast in feature extraction and description, yet the classification performance lag behind the state-of-the-art methods. Therefore, the second version was developed by adding colour density information to the first version, with the hope of improving the classification performance. It has shown better object recognition performance than the first version. However, when the application contains a massive number of object categories, the second version has also shown less performance compared to the classification performance of best existing methods. It has been identified that the simple spatial-chromatic descriptor introduced in the first version over-abstracts the visual information, resulting in degradation of the discrimination power of the final descriptor. Moreover, both versions use a simple colour quantization technique as an approximation to colour dithering to avoid computationally expensive operations. However, in the third version, it uses a fast colour dithering technique, which creates real dither patterns, compared to the first two versions, and combines it with a novel colour-based hessian matrix evaluation to detect

stable colour feature points. In addition to the improvement of features, the descriptor also has been improved by incorporating more details, without sacrificing the invariance properties.

3.2 Colour dithering as a dimensionality reduction technique

Colour dithering is not commonly used for dimensionality reduction in any field, for it is a colour representation technique typically used in printing and display rendering systems. The concept exists because it works with the human visual system when it comes to reconstructing many absent colours by combining several available colours. Theoretically, it can be argued that dithering can be used for reducing redundancies. A visual object may contain many regions with different colours, and each region may contain many pixels that apparently bear the same colour. In the context of data, it is a kind of redundancy, as the colour of the whole region can be defined only once, instead of keeping all the pixels. Colour dithering helps utilize the spatial domain in such a way that the space of homogeneous colour region encodes many absent colours by arranging the actual colours in different patterns and densities, which proves that dithering reduces the redundancy of the spatial domain by exchanging it with spectral resolution.

Most of the algorithms designed for dithering is sequential and relatively computationally expensive compared to other colour quantization techniques. Therefore, simple colour quantization is used, instead of a real dithering technique, by inspiring from CDPC for the initial version of the proposed work.

3.3 Salient Dither Pattern Feature (SDPF)

SDPF descriptor is a spatial-chromatic histogram of a set of feature points that are detected by filtering the locally salient dither patterns. The core concept used in the SDPF was first introduced with CDPC [87], which is a syntactic visual feature devised for shapeless visual concept interpretation. CDPC concerns about decreasing the number of colours by employing colour dithering. The CDPC feature extraction algorithm is involved in several steps; i.e. applying an average filter on the frame to reduce noises, colour quantization of the resultant frame, re-code the colours in

quantized RGB space, store the four-colour codes in each of the patterns which consisted of four neighbouring blocks of colours, sorting the four-colour codes of each pattern in descending order to avoid the permutations and finally create the probability distribution of the patterns for the frame [52].

3.3.1 SDPF feature point extraction

Unlike CDPC, the SDPF considers only dither patterns that are locally salient, instead of selecting the patterns based on the order of the probability of occurring over a specific spatial region. This addition results in extracting a set of key points, as well as suppressing the patterns distributed redundantly over flat regions. However, the SDPF algorithm intrinsically follows the first step of the CDPC algorithm, and adopts the colour quantization, colour coding and the final description steps innovatively to incorporate the spatial information and to reduce the dimensionality further.

Like in the CDPC, to extract SDPF points, averaging colour filter as in (3.1) is first applied.

$$f_{R,G,B}(x,y) = \frac{1}{N_G} \sum_{(p,q)\in S} f_{R,G,B}(p,q) \tag{3.1}$$

where *S* is a set of pixels in a block of a dither pattern (see Figure 3.1), that are denoted with their location in 2D space, and N_G is the cardinality *S*. The $f_{R,G,B}(x,y)$, is a vector containing the values of RGB channels. The quantization step is avoided in the SDPF algorithm at this stage to save the computation time of processing the non-salient contents of the frame.

A dithering pattern is a square shape area that consists of four blocks. The dimension of a block is always an even number of pixels in both vertical and horizontal axis. Each dither pattern consists of four colours. Figure 3.1 illustrates a dither pattern that is considered in the proposed work. Any such dither pattern, that is significantly different from its surrounding patterns, will be considered as an SDPF.

Any single dither pattern is always surrounded by eight dither patterns (p = 1, 2,..., 8). The colour distance (D_p) from the p^{th} surrounding pattern (P_p) to the middle pattern n *(i.e., P_n)* is calculated using (3.2),

$$D_p = \sum_{i=1}^{4} \sum_{c=1}^{3} \left| C_c(E_i(P_p)) - C_c(E_i(P_n)) \right| \tag{3.2}$$

where $C_c(E_i(P_p))$ denotes the c^{th} colour component of the i^{th} element (out of four) in the p^{th} surrounding pattern. The notation C_c is a colour component in RGB colour model. Therefore, the value of c is 1 to denote the red component, 2 for green, and 3 for the blue.

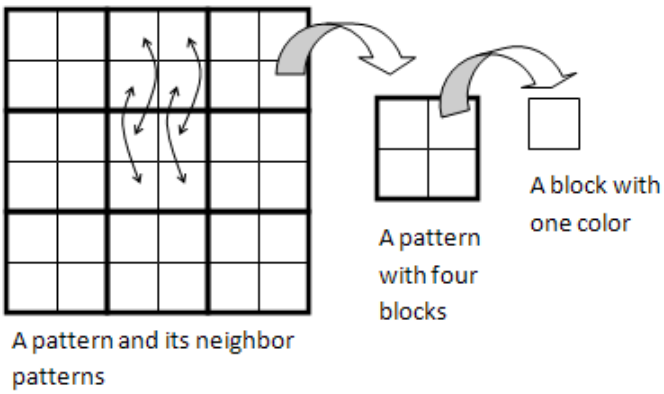


Figure 3.1: 3x3 neighbour dither patterns with the anatomy of a single dither pattern.

Then the dissimilarity of the middle pattern from its all eight neighbour patterns is decided by (3.3).

Let S_S be the set of potential feature points, and T be a threshold value.

$$\forall p: D_p > T \rightarrow P_n \in S_s \tag{3.3}$$

When T is low, it yields many feature points along the edges and textured regions. At the beginning of the study, T was empirically selected where it does not make dense feature points at edges, but corners and small islands with different colours from their

background. However, after carrying out several trial-and-error attempts, a more consistent approach has been identified in which setting *T* to a fraction of the average of colour dissimilarities of each pair of adjacent patterns in eight surrounding patterns as specified in (3.4) yields better classification accuracy.

$$T = \frac{T_K}{8} \sum_{n=0}^{7} \sum_{i=1}^{4} \sum_{c=1}^{3} \left| C_c(E_i(P_{(n+1) \bmod 8})) - C_c(E_i(P_n)) \right| \tag{3.4}$$

The T_K is the fraction of the average of the colour dissimilarities, that is to be assigned to the threshold *T*.

The experiments that was conducted with different combinations of concept categories in both Caltech [88] and Corel-1000 databases [89] have demonstrated that the best average classification accuracy can be obtained when the T_K is set to 1/3. The experimental results can be found in APPENDIX B.

The vector $\{D_1, D_2 \ldots D_8\}$ gives the measurement about the dissimilarity with each neighbour pattern. This measurement denotes the how closer the pattern to be an SDPF, which is considered as the strength of the feature. The rest of the algorithm involves several comparisons among the feature strengths. Handling a vector is computationally more expensive compared to handling a scalar. Therefore, the total strength of a feature point is expressed by a single value. The summation of the D_p values is empirically selected to express the feature strength, which has a lower computational cost compared to the vector magnitude, which involves eight multiplications, eight additions, and one square root operation. The other advantage is that the summation of vector components is more sensitive to individual vector elements than vector magnitude, which ultimately ensures the essentiality of the substantial contribution of all elements to make the feature an SDPF point. The feature strength of a dither pattern in the set S_s is calculated by adding the eight colour distances, which have been measured from the eight neighbour patterns to the current pattern, as shown in (3.5).

$$m = \sum_{p=1}^{8} D_p \tag{3.5}$$

The points in elements of S_s seemed to concentrate on several small regions in images. Therefore, the number of feature points is reduced by suppressing non-maximal features. It should preserve the features with the locally highest strength.

For the non-maximal suppression, a window R_p of a constant dimension is used. The centre of this window is at the point p. Note that all of the points within the window is associated with their strength m. The set of SDPF points S_f is obtained by employing the non-maximal suppression according to the rule defined in (3.6).

$$\begin{aligned} &\forall (p,m) \in S_s: \\ &(R_p \subset S_s \; AND \; \forall (p',m') \in R_p : m \geq m') \rightarrow (p,m) \in S_f \end{aligned} \tag{3.6}$$

The potential feature point and its strength is shown with its association as an ordered pair i.e. (p,m). In order make the definition of the rule clearer, the points taken from the set of potential features are named with *(p,m)* and the points taken withing the non-maximal suppression window are named with *(p',m')*.

The SDPF points in S_f currently associate with the location, set of dither colours and the feature strength only. It has been described numerically to conveniently use in intended applications. A spatial-chromatic histogram is used to describe the distribution of SDPF points.

3.3.2 SDPF descriptor

Since the histogram encodes both the geometric and chromatic information, the SDPF features are required to obtain their spatial and colour information in the reduced dimensional space. The centroid distance function is used for the purpose of encoding the spatial information, whereas a hue colour quantization method is used for the purpose of encoding the colour information.

Centroid Distance Function

Centroid distance function has been classified into one-dimensional shape representation, which cannot be used to reconstruct the original shape. It has been an excellent representation that can be found in several studies, including [90], [91]. The centroid distance function transforms geometric details from two-dimensional space to a single dimensional space. It is invariant to the rotation but not to the scaling [8]. Shape or geometrics of contents in an image can be described by using a boundary based or a region-based method combined with the centroid distance function. In this proposed method, SDPF points are distributed all over the image instead of residing on boundaries. Therefore a region-based method is used, that has been described in [8]. Therefore, the centroid for the set of SDPF in S_f can be calculated using (3.7),

$$C_x = \frac{\sum_{i=1}^{n} P_x(i)}{n}, C_y = \frac{\sum_{i=1}^{n} P_y(i)}{n} \tag{3.7}$$

where, (C_x, C_y) is the coordinates of the centre, $P_x(i)$, and $P_y(i)$ denote the coordinates of i^{th} pattern in S_f , and $n=|S_f|$.

The $r(t)$ denotes the Euclidean distance equation, which can be used to calculate the distance from any of the SDPF point to the centroid (C_x, C_y) as given in (3.8).

$$r(t) = \left([x(t) - C_x]^2 + \left[y(t) - C_y\right]^2\right)^{\frac{1}{2}} \tag{3.8}$$

The centroid distance representation can be considered to have translation invariance as it is normalized by its absolute location (the centroid). It resists to noise and geometrical occlusion and has very low computational complexity [8].

The spatial details of SDPF are encoded by using the centroid distance. The resulting histogram has a separate axis to represents this centroid distance attribute of the set of SDPF points. The range of n^{th} bin at the centroid distance axis of the histogram can be calculated by using (3.9),

$$Bin(n) = \begin{cases} \dfrac{max\left(D_f^2(i)\right)}{k}, & n = 1 \\ Bin(n-1) + \dfrac{max\left(D_f^2(i)\right)}{k}, & 1 < n \leq k \end{cases} \tag{3.9}$$

$$D_f^2(i) = [P_x(i) - C_x]^2 + \left[P_y(i) - C_y\right]^2 \tag{3.10}$$

where, k is the total number of bins in distance axis of the histogram.

It is noteworthy that $D_f^2(i)$ is same as the square of r(t) defined in (3.8) that is purposely used to avoid the square root operation which is computationally expensive. Each of SDPF point in S_f is assigned to one of the k bins after calculating the ranges for the bins. Subsequently, the 2D histogram is built by encoding the colour information of SDPF.

Colour Information Reduction

Each SDPF point contains four colours. It is discovered that HSV colour based histograms are more suitable for expressing semantic information [92]. The colour bin index of any colour is generated using the following mapping function illustrated in (3.11).

$$F_c(x) = \begin{cases} \left\lfloor \dfrac{R(x)}{\dfrac{GRAY_{max}}{12} + 0.5} \right\rfloor, & R(x) = G(x), \\ & G(x) = B(x) \\ 0 & R(x) \geq 2 * G(x), G(x) > B(x) \\ 1 & R(x) \geq G(x), G(x) > B(x) \\ 2 & G(x) > R(x), R(x) \geq B(x) \\ 3 & G(x) > 2 * R(x), R(x) \geq B(x) \\ 4 & G(x) \geq 2 * B(x), B(x) > R(x) \\ 5 & G(x) \geq B(x), B(x) > R(x) \\ 6 & B(x) > G(x), G(x) > R(x) \\ 7 & B(x) \geq R(x), R(x) > G(x) \\ 8 & B(x) \geq 2 * R(x), R(x) > G(x) \\ 9 & B(x) \geq R(x), R(x) > G(x) \\ 10 & R(x) > B(x), B(x) \geq G(x) \\ 11 & R(x) > 2 * B(x), B(x) \geq G(x) \end{cases} \tag{3.11}$$

$F_c(x)$ is the function that calculates the index for colour element x in an SDPF point where $R(x)$ denotes the red channel, $G(x)$ denotes the green channel, and finally $B(x)$ denotes the blue channel. $GRAY_{max}$ is the maximum value in the numerical representation of a colour component in RGB colour space (in this implementation, it is 255). In this function, any combination of RGB values is mapped into 12 indices. The number of indices was determined using an experiment, which will be explained in the next subsection. The analogy of this mapping function to hue quantization is presented in Figure 3.2. This quantization results in reducing a large number of colours in RGB to 12 colours by introducing intensity invariance and removing the colour correlation in RGB components [93]. When the saturation is zero, the function maps the grey values to one of the 12 indices, as illustrated in Figure 3.2. This feature makes the histogram capable of handling both colour and greyscale images. The next subsection explains how the histogram is populated using SDPF points, and the experiment of selecting the optimal number of colour and distance bins.

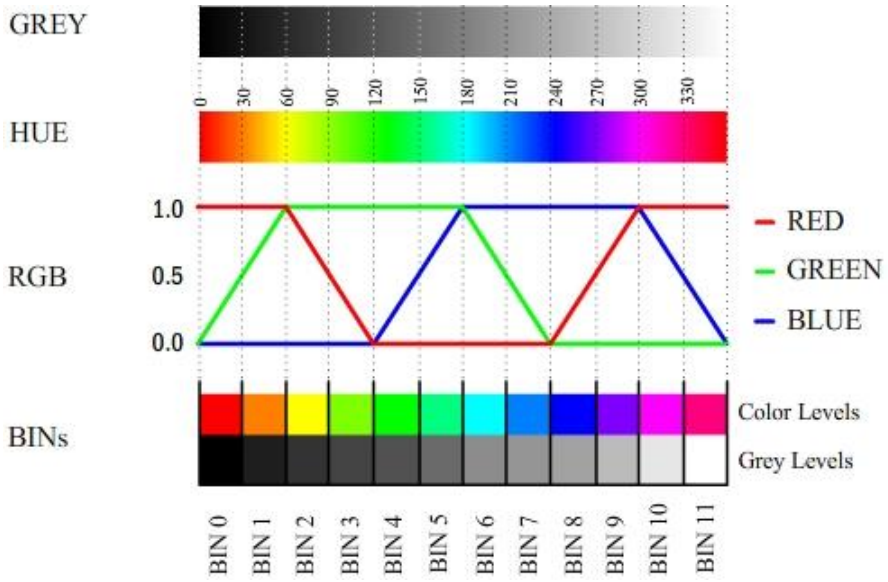


Figure 3.2: The analogy of the function $F_c(x)$ to hue and intensity value quantization.

Populating the SDPF Spatial-Chromatic Histogram

Throughout the previously explained steps of the algorithms, a set of SDPF points, allocated to a set of bins, has been produced based on the centroid distance, as displayed in Figure 3.3. The 2D histogram can be populated with the count of the four blocks in each SDPF against the colour bins and the distance bins calculated in the previous steps.

Figure 3.3 The set of SDPF points allocated to the set of distance bins. The yellow markers show the SDPF points, whereas the green cross marker is the centroid. The coloured bands are the distance bins.

Let B_d be the set of distance bins that are used as one component of the intended 2D histogram and $b_i \in B_d$ ($i=1, 2, 3, \ldots k$) where b_i is the i^{th} bin, which contains the relevant set of SDPF points.

Let $h_{i,j}$ be a bin in the 2D histogram, where i is the coordinate on distance axis, and j is the coordinate of the colour axis. The following algorithm is used to populate the 2D histogram.

```
For i=1 to k
        For each Salient Dither Pattern P in b_i
                For Each Colour Element E in P
                        j = F_c(E)
                        h_i,j = h_i,j+1
                End For
        End For
End For
```

Rescaling the resultant histogram values to a known range, such as 0-1, introduces the scale invariance.

The histogram that is created from the above algorithm is considered as the SDPF descriptor. This descriptor is then used as the input of any classification method to classify an image or video frame to high-level objects. In this study, Support Vector Machine (SVM) is mostly explored as the classification tool of the proposed descriptor. The optimal value of k (the number of distance bins) and the best number of colour bins were found by conducting an experiment. The average classification rate of images from Corel-1000 and Caltech databases was obtained for different combinations of values for k and the number of colours. An empirical range of 3-10 was set to the k in order to limit the exploration. 6, 12 and 24 number of colours were used in the colour axis to explore the best combination. About 10 visual concept categories and 40% of images from each category have been utilized for the training (C-SVM), and the remaining were used to test the model. According to Figure 3.4, the

maximum correct classification rate is found when the number of distance bins is equal to 4 and the number of colour bins is equal to 12.

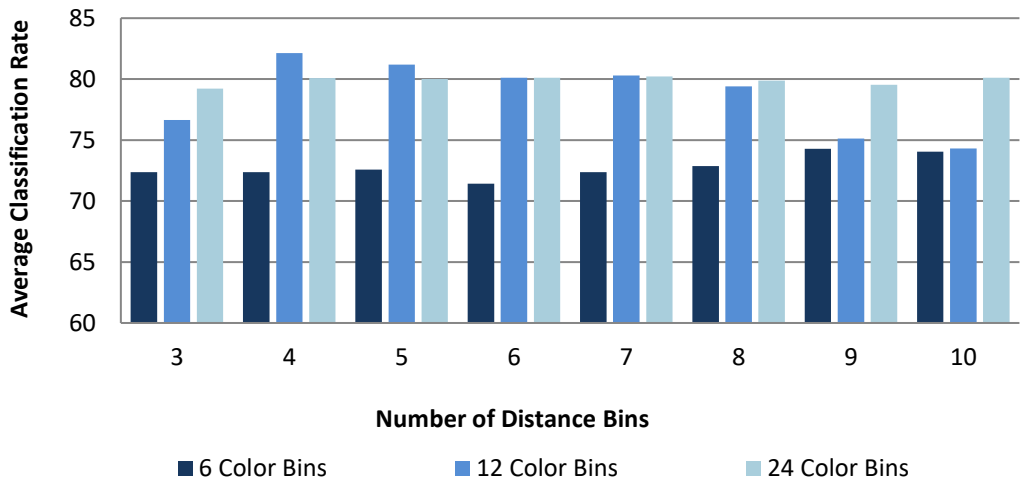


Figure 3.4: The experimental results to find the optimal dimension for SDPF descriptor.

Theoretically, the spatial-chromatic histogram of SDPF is robust to any of the common geometric transformations, due to the inherited properties from the centroid distance measurement and the normalized distance bin selection method. However, the repeatability of SDPF features over rotational and scale transformation impacts the overall geometrical invariance; as a result, it was assessed by a well-formed experiment, and the results have been presented and discussed in Chapter 7.

Although the SDPF descriptor has demonstrated the lowest dimensionality, a competitive robustness to rotation and scale variances and a competitive advantage in computational time over the state-of-art, the experimental results have shown that it was less behind the classification accuracy of the existing best performed visual descriptors for some of the visual concepts, which have been further discussed in Chapter 7. Therefore, an innovative descriptor, namely, Dither Density Descriptor (DDD), is introduced and fused with SDPF descriptor to improve the classification accuracy without sacrificing its beneficial properties.

3.4 Dither Density Descriptor (DDD)

Dithering can make the human eye see more colours by varying the spatial arrangement of a smaller number of colours. The SDPF included the spatial distribution of the dithered colour in its descriptor. However, as the experiment results directed, the dithered colour itself does not provide the details of the absent colours, but the density of colour can quantify the specific shade of that colour. Inspired by this phenomenon, many virtual colour levels could be encoded out of a small number of actual colour levels using a pair of values, the colour level, and its spatial density. The dithered image in Figure 3.5 contains only two actual colours, i.e., black, and white, but when considering the spatial regions in the image, it shows many virtual colours that can be represented with the actual colour and its pixel density over the selected region. This concept allows a simple population count algorithm to represent the local colour distribution in an extremely reduced dimensional colour space for a given local region of visual content.

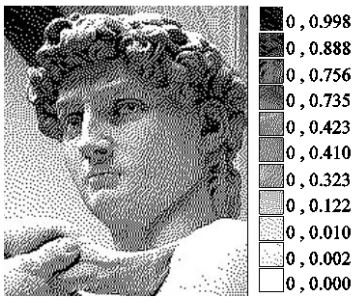


Figure 3.5: Coding grayscale with dither density utilizing a single bit per pixel. The spatial density is calculated using square-shaped regions.

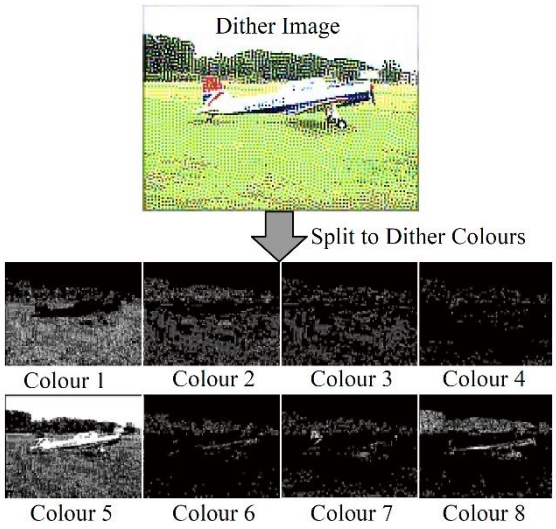


Figure 3.6: Splitting the dither image to several binary images based on the dither colours.

The Dither Density Description (DDD) algorithm has three main steps, i.e., applying to dither, creating binary images by splitting the dithered image to the selected colours as displayed in Figure 3.6 and finally applying population count by counting nonzero pixels for each of the binary images that correspond to different colours in the selected dither colour gamut. In this study, a fast error diffusion (E-D) dithering technique has been adopted, which is an improved version of Floyd–Steinberg dithering described in [94]. In order to preserve the rotation, scale, and translation invariance properties of the fused SDPF-DDD descriptor, the density calculation is carried out with a fixed number of circular-shaped regions centred at the centroid of SDPF feature points. The dither colour density can be calculated for a given region by (3.12),

$$D_c(r) = \frac{popcount(r)}{area(r)} \tag{3.12}$$

where, r is a region $D_c(r)$ is the density of the dither colour c, *popcount* denotes the population count of r, and *area* is the geometric area of the r.

The number of circular regions and dither colour gamut is selected by assessing the classification rate for many different possible combinations. The colours are selected from the outer most boundary of the RGB colour space, where both brightest and darkest colours are found.

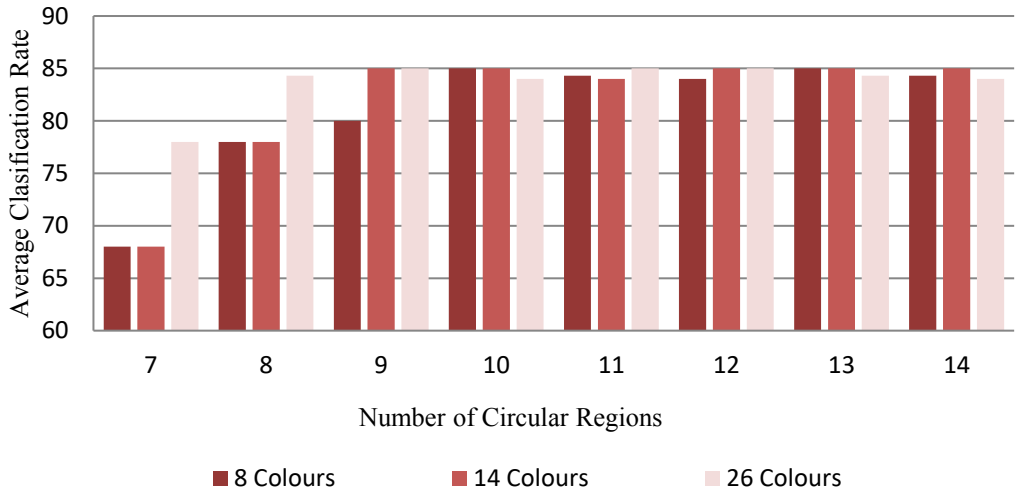


Figure 3.7: The experimental results of finding the optimal number of colours and circular regions.

The assessment results of the selection of an optimal number of regions for dither density calculation and the number of colours in the dither colour gamut are presented in Figure 3.7, which suggest that the average classification rate is saturated after the number of circular regions becomes 9, and the number of colours becomes 14. However, the number of circular regions has been set to 10 and the number of colours to 8 as the dimensionality equals 80, and it indicates the saturated average classification rate.

The density values of 8 colours in 10 circular regions are concatenated to the SDPF descriptor and then normalized for the purpose of constructing the final SDPF-DDD descriptor.

Although SDPF-DDD has shown better classification performance with a smaller number of object types, further experiments have suggested that it lags behind the state-of-the-art, when the number of object types get extremely higher. Consequently, it was further improved by stabilizing the key points and incorporating more details to the final spatial-chromatic descriptor.

3.5 Improved Hessian based salient dither pattern feature (HSDPF)

The proposed improvement of the third version of the SDPF comprises of the following sub processes in its extraction and description procedure.

1. Quantize colours
2. Calculate the determinant of Hessian
3. Detect salient dither patterns by analysing the Hessian response
4. Suppress the non-maximal

Some critical observations on the failures of the SDPF revealed that the linear colour quantization, which has been used in the original implementation, could cause to lose essential details in image regions where the colour contrast is very low. Therefore, the quantization step has been improved by employing an error diffusion (ED) colour dithering method that is proposed in [94]. Unlike in the generic colour quantization approach, the ED-dithering technique generates various patterns consisting of different colours for approximating the original colour. Figure 3.8 illustrates two regions with slightly different colours precisely (153,255,153) and (171,255,119) in RGB colour space. Figure 3.8(b) shows that simple linear quantization with 12 colour levels cannot distinguish between the two colours. However, the patterns shown in Figure 3.8(c), that have been generated from the ED dithering technique preserves the contrast between the two colours. The comparison of Figure 3.8(b) and Figure 3.8(d) demonstrate that the technique helps to preserve the colour contrast, which could be lost in generic colour quantization techniques. The actual colour contrast is preserved by means of different colour patterns.

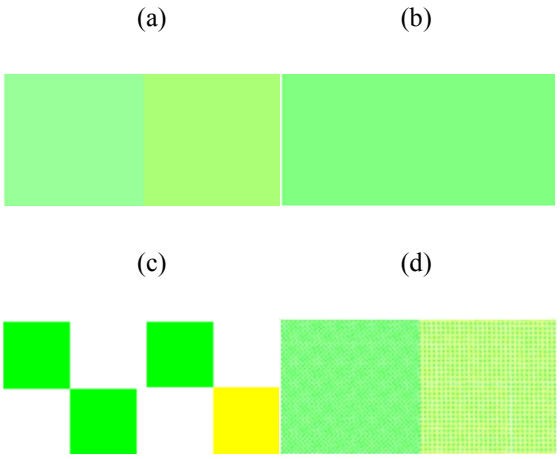


Figure 3.8: Improved ED-Dithering algorithm preserves the colour contrast. (a) two regions with slightly different colours (b) Linearly quantized (c) Enlarged dither patterns (d) ED colour dithering-based quantization.

The improved ED Dithering algorithm is derived from the well-known Floyd-Steinberg error diffusion dithering algorithm. The core improvement relies on a set of optimal ED coefficients, which can be identified offline. The pre-calculated optimal coefficient can be selected based on the input signal. The optimization has been done by varying the coefficient until the difference between the Fourier spectrum of a set of input signals its blue noise is minimized [94]. It has been displayed that the optimization results in fewer dithering artifact in the dithered image. Moreover, this improved ED-Dithering is computationally more efficient compared to the generic implementation of Floyd-Steinberg dithering technique. The study in [94] suggests that the computational efficiency is a result of utilizing lesser memory accesses and arithmetic operations in the whole algorithm. The first step of this improved dithering process the colour quantization which is carried out via (3.12),

$$N'_{00} = C_i \; ; \; C_i \in C_d, C_i \sim N_{00} \tag{3.13}$$

where, N_{00} is referring to the original colour of a pixel, C_d is the set of all the colours in the quantized space, and N'_{00} is the colour assigned by the dithering process. $C_i \sim N_{00}$ means that the new colour should be selected as it is the numerical closest to the original colour. The different between the new and old colour will be calculated for the diffusion, by using (3.14),

$$E = N_{00} - N'_{00} \tag{3.14}$$

where E is the error of quantizing the colour. Finally, the error will be distributed to a set of neighbours weighted as illustrated in (3.15);

$$\begin{aligned} N_{10} &= D_{10}(N_{00}) \times E \\ N_{01} &= D_{01}(N_{00}) \times E \\ N_{-11} &= D_{-11}(N_{00}) \times E \end{aligned} \tag{3.15}$$

where N_{10} is the colour of the next pixel, N_{01} is the colour of the pixel at the same column but in the next row and N_{-11} refers to the colour of pixel at the previous column in next row. D_{10}, D_{01}, and D_{-11} are the respective error diffusion coefficient vectors of N_{10}, N_{01} and N_{-11}. These error diffusion coefficient vectors can be calculated offline,

and the pre-calculated table of values can be found in APPENDIX A, which has been initially calculated in [94].

This improved version of the SDPF use the colours at the end points of the colour cube in RGB space as shown in Figure 3.9. The effect of different numbers of dither colours for the classification performance was studied, and will be presented in the next subsection, which describes the optimization of the dimensionality. This specific method of colour selection makes it easy and speedup the searching process of the nearest dither colour. The Euclidean distance-based conventional method to retrieve the closest colour requires 24 multiplications, 16 additions, and seven logical comparisons per pixel. However, the proposed approach requires only three logical comparisons per pixel at any given time. The binary search tree of colours is given in Figure 3.10. The numbers which range from 1 to 8 in Figure 3.10 denote the indices used for representing the selected dither colours, whereas the values of R_h, B_h, and G_h are shown in Figure 3.9.

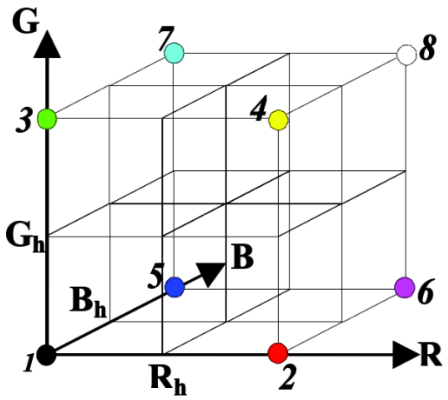


Figure 3.9: Dither colour set in RGB space

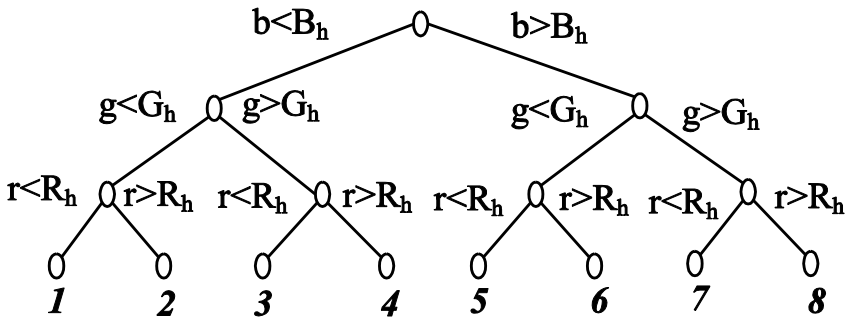


Figure 3.10: Binary search tree of dither colours

The specific arrangement of the colours in the four blocks of dither colour unit, does not make the illusion of different colours as shown in Figure 3.11. Consequently, the permutations can be safely removed by sorting the colour indices in each of the patterns.

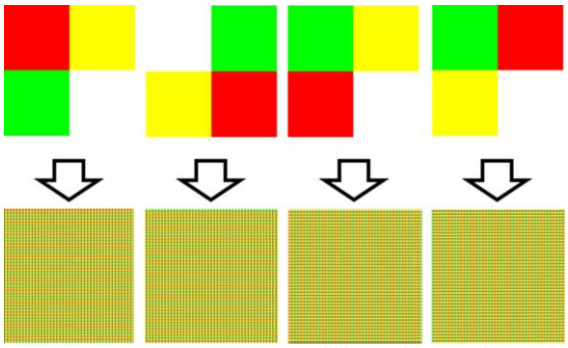


Figure 3.11: Different permutations of the same set of colours with the resultant overall colours.

An SDPF is defined as a pattern with four coloured regions, that is significantly different from its neighbouring patterns. However, due to the tremendous visual difference, the dissimilarity of two colours in the dither colour space cannot be precisely quantified, other than specifying either similar or dissimilar. Therefore, the difference between two dither patterns is obtained by using (3.16),

$$d(P_{i,j}, P_{k,l}) = \sum_{c=0}^{3} w_c \tag{3.16}$$

$$w_c = \begin{cases} 1 & P_{ij}(c) \neq P_{k,l}(c) \\ 0 & Otherwise \end{cases} \tag{3.17}$$

where $d(P_{i,j},P_{k,l})$ quantitatively denotes how the two patterns $P_{i,j}$ and $P_{k,l}$ $P_{i,j}$ are different. c denotes the index of the colour out of the four colours in a single pattern where $P_{x,y}(c)$ denotes the colour corresponds to the index in pattern $P_{x,y}$. Moreover, w_c only check whether the two colours from corresponding locations in a pair of dither patterns are similar (0-unit distance) or dissimilar (1-unit distance).

Studies have identified that extreme points in a distribution can be detected through the analysis of the Hessian matrix [40]. However, no existing method focuses on detecting critical points in the distribution of local colour patterns. Therefore, a novel technique, which symbolizes the general Hessian based critical point analysis

approach, has been introduced. Hessian matrix analysis requires to obtain the second order derivation of the distribution at the given points. However, there is no existing method to calculate the second order derivative of a distribution of dither patterns. Therefore, an approximation is used to obtain it. Then the determinant of the hessian matrix is calculated for the purpose of finding any extrema in the distribution. The calculation of the Hessian matrix with the approximated second order derivation is given in (3.18) and (3.19),

$$H(i,j) = \begin{bmatrix} L_{xx}(i,j) & L_{xy}(i,j) \\ L_{xy}(i,j) & L_{yy}(i,j) \end{bmatrix} \tag{3.18}$$

$$L_{xx}(i,j) = d\big(P_{i-1,j}, P_{i,j}\big) + d\big(P_{i,j}, P_{i+1,j}\big)$$

$$L_{yy}(i,j) = d\big(P_{i,j-1}, P_{i,j}\big) + d(P_{i,j}, P_{i,j+1})$$

$$L_{xy}(i,j) = \frac{1}{4}\Big(d\big(P_{i-1,j-1}, P_{i-1,j+1}\big) + d\big(P_{i+1,j-1}, P_{i+1,j+1}\big) + d\big(P_{i-1,j-1}, P_{i+1,j+1}\big) + d\big(P_{i-1,j+1}, P_{i+1,j+1}\big)\Big)$$

$$det(H(i,j)) = L_{xx}(i,j) \times L_{yy}(i,j) - L_{xy}(i,j)^2 \tag{3.19}$$

The possible SDPF points are nominated by thresholding the determinant of the Hessians, obtained using the dither patterns; the procedure which should produce the steadiest critical point that theoretically produce the most consistent and repeatable SDPF point. The prospective SDPF points are chosen according to the condition given in (3.20).

$$D(i,j) > T(i,j) \rightarrow \Big(P_{i,j}, D(i,j)\Big) \in S_s \tag{3.20}$$

where,

$$D(i,j) = \ abs\ \big(det(H(i,j))\big)$$

S_S includes all the prospective SDPFs. The threshold *T(j,j)* is calculated according to (3.21).

$$\begin{aligned} T(i,j) = {} & d\left(P_{i-1,j-1}, P_{i,j-1}\right) + d\left(P_{i,j-1}, P_{i+1,j-1}\right) + d\left(P_{i+1,j-1}, P_{i+1,j}\right) \\ & + d\left(P_{i+1,j}, P_{i+1,j+1}\right) + d\left(P_{i+1,j+1}, P_{i,j+1}\right) \\ & + d\left(P_{i,j+1}, P_{i-1,j+1}\right) + d\left(P_{i-1,j+1}, P_{i-1,j}\right) \\ & + d\left(P_{i-1,j}, P_{i-1,j-1}\right) \end{aligned} \tag{3.21}$$

The pattern that is in question is surrounded by eight other patterns. In this context, the significance can be defined as that the how different the centre pattern from the total difference of surrounding patterns relative to one another. The equation (3.21) denotes the summation of the differences between surrounding adjacent patterns and using this value as the threshold yields better classification accuracy. It was observed that this version of SDPF (HSDPF) also needed to be undergone a non-maximal suppression process similar to the original version of SDPF. The same rule specified in (3.6) is applied to this version. However, since the feature strength is calculated in different way in this version, the same rule is represented with the relevant terms as given in (3.22).

$$\begin{aligned} & \forall \left(P_{i,j}, D(i,j)\right) \in S_s \\ & \left(R_p \subset S_s \; AND \; \forall \left(P_{i',j'}, D(i',j')\right) \in R_p : D(i,j) \geq D(i',j') \rightarrow \left(P_{i,j}, D(i,j)\right) \right. \\ & \qquad \left. \in S_f \right) \end{aligned} \tag{3.22}$$

where, R_p is a window which is a subset of the potential HSDPF for non-maximal consideration, centred at the pattern $P_{i,j}$, whose strength is given by the determinant *D(i,j)*. S_f is the resultant set that includes HSDPF points after non-maximal suppression.

A 3D spatial-chromatic histogram is used to describe the extracted HSDPF points. The histogram contains both the geometric and chromatic details.

3.5.1 HSDPF descriptor

The details of the colours in a pattern are referred from HSDPF point. The colours in the patterns are already in an extremely lower-dimensional space compared to the original RGB colour space. Particularly the number of colours available is selected to be 8. However, the spatial distribution of different combinations of dither colours further gives more salient clues about the original colours in different regions of the image. The spatial information is extracted, considering two attributes of an HSDPF point, namely.

1. The centroid distance
2. The angle between the point and the dominant direction of the distribution of HSDPF points

The details of centroid distance measurement and the method of binning the distances can be found in the previous subsection, which describes the original implementation of SDPF. However, in this improvement, an additional step, i.e., the calculating angles of points with orientation normalization, has been applied to encode more details into the spatial chromatic histogram to increase the discriminative ability.

Orientation Normalization

The centroid distance encodes only single-dimensional spatial property, which cannot be used to reconstruct the original distribution of the points. Encoding both the centroid distances with the angle that the point creates a reference direction can fill the lacking details in the previous descriptor. Therefore, the angles of SDPF points are considered, while constructing the improved descriptor. This additional detail in the final spatial-chromatic histogram should make the final descriptor more discriminative. However, it is challenging to find a consistent reference direction to obtain the angles, regardless of the orientation of the object. It is a crucial requirement, as it decides the rotational invariance of the descriptor. Hence, an orientation normalization technique has been proposed. In this proposal, the angles between the HSDPF points and the dominant direction of the HSDPF distribution are calculated. Since the SDPF points are extracted with a stable technique over different orientations, the dominant angle should

not significantly differ in case of an orientation difference. The dominant orientation is calculated by identifying the best-fitted line for the SDPF points using the least square method (3.23),

$$m = \frac{\sum_{n=1}^{N}(x(n) - x_c)(y(n) - y_c)}{\sum_{n=1}^{N}(x(n) - x_c)^2} \tag{3.23}$$

where *m* denotes the gradient. However, this gradient cannot discriminate between right and upside-down (turned by 180^0) poses of the object. Therefore, the difference in the densities of SDPF points is considered to calculate the reference angle. First, the object is divided into two sections as upper and lower halves over the dominant direction calculate using (3.23). The boarder of the two section should lie on the centroid of the HSDPF point distribution. Then the HSDPF point density of each of the sides is calculated. Finally, whether the object is upright, or upside down is decided based on the density difference, to calculate the reference angle. The perpendicular line can be obtained via (3.24).

$$y = \frac{-1}{m}(x - x_c) + y_c \tag{3.24}$$

HSDPF points can be grouped into one of the sections via (3.25),

$$P_{side}(n) = \begin{cases} 0 & l(n) = 0 \\ 1 & l(n) > 0 \\ 2 & l(n) < 0 \end{cases} \tag{3.25}$$

$$l(n) = y(n) + \frac{1}{m}(x(n) - x_c) - y_c \tag{3.26}$$

where $P_{side}(n)$ is an HSDPF point, with the location coordinates *x(n)*, and *y(n)*. The initial orientation can be obtained using (3.26),

$$\theta_0 = \begin{cases} \tan^{-1} m & |U_1| > |U_2| \\ \tan^{-1} m - 180^0 & |U_1| < |U_2| \end{cases} \tag{3.27}$$

where U_1 and U_2 denote the HSDPF points from the two sections as two sets. If the value of θ_0 is higher than 360, it will be revolved clockwise by 360^0. Similarly, if it

has a negative value then it must be rotated counterclockwise by a full round. The angle of HSDPF point is calculated using (3.28),

$$P_\theta(n) = \tan^{-1}\left(\frac{y(n) - y_c}{x(n) - x_c}\right) - \theta_0 \tag{3.28}$$

where, $P_\theta(n)$ denotes the angle of n^{th} HSDPF point created with the centroid with orientation normalization by subtracting θ_0. If $P_\theta(n)$ is greater than 360, it will be revolved clockwise by 360^0 degrees. If its value is negative, then it should be rotated counter clockwise by subtracting 360^0 degrees. After obtaining the angles of all HSDPF points the angle bins (B_a) are populated with the normalized angles that are calculated for SDPF points. The best number of angle bins (k_a) was selected by experimentation carried out by assigning several different values. The experiment carried out for selecting the optimal value for k_a is explained in the next subsection. The angle axis of the histogram is populated using (3.29),

$$B_a(n) = \left\lfloor \left(\tan^{-1}\left(\frac{y(n) - y_c}{x(n) - x_c}\right) - \theta_0 \right) / R_a \right\rfloor \tag{3.29}$$

$$R_a = 360/k_a \tag{3.30}$$

Figure 3.12 shows sample instances of an object that are output from different steps in the SDPF algorithm. Figure 3.12(b) shows that the object resembles the original colours even after been quantized to eight different colours using the ED dithering. Figure 3.12(c) shows that the SDPF point cloud abstracts the shape and spatial structure of the object. After extracting all necessary details, such as the dominant orientation as visualized in Figure 3.12(d), the normalized angles, and the dither colour indices of all the extracted SDPF points, the SDPF spatial-chromatic histogram descriptor can be built as described in the next section.

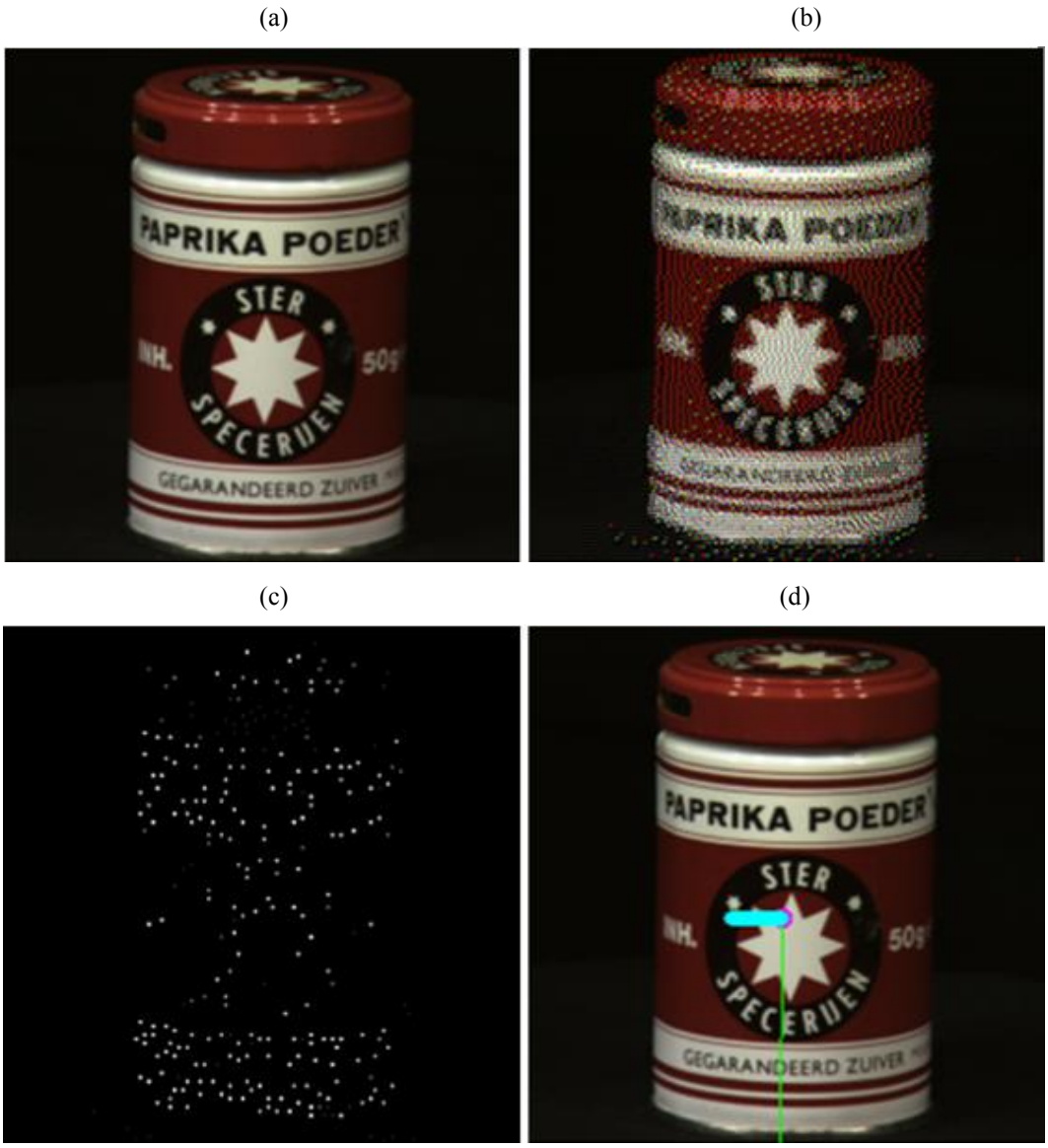


Figure 3.12: Different instances of an object in the SDPF algorithm. (a) the original image, (b) dithered image (c) extracted SDPF points, (d) calculated dominant orientation.

Construct HSDPF Descriptor

Each of the HSDPF points associate with four dither colours, the centroid distance, and the angle. Each property can be represented with three different bins B_d, B_a and B_c as given in (3.31) and (3.32),

$$P_{(x,y)} = \{B_d, B_a, B_c\} \tag{3.31}$$

$$B_c = \{c_1, c_2, c_3, c_4\} \tag{3.32}$$

where, $c_1 ... c_4$ denotes the four colours of the dither pattern. A three-dimension histogram is used with the set of three bins to describe the distribution of HSDPF points.

The histogram consists of the following axes.

1. Chromatic axis
2. Distance axis (to keep the distribution of centroid distances)
3. Angle axis (to keep the distribution of orientation-normalized angles)

The ranges for the distance bins B_d and the orientation bins $B_{a,}$ can be calculated using (3.9) and (3.29) respectively. However, as each pattern consists of four colours, a single pattern contributes exactly four times to one or several chromatic bins, as mentioned in (3.32). When the descriptor is completed by populating the histogram, it is then normalized to a known range; 0-1, which will omit the descriptor's dependency on the resolution of the image.

Dimensionality Optimization of the Histogram

A support vector machine (SVM) is used with the final output descriptor of the SDPF algorithm to classify the images to one of the object categories. The optimal values for the dimension of distance axis k_d, the dimension of orientation axis $k_{a,}$ and the dimension of chromatic axis k_c are explored through an experiment, which uses Caltech dataset [88], which consists of images with a single object. The metric used for the evaluation is the average classification rate. All the combination of values of the bins were explored with the following ranges,

1. k_d ranges from 3 to 10
2. k_a with the amounts of 4,8,12 and 15,
3. k_c with 6, 8, 14, and 26 of discrete values.

The ten class labels provided in the Corel-1000 dataset were used for the experiment. The train split of the data consists of 40% of total data, whereas the remaining 60% is used to test the model. The SVM is constructed as a C-SVM model. The classification rate is obtained for the test split using the trained SVM model. The Caltech dataset was

not used in any of the other experiments to assess the consistency of this bin selection. Figure 3.13 presents the average classification rates obtained for all combinations of different k_d, k_a, and k_c.

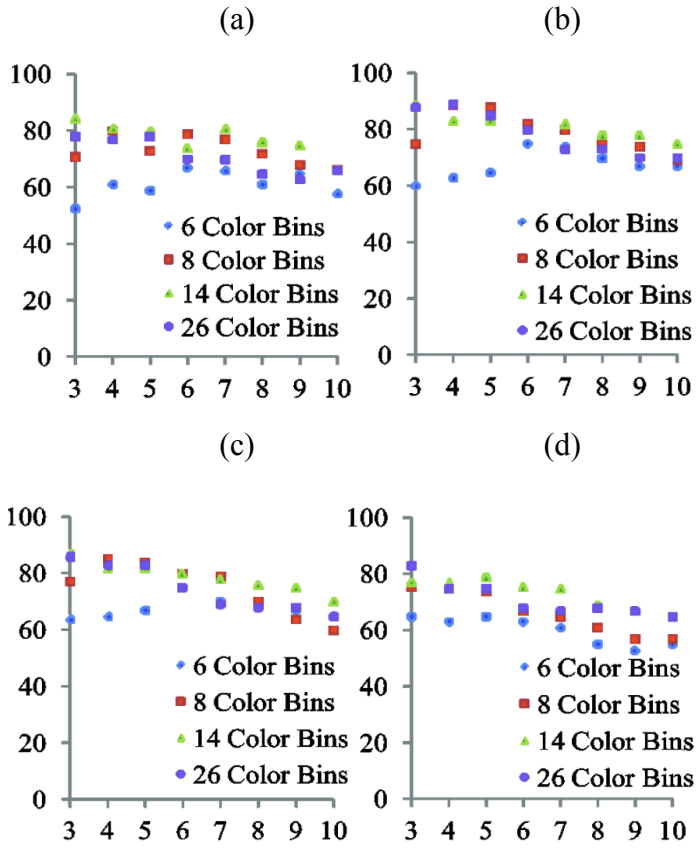


Figure 3.13: The accuracy vs. the dimension over distance axis k_d and chromatic axis k_c. (a) k_a = 4, (b) k_a = 8, (c) k_a = 12 and (d) k_a = 15.

It can be observed that the experiment shows the best classification rate, which is 89%. The best value is found for the configuration of k_d=4, k_a,=8, and k_c=26. However, there are few other options that give the classification rate of 85.5% for the configurations of k_d=3, k_a,=8, k_c=14 and k_d=4, k_a,=8, k_c=8. It can be observed that the dimensionality is 832 when the classification rate indicates its highest value. However, when the classification rate is the second best, that is 85.5%, the dimensionality of the descriptor is greatly reduced; at one instance it is 336, and in the other, 256. The options remaining for the selection are 89%/832, 85.5%/336, and 85.5%/256. This study focuses more on balancing both classification rate and the dimensionality that is crucial in the cost of computing; the instance of k_d=4, k_a,=8, and k_c=8 in which the overall

dimensionality is 256, have been selected. All further experiments were carried out by keeping the dimension of the descriptor at these optimal values.

3.5.2 Classification of SDPF descriptor

Initially, although the nearest neighbour classification was used to classify this improved SDPF descriptor, the results were unacceptable. The experimental results are shown in (APPENDIX C). Therefore, SVM is used by setting its kernel to a cubic polynomial. The kernel trick used in the SVM makes the training phase faster, compared to the techniques that search for a precise mapping function between the descriptor's space to a higher dimensional linearly separable space [95]. Further, the use of SVM enables its training phase to be conducted on low-end devices. This thesis reports the performance of the HSDPF by classifying with SVM on different hardware platforms to measure the robustness of the techniques over both high-end and low-end computer platforms.

3.6 Summary

This chapter elaborates on three significant versions of the proposed low-dimensional feature descriptor, which can be used for object recognition. Initially, it has explained the way of using an approximation for colour dithering as a dimensionality reduction technique. Then it has expressed how this nature-inspired dimensionality reduction technique evolved into an efficient feature descriptor that encodes both colour and spatial information. Moreover, this chapter has elaborated how the initial version of the proposed work was improved by first introducing additional dither pattern density information, and then stabilizing the feature points obtained by employing a fast-dithering technique combined with a novel colour-based hessian matric analysis. The next chapter will explain the proposed object segmentation technique with its scientific background.

CHAPTER 4

OBJECTS SEGMENTATION

This chapter introduce a method for segmenting video frames based on spatial, spectral, and temporal properties of a video, by clustering the visual feature proposed in Chapter 3. The proposed method has been improved over two major versions, and both have been described in this chapter.

4.1 Overview of the proposed methods to segmentation

The proposed method of segmentation primarily focuses on clustering SDPF feature points. The segments are expected to be meaningful, and both syntactically and semantically correct. The proposed method utilizes three main elementary features, namely:

1. Temporal details that have been extracted by the optical flow estimation technique.
2. SDPF colour patterns
3. Spatial properties

Initially, the clustering is applied using K-means algorithm directly on the spatial, spectral, and temporal attributes, and then the frame is segmented to an estimated number of objects. However, the distribution of errors in the estimation of the number of objects have demonstrated a significant variation. This inconsistent error affects the entire process of the segmentation due to the error propagation up to the clustering process. This problem has been addressed by another proposal. This improved version uses a low-level segmentation technique called “superpixel” combined with a segment refining process. The refining process gradually merges the segments that are resultant from the superpixels using specific merging criteria with a tuning parameter. This refinement process makes it feasible to overcome the challenge of estimating the number of objects dominated in a set of frames in a video. The benefits and limitations of the two versions have been evaluated comparing with each other and applying different state-of-the-art clustering algorithms. The results have been reported and discussed in Chapter 7.

4.2 K-means based feature clustering for segmentation

The proposed method relies on several spatial, spectral, and temporal properties and the K-means clustering algorithm. The first step of the proposed approach is to select the properties of SDPF to consider in clustering. The K in the K-means algorithm corresponds to how many dominant objects appeared in a frame. Since the K-means clustering algorithm requires a value for the K prior to the using over the set of data points, an estimation of the number of the objects, based on the number of SDPF, has been proposed. The next section explains the selection of the properties of SDPF.

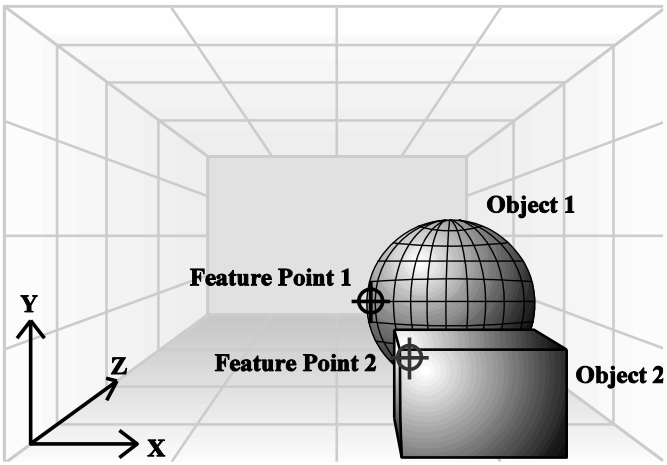
4.2.1 Selection of properties

One of the potential properties which can be used for the clustering is the spatial density property of SDPF points. However, it is not sufficient in case of overlapping objects when projected to 2D space from the 3D space. The present study does not focus on specialized hardware devices, such as cameras, that can perceive depth. Therefore, if two or more objects appear to share the same density of SDPF points, it may result in under-segmentation. The proposed method utilizes several more properties of the SDPF, including the chromatic component, and the temporal properties. SDPF point consists of four dithered colours in addition to its spatial coordinates. Further details of SDPF can be found in Chapter 3 (previous chapter). The proposed method utilizes these dither colours and spatial property of SDPF for the segmentation by using K-means clustering.

The incorporation of the chromatic property partially overcomes the problem of overlapped objects. However, it further introducing an adverse side-effect that over-segments the SDPF points, causing regions with extremely different colours within a single object. This is true for any complex object, including the ones that are made with multiple types of materials. In the proposal, this adverse consequence is mitigated with help of the motion properties of SDPF points. The moving objects have a higher probability of motion with different velocities; hence the motion details can be used to accurately discriminate the regions of different objects. If the objects are static, but the camera is moving, then again, the objects in different depths will have different relative

motion details, as illustrated in Figure 4.1. In the second case, the motion details still help to discriminate different objects accurately.

(a)



(b)

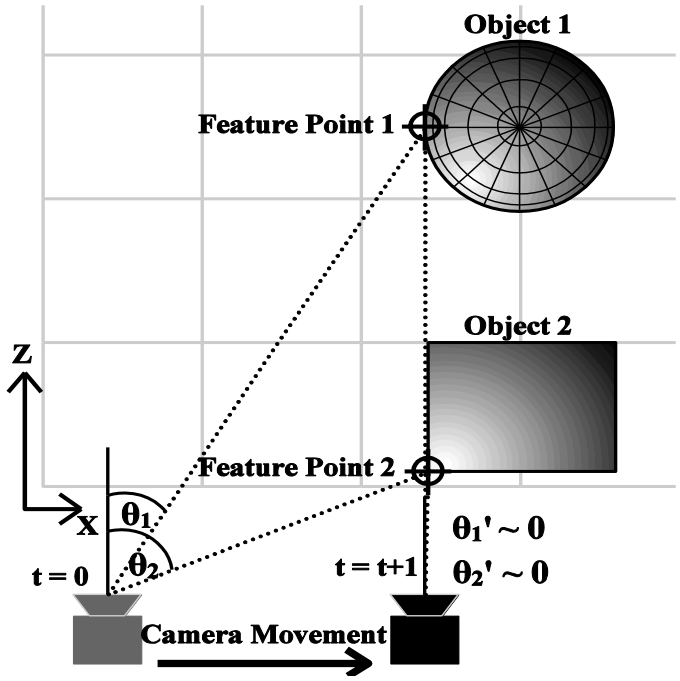


Figure 4.1: Relative motion difference of feature points detected at different depth with a moving camera. (a) front view (b) top view

Figure 4.1 shows a scenario where two objects at different depths appear overlapped from the front view. Figure 4.1(a) shows the front view of the scene which is the only available perspective in a video frame where Figure 4.1(b) shows the actual scene from the top. There are two feature points that are referring to corners of each object, shown in the top view. The two key points perceived with the θ_1 and θ_2 angles refer to the standard camera viewing plane when the time $t=0$. The two camera positions at $t=0$ and $t=t+1$, and two feature points on the two objects collectively resemble two triangles that share a base but having the diagonals and the remaining bases with different length. This geometrical difference clearly ascertains that $\theta_2 > \theta_1$. With the movement of the camera from $t=0$ to $t=t+1$, both the angles θ_1 and θ_2 gradually become zero. Since the angle differences created by the two feature points are different within the same time duration, it can be concluded that the angular velocities of the two points are also different. The projection of the feature point in three-dimensional space to the two-dimensional x-y plane that correspond to the front view depends entirely on the two angles. Therefore, it can be further observed that the linear velocities of two feature points on the 2D plane of the front view are also different. This difference can be utilized to discriminate objects that are positioned at different depth regardless of the overlapped projection to the front view. Furthermore, it is noteworthy that this method may work even with non-moving objects if the camera is moving.

This motion detail can be obtained if the point correspondence can be obtained at different consecutive frames. In the proposed method, the optical flow is obtained to detect the flow of feature points. The SDPF points are used for obtaining the optical flow by following a flow detection method named as Lucas-Kanade (LK). After capturing the apparent motion, the direction and magnitude of the motion, are calculated.

4.2.2 Attributes of the data for clustering

The data is prepared through the following steps to employ the clustering algorithm.

1. Extracting SDPF points from the current frame.
2. Obtain the flow of the points using the remaining consecutive frames.

3. If the next frame is unable to find at least 50% of the SDPF points tracked since the first frame, set the next frame as the current frame, and continue from the first step.
4. Using the feature point in the current frame and tracking it in the next frame to calculate the flow angle and motion magnitude.

In step 3, it is necessary to check whether at least 50% of the features can be tracked in the next frame because, lack of tracking clues a shot boundary in the video, which consists of extremely different consecutive frames, where different objects have appeared. In step 4, the flow angle and the magnitude are calculated. However, the angles calculated by using the points that are tracked by the default tracking algorithm of LK optical flow result in inconsistent values when the magnitude is extremely low. Hence, instead of directly using the flow angle, first, it is scaled with the magnitude of the flow, which helps remove the effect of noisy angles by getting less contribution from them to the final clustering result.

After employing the above algorithm, each of the SDPF points gets a vector that has the following elements.

1. x coordinate
2. y coordinate
3. direction of motion
4. velocity of motion
5. chromatic information in SDPF

The vector is then clustered in 8-dimensional space, as the vector consists of eight elements. Since the K-means clustering algorithm is employed, a value for the K should be given to define the initial cluster points. In this application, K denotes the number of dominant objects in the consecutive frames. In this study, it is hypothesized that the number of objects is in a consistent relationship with the number of SDPF points extracted from a frame. The method proposed for estimate the number of objects is described in the next sub section.

4.2.3 Estimation of K in K-means

The number of SDPF points can be used as a good approximation for the K as if there are multiple objects then there must be multiple disturbing dither patterns which ultimately increase the number of SDPF points detected. The first step involves in finding the exact number of dominant objects from 50 randomly selected frames. CamVid dataset [96], which consists of data of an over 10 minutes long video of high-definition resolution with 30 frames per second of temporal resolution. These videos have been captured using a camera mounted on a moving vehicle. The number of SDPF points in each of the frames is recorded with the number of dominant objects. The data collected from the dataset is presented in APPENDIX D. A curve, which represents a model that can be used to approximate the number of objects when the number of SDPF (n) is given, was fitted to the data recorded for the dataset. The number of objects can also be interpreted as the number of clusters, which is the value of K. The equation for the curve has been obtained and presented in (4.1). This equation is then used to estimate the K in all the experiments.

$$K = 3.117 \ln(n) - 13.7 \tag{4.1}$$

The equation vastly relies on the dataset that is used for obtaining the data to which the curve is fitted. The reason for this dependency the number of objects versus the number of features extracted from a frame highly reliant on the relative scale of the object that appeared in the image, the quantity of different coloured subcomponents of the object, and the resolution of the image. The data dependency of the estimation limits the usage of this technique, where the annotated samples are not available for the domain of an application. Further, the annotation of data manually is a tedious task that consumes an enormous amount of time. Furthermore, the manual annotation is also very subjective; hence the consistent performance from the technique cannot be expected. These limitations inspire finding another method without data or domain dependency. The second method is explained in the following subsection.

4.3 Segmentation of video frames to an unknown number of objects

The necessary outcome of this proposed method is to improve the previously explained static and moving objects segmentation approach, using a fully unsupervised visual feature clustering technique for videos with camera motion. The proposed improvement is based on short term optical flow details of SDPF feature points and appearance-based cues. The approach includes a refinement process for the set of feature points to estimate the regions of high-level objects by using colour cues and the differences in local motion trajectories from a set of estimated low-level segments. Superpixel is one of the popular techniques implementable to image segmentation [97], [98]. The comparison presented in [99] shows that Simple Linear Iterative Clustering (SLIC) superpixel is the best performing superpixel method in terms of efficient memory consumption, low time consumption, and the accuracy of segmentation. SLIC algorithm is used in the proposed method to obtain a set of primary spatial segments. The SLIC is applied to images after converting them into Lab colour space. The Lab colour components are weighted before applying the SLIC for the purpose of reducing the effect from differently illuminated regions. The SDPF has been used for key-point extraction due to the invariant properties mentioned in Chapter 3. The optical flow of the key-points has been obtained using the pyramidal implementation of LK optical flow [100]. The refinement of the preliminary segments has been achieved by calculating the similarity of short-term time series of histogram of oriented optical flow (HOOF). Radial Basis Function (RBF) is used as the kernel combined with Bhattacharya distance [101] to measure the similarity of HOOFs. The mean colour of each of the superpixels is also used as a supporting metric for achieving a better refinement result. This algorithm has been implemented to work with bottom-up approach to refine the low-level segments to form high-level segments that are corresponded objects. The process is initialized by considering the neighbourhood relationships of key-points and the superpixels in which the key-points are located. Then the superpixels are gradually merged until the segments correspond to each of the objects formed. The proposed approach depends highly on the performance of SLIC algorithm. The specific adoption of SLIC algorithm is discussed in the next subsection.

4.3.1 Simple Linear Iterative Clustering

Superpixels detects pixel-level redundancy in a video frame or an image by means of the colour and intensity. It significantly reduces the complexity of subsequent processing tasks by clustering the redundant pixels together. The literature suggests that the superpixels are increasingly useful for the segmentation of semantic concepts and objects in videos [97]. SLIC superpixels can be obtained by clustering of pixels locally by considering a five-dimensional feature of a pixel, namely the three colour components and the coordinates of the pixel in 2D space. The original implementation of SLIC suggests using Lab colour space that has been standardized by the commission of illumination (CIE). Since SLIC is the core component of this segmentation approach, it is described in detail in this thesis for the purpose of convenience in referencing.

The SLIC algorithm begins by placing a pre-specified amount of approximated superpixels in cells of a grid, with nearly equal grid intervals. This process of placing superpixels avoids the pixels that are on strong edges, having a higher gradient. This step is necessary, as it ensures minimum noise in the approximation of superpixels. Then the pixels are locally clustered by applying K-means. Both the Euclidean distances of Lab colours and spatial distances, which are weighed by spatial proximity emphasizing factor specified in [99], are added together to form a single attribute that can be used in the clustering. The SLIC has been used in this study mainly because it has a linear time complexity compared to the quadratic complexity of original K-means algorithm. Moreover, SLIC ensures that the area in the original frame that is covered by a superpixel has a great homogeneity [99].

The clusters formed by SLIC algorithm results in superpixels that typically segment a single complete object apart to many smaller pieces. For utilizing the motion details to reduce this over-segmentation with the refining process, the histogram of oriented optical flow is calculated.

4.3.2 Histogram of Oriented Optical Flow

The concept of Histogram of Oriented Optical Flow (HOOF) was first introduced in [101] for recognizing human activities. The same study has revealed that the motion details of object can be characterized by using a sequence of HOOFs captured over the time. To calculate HOOF, it is required to obtain a histogram by binning the primary flow angles of optical flow vectors, weighing them according to its vector magnitude. In practical situations, the estimation of the optical flow over different frames is very noisy due to the errors that occur in the key-point tracking process. The noise seems increased significantly as the velocity of the motion gets smaller. Therefore, to reduce the contribution from the noisy points, the feature points are weighted by their velocity before populating the histogram. This approach results in some of the flow angle bins get less count even though there are many feature points flowing to that direction but with very lower magnitude. As described in [101], HOOF is not a feature in Euclidean space; hence it requires a proper projection function to project the HOOFs from its original space to an Euclidean space which may be a higher dimensional than the original space of the HOOFs. This method is proposed assuming that the neighbouring superpixels of another superpixel belong to a region of single object should contain a similar sequence of HOOFs. This may not true if the time considered for extract the sequence of HOOFs is very long due to most of the objects in a video des not last for longer. Hence, it is always better if the approach work with shorter period.

SLIC and HOOF are the base of the proposed object segmentation algorithm. The overall clustering process of SDPF points using the SLIC and HOOF is described in the next section.

4.3.3 Clustering feature points

The clustering algorithm is initialized by extracting SDPF points from the first frame. Then the optical flow is calculated for each SDPF point using the of the LK optical flow algorithm. The pyramidal implementation of this algorithm is used for making it robust for scale differences. A predefined quantity of consecutive frames (T_f) is used to estimate the optical flow by tracking the SDPF points. For every T_f frames, the angle and magnitude of flow trajectory for each SDPF points are recorded. The T_f was

selected empirically, and it was kept at 3 over all the experiments. The smaller value of T_f enabled assessing the proposed method by utilizing a little (more recent) history of motion.

Once the number of frames which has been used to estimate the optical flow reaches the T_f, the SLIC superpixel algorithm is employed to initialize the clustering process. Note that there is no dependency between the optical flow estimation and the SLIC algorithm. Hence the order does not matter. The original implementation of SLIC converts the input image to CIELAB colour space from its original RGB space. The perceptual similarity of a pair of colour points in CIELAB space can be quantified with Euclidean distance, if it is sufficiently small. The *L* colour component of the CIELAB colour model is closely related to the brightness in a scene, which is the none-chromatic perception of the human visual system [102]. Therefore, a significant difference in L component can be seen in the presence of shadows, compared to any non-shadowed regions of the same object. This high sensitivity of L component to highlights that results from blockages of source light, and different source light intensities over different video frames can cause to detect false edges in the estimation of superpixels. The ultimate impact of these facts about L component is that it misguides the SLIC algorithm by creating false boundaries of object due to the non-uniform shadows. Therefore, the L component is weighted before utilizing it in SLIC. The weighting can be done as given in (4.2).

$$L'ab=\begin{bmatrix} m & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} L \\ a \\ b \end{bmatrix} \tag{4.2}$$

The *m* is empirically selected and set to 0.5 throughout all the experiments. The *m* proportionally increases the number of false boundaries that are detected by SLIC. The decrease of *m* causes the SLIC algorithm to neglect actual object boundaries, due to colour differences, which may not result from lack of illumination, but the material properties.

In addition to the colour of the pixels in Lab colour space, the SLIC algorithm further requires the following parameters.

1. minimum distance (a threshold) for colour proximity (N_c)
2. number of clusters (K) which denotes the number of superpixels

Both the N_c and K has empirically been selected and kept as 10 and 50, respectively. It is noticeable that these two values may be required to be fine-tuned for different datasets with a different spatial resolution of the frames.

A superpixel does not represent an entire object instead of a region with relatively uniform colour, which belongs to a single object in higher probability. Therefore, an object may contain more than one superpixel, which is required to be merged by a refinement approach. The HOOF is just a motion cue that denotes whether the different superpixels have similar or dissimilar motions over a short period of time. However, the refinement process takes advantage of a heuristic that is based on colours of the superpixels. This colour-based heuristic in the refinement process discriminates neighbouring superpixels, which consist of the same dominant motion yet belong to two objects. Superpixel algorithm calculates the mean colour of all the regions which clusters to different superpixels as a by-product. This average colour is one of the criteria for a merging pair of superpixels. This mean colour is used as the heuristic of the refinement approach in addition to the other cues mentioned above. The colours are compared by using (4.3),

$$D_c=\sqrt{(L'_1-L'_2)^2+(a_1-a_2)^2+(b_1-b_2)^2} \tag{4.3}$$

where L' is the weighted L colour component and a, b corresponds to the remaining components in the original Lab colour space of the mean colour computed per superpixel.

In the next step, the HOOF is calculated per superpixel by utilizing the optical flow of feature points. The HOOF is computed locally, i.e., one per superpixel. The feature points found within the superpixel are used to get the optical flow and then construct a HOOF. However, HOOF descriptors, which are calculated all consecutive frames, can be different due to the motion that appear in videos. Therefore, instead of constructing one HOOF for the whole set of frames, a time-series of HOOFs is obtained, considering a window of three frames per HOOF. The flow of a feature

points can be considered as a motion vector. If the motion vector of a feature is given with horizontal and vertical components it can be described as v=$[x, y]$. The angle of the resultant vector can be obtained by (4.4)

$$\theta = \tan^{-1}\frac{y}{x} \tag{4.4}$$

where, the range of θ is $-\pi<\theta\leq\pi$. The range of the θ is then mapped to $0<\theta\leq 2\pi$ for calculating the corresponding bin *b* by using (4.5).

$$b = \left\lfloor \frac{\theta}{B} \right\rfloor \tag{4.5}$$

where, *B* is the dimension of the final histogram over angle axis. It is initialized with 30, as suggested in [101]. The contribution from a flow vector to the HOOF is calculated by (4.6);

$$h_b = h_b + \sqrt{x^2+y^2} \tag{4.6}$$

where, the $h_b$is the current frequency of b^{th} bin. All the local flow vectors that correspond to a superpixel are considered to calculate the HOOF of that superpixel. A set of HOOFs obtained for a recorded flow data of feature points within a set of consecutive frames forms a time series that characterizes the motion of the superpixel. At the end of the process, each of the superpixels will have a time series that consists of a set of HOOFs obtained for flows of a predefined number of frames.

HOOF is a vector that does not reside in Euclidean space [101]. Therefore, Euclidean distance does not quantify the similarity of a pair of HOOF descriptors. A vector in a non-Euclidean space can be mapped to a high-dimensional Euclidean space to measure the similarity with Euclidean distance. However, finding a mapping function is not a straightforward task; hence the method specified in [101] that is called as kernel trick is used. In this proposed method, the similarity of two HOOF descriptors is computed using (4.7),

$$k(h_1,h_2)=\sum_{i=1}^{B}\sqrt{h_{1,i}h_{2,i}} \tag{4.7}$$

where, h_1 and h_2 denote the pair of HOOFs that are used to measure the similarity. The study in [101] has specified that the above kernel could also be interpreted as an applying radial basis function on the distance between two HOOFs measured using Bhattacharya's method. The specific equation is given in (4.8),

$$k_{RBF}(h_1,h_2)=\exp\left(-d(h_1,h_s)\right) \tag{4.8}$$

where, $d(h_1,h_s)$ denotes Bhattacharya distance between the two normalized histograms h_1 and h_2. The range of RBF kernel k_{RBF} is defined within [0, 1], where the value of 1 is interpreted as 100% similar and 0 as less similar.

Since the comparison is not only between two HOOPs but also between two time series of HOOFs, it is required to extend the similarity measurement. Therefore, initially the RBF is computed for all the pair of HOOFs at neighbouring superpixels that have been created for the predefined number of consecutive frames. Then their magnitude is computed (4.9),

$$D_h=\sqrt{\left(k_{RBF}^{t=1}\right)^2+\ldots+\left(k_{RBF}^{t=T_f}\right)^2} \tag{4.9}$$

where, D_h quantifies how similar the two sequence of HOOFs that have been computer over set of consecutive frames, and $k_{RBF}^{t=1}$ denotes the RBF obtained for a pair of HOOFs that have been computed for a pair of neighbouring superpixels.

4.3.4 Refining process of superpixels

A segment of an actual object is a unification several superpixels. Therefore, to accurately segment an object, multiple superpixels have to be merged together. Since the object is already segmented to smaller pieces a refinement is required based on predefined merging criteria. The refining process consists of the following six steps.

1. Calculating D_h and D_c for a superpixel, considering all its 8-connected neighbouring superpixels.

2. Checking the merging criteria; Either $D_h > T_h$ or $D_c < T_c$ where T_h and T_c are some thresholds given at the beginning of the algorithm.

3. If the conditions in step 2 satisfied, look for any cluster number already assigned for the current superpixel.

4. If at least one of the superpixels are considered as a cluster number, then it is assigned to both the superpixels. Further, the average colours of the two superpixels and their corresponding time series of HOOFs are merged. Merging of the two average colours can be done by calculating their average. The merging of two time series of HOOFs is done by adding the frequencies of corresponding bins together.

5. If the step 4 detects neither is assigned to a cluster, both should be assigned to a newly created cluster. The cluster number of the new cluster should be one above the number of existing clusters. Finally, the time series of HOOFs of the two superpixels are merged, and their colours are averaged.

6. Then the process must be repeated starting from the second step till there are no superpixels that are not assigned to a cluster.

The most difficult challenge of the refining process of the superpixels is the selection of the set of conditions for the process of merging the superpixels. Because these set of conditions has precisely represented how similar a pair of superpixels, in the context that the similarity is defined as higher similar mean to be the two superpixels belong to the same object. The infinite varieties of colour distribution, which can be observed in different natural objects, do not support limiting the merging criteria only to the colour-based cue. Hence, two neighbour superpixels may contain extremely different average colours, yet belong to the same object. Therefore, in the proposed segmentation method, the merging conditions were selected considering at least a single information out of motion or colour heuristic is always active in the refinement process.

Some reasonable values for both T_h and T_c, had to be explored. The values explored exhaustively by searching all values starting from the maximum of T_h which is 3 (due to the history of motion considered in this study is limited to three frames). It is found that T_c can effectively vary in the range from 0 to 30. The searching space of T_c could be kept at 30 because the suppression of L component by the weighing process applied to pixels in CIELAB colour space. T_c is kept at 15 which is the middle of the range and this empirical selection has been verified by manually inspecting correctness of merging superpixels for different values of T_h. As the conclusion drawn from the results, T_h and T_c , have been set to 1 and 15, respectively. The design of the experiments for evaluating the performance of the segmentation algorithm is reported in Chapter 6, whereas the results are critically discussed in Chapter 7.

4.4 Summary

This chapter elaborates on two significant versions of the proposed object segmentation methods based on feature clustering. Initially, it has explained the attributes of each SDPF feature point, such as the colours and flow direction, and its magnitude is to be clustered to segment the objects. Then it has expressed the way of clustering by devising a method to estimate the number of objects, by using the quantity of features found in a video frame. The second version has been proposed to overcome the errors in the estimation of the number of objects by using superpixels; the histogram of oriented optical flow with a novel refinement technique. The clustering is done with custom distance functions, defined by carefully analysing the feature space, and the proposed refinement technique works combining the different regions using the chromatic, spatial, and temporal details.

CHAPTER 5

CORRESPONDENCE MATCHING FOR OBJECT DETECTION

This chapter presents the proposed method for obtaining a local compact binary descriptor that can be utilized for geometrically invariant object detection by correspondence matching. The method employs a novel deep learning approach to learn a binary representation for local image patches by applying an unsupervised learning technique.

5.1 Overview

In this section, the proposed Deep Neural Network (DNN) architecture, with two main components, is briefed first, followed by a description of the details of each component. Further, a novel learning approach is explained, which can be utilized train the DNN proposed in this study.

The proposed DNN model consists of two main components: Network-in-Network (NIN) model and Restricted Boltzmann Machine (RBM). NIN is used for learning geometrically robust features. The role of the RBM component is to map the features obtained from the NIN component to binary codes. A new learning approach is proposed for training both components. There are two objectives of the proposed learning algorithm, namely, learn invariant-features and mitigate the intra-class variance of the generated binary codes. The weights and biases of the NIN component are initialized with a pre-trained model. This pre-training can be conducted, with the original implementation of NIN purposed for image classification, using an extensive image dataset. The preliminary training may use backpropagation. To avoid the practical limitations resulting from enormous memory requirement for the training, stochastic gradient descent (SGD) can also be used with the backpropagation. Once the NIN component is initialized, the network model is then adopted to the proposed DNN by replacing the global average layer that that can be found as the last layer of NIN model, with a local average layer. This local average layer can expose the features extracted using the first few convolutional layers that have been implemented with multiple multi-layer perceptrons (mlpConv), to the RBM's input layer. This proposed

DNN is trained in two phases, namely, the training RBM and fine-tuning. Fine-tuning is further divided into two tasks, i.e., calculating representative and fine-tuning features. The approach can be briefed as follow,

1. **Training RBM Component:** to learn a mapping function from mlpConv features of different input image patches to discriminable binary codes. The training is conducted using the Contrast Divergence (CD) algorithm as it is being an unsupervised algorithm and the one recommended by the inventors of RBM [103].
2. **Fine-tuning**
    a. **Obtain Representatives:** The mapping function that has been learned at the first level can generate different binary codes for the same input image patch due to different orientations. A new target binary code is generated that represents a set of outputs of a patch if input by applying various geometrical transformations.
    b. **Fine-tuning features:** The weights and biases assigned to the NIN initially are fine-tuned with the representative binary codes, using backpropagation.

The above three levels of training can be conducted several times until the required accuracy is gained. It should be noted that the stopping criteria of this repetitive learning approach are not studied in this study. However, the evidence that an empirically selected number of iterations works reasonably will be examined in Chapter 7.

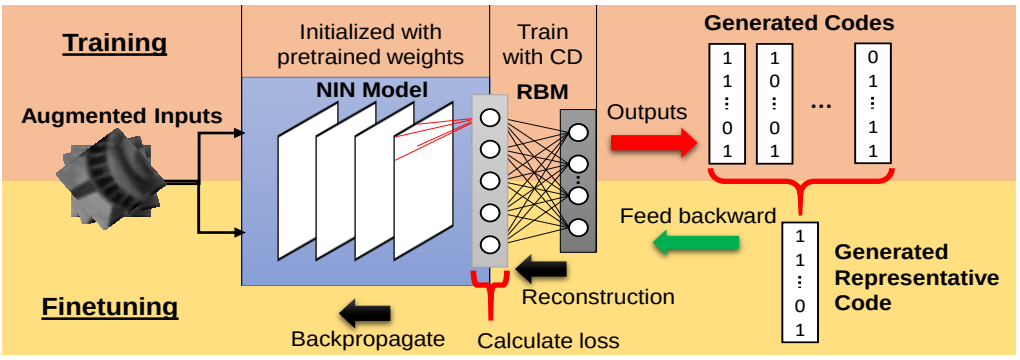


Figure 5.1: The novel NIN-RBM hybrid model with the proposed learning method

The proposed DNN with the two main components is illustrated in Figure 5.1. The two strips of background colours denote the training and fine-tuning phases of the proposed learning approach, i.e., the training and fine-tuning. In the training section, it has been illustrated how the input data is flown through the pre-trained NIN component to train the RBM component. At the rightmost end in Figure 5.1, it illustrates how the outputs of RBM are fed to generate representative binary codes. The bottom horizontal strip in Figure 5.1 shows the fine-tuning section. It illustrates that the generated binary code that represents the outputs of all augmented inputs, is fed back for fine-tuning the parameters in NIN. The method used for training the NIN for image labelling is elaborated in the following sub section. Then the approach used in the first level of training, i.e., training of RBM, is explained in detail. The theoretical background and the algorithm of generating the representative binary codes are explained, followed by an elaboration of the fine-tuning phase of NIN.

5.2 Training NIN model

NIN model distinguishes itself from a conventional convolutional neural network (CNN) model by using micro neural networks instead of typical convolutional filters. Although, in theoretical manner, all the convolution layers can be replaced with the said micro neural networks, practically a small number of real convolutional filters are used combined with the micro neural networks. These micro neural networks are placed inside each of the convolution layers, and the compound is called mlpconv. The study in [86] claims that a few layers of mlpconv can equally function as a traditional

CNN. However, it further elaborated that NIN requires a significantly smaller number of parameters (i.e., neurons) to compete with a giant traditional CNN fairly. Nevertheless, [86] has demonstrated that typical backpropagation can be used to train NIN similar to the training of any conventional CNN, with any kind of image dataset with annotations.

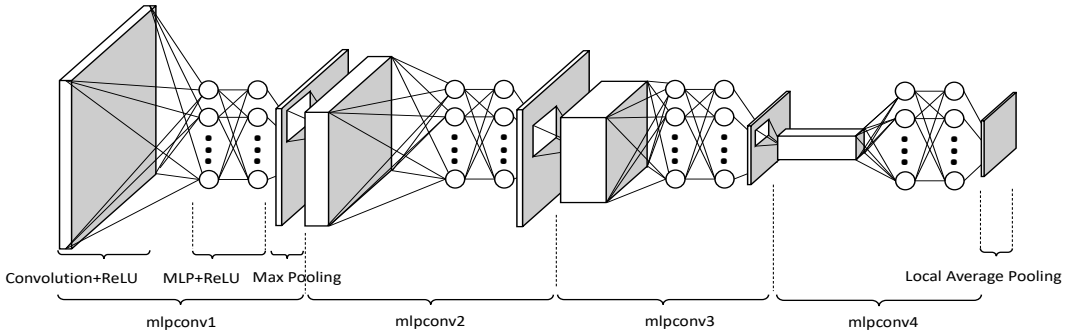


Figure 5.2: Newly adapted NIN model by replacing the last pooling layer

Figure 5.2 illustrates the internal structure of the NIN component used in the proposed DNN, which comprises of only four layers of mlpconv, with each containing a multilayer perceptron (MLP) and a max-pooling layer. The max-pooling layer can select the local maximum from the feature map and propagate them forward in the network. This max-pooling layer helps the DNN to be invariant to a slight translation of the content in an input image. According to Figure 5.2, the model is extremely small when compared with most of the well-performed traditional CNNs, such as the ones that are proposed in the studies of [54], [56]. The relatively shallower depth makes it feasible to train faster. The smaller breath (number of filters that are learned) can learn a set of features that is extremely compact. A visualization of a set of sample filters that have been learned is presented in APPENDIX F. The last layer of the original implementation of the NIN model is a global average layer [86]. Unlike most of the traditional CNNs, NIN uses this global average layer as its classifier, whereas the traditional CNN uses a fully connected layer with a softmax activation function. In this proposed method, the global average layer is used only in the pre-training phase. From the first level of the two primary training processes in this proposed method, a local average pooling (LAP) layer is used instead of the global average pooling (GAP) layer. The GAP does not expose the features through itself. It might be better for classifying

the features and ultimately obtain the class labels of the input. But since the averaging operation is not reversible, it is considered an extremely destructive operation. In the proposed method, it is not suitable to prevent flowing the detailed features from the output layer of the NIN component. In contrast, the LAP is less destructive compared to the GAP, yet it reduces the dimensionality of the feature map that is produced by the set of mlpconvs. Maximum pooling, which is commonly used in conventional CNNs [54], is another technique that can be employed to achieve translation invariance and dimensionality reduction. Maximum pooling discards all locally non-maximal features. However, the output of LAP layer get the contribution from all the features, that may impact the output of the whole model. NIN consists of a set of mlpconv layers that utilize Rectified Linear Unit (ReLU) function as the activation. The output of such mlpconv can be computed via (5.1) for *n* layers,

$$\begin{aligned} f_{i,j,k_1}^{1} &= max\left({w_{k_1}^{1}}^{T} x_{i,j} + b_{k_1}, 0\right) \\ f_{i,j,k_n}^{n} &= max\left({w_{k_n}^{n}}^{T} f_{i,j}^{n-1} + b_{k_n}, 0\right) \end{aligned} \tag{5.1}$$

where, $x_{i,j}$ is the response from a convolutional operation in which the centre point is *(i,j)*, w denotes the vector of weights, b denotes the bias, and k denotes the channel of the convolution response. Successively, the response of GAP is computed from f_{i,j,k_n}^{n}. In this novel NIN-RBM hybrid method, the NIN component contains exactly four layers from mlpconv type; each consists of a convolution layer attached to a set of MLPs that are activated using ReLU (Rectified Linear Unit). Each of these mlpconvs outputs a feature map in different scales. Each of the first three consecutive mlpconv layers is subsampled using a maximum pooling layer, before it is fed into the next mlpconv layer. The sizes of the filters and other internal dimensions in the network should be modified according to the dimension of input image patches. An illustration of the modified NIN network with original filter dimensions and other details can be found in APPENDIX F.

Since the size of the dataset does not fit the memory in a practical system, stochastic gradient descent (SGD) is used for training the model. According to the research work specified in [86], the cross entropy loss of the softmax classifier was used in training. A huge image dataset with annotations is used to train the NIN network. After the

training is converged, the parameters of the NIN are transferred to the NIN component that is used in the proposed method. It should be noted that the NIN component in the proposed method differs from the original NIN, which is used to obtain the pre-trained weights by the last classification layer, i.e., LAP versus GAP layer. The output of the LAP layer is attached to the RBM. The parameters in the NIN component are kept frozen temporarily, during the training of RBM. The purpose of the next training stage is to train the RBM component using the output from the NIN component.

5.3 Training RBM

RBM is neural network that strictly consists of a single layer. In the proposed method, a Gaussian-Bernoulli RBM (GBRBM) is used. GBRBM is considered a as generative model that learns a Gaussian- Bernoulli mapping function over a set of inputs and a set of generated binary codes. A GBRBM can generally be trained using an unsupervised learning algorithm similar to the original RBM that maps Bernoulli distribution to a Bernoulli distribution. Once the training of RBM is converged, it can map data points such that the features of image patch from point in Bernoulli distribution. Practically, GBRBM converts a data point that can be denoted with a set of real numbers to data points that can be denoted by a set of binary numbers. The GBRBM used in this study generates a set of compact binary codes for the input obtained from the mlpconv. The GBRBM can simply convert the features extracted for an input image using the NIN component, to a binary code. The only layer in GBRBM is called a hidden unit, whereas the input is called the visible unit. In this proposed work, the real-valued output of the LAP layer in NIN component is fed into the visible unit of GBRBM. Then it computes the output using (5.2),

$$p(h_j = 1|x) = sigm(W_j.x + b_j) \tag{5.2}$$

where, $p(h_j = 1|x)$ denotes the probability the h_j outputs 1, provided the input x, W_j and b_j denote the weight matrix and the bias corresponds to the computation of h_j. The probability is obtained by using the sigmoid function (*sigm)*. GBRBM can reverse this computation to reconstruct the values at visible layer by giving a binary code to the hidden unit. This computation is done by using (5.3),

$$p(x_k|h) = W_k.h + c_k \tag{5.3}$$

where, $p(x_k|h)$ denotes the probability of x_k element in the visible layer, providing the binary vector h, W_k and c_k denote the weight matrix and bias that correspond to the computation of the x_k. The SGD algorithm is employed using contrastive divergence (CD) learning method, which is a technique that can be used to train an RBM without supervision. CD method requires a negative sample to calculate the loss to back-propagate. Since the data does not contain any negative sample, it is generated using Gibbs Sampling. It can be repeated many times to obtain the negative samples. The number of repetitions is typically mentioned after the term "CD". In this proposed method, CD-1 (single iteration of Gibbs Sampling) is used to generate the negative samples for the first phase of training the RBM. Multistep Gibbs sampling can also be applied, but it takes an enormous amount of time; hence the single step is used. However, CD-5 (5 repetitions) is used in the RBM training of the fine-tuning phase. The use of CD-5 causes estimating relatively less biased negative samples. However, the effect of this CD-5 based negative samples is regulated using a an extremely smaller learning rate. The weights and biases of the RBM are computed by using (5.4), (5.5) and (5.6),

$$W = W + \alpha(h(x)x^T - h(\tilde{x})\tilde{x}^T) \tag{5.4}$$

$$b = b + \alpha\big(h(x) - h(\tilde{x})\big) \tag{5.5}$$

$$c = c + \alpha(x - \tilde{x}) \tag{5.6}$$

where the learning rate is denoted by α , the feature map of a sample training input is denoted by x, a negative sample is denoted by $\tilde{x}$ which is calculated using Gibbs sampling and the function $sigm(W.x + b)$ is denoted by $h(x)$.

The RBM is trained by utilizing the input images in their original form without applying any geometrical augmentation. It is noteworthy to observe that an RBM learns the mapping of two distribution without any supervision. In this proposed method, the RBM maps a set of features that are assumed to behave in a Gaussian distribution to a Bernoulli distribution. The data points in the Bernoulli distribution are binary values. A visualization of a set of binary codes that were generated by RBM is presented in APPENDIX F. These binary values are expected to be compared in

hamming space. However, due to the unsupervised learning, other visual variances, such as geometrical and illumination variations of an input image, may be mapped into extremely different binary values. The core assumption of this mapping is to get similar binary values for similar content of input images, regardless of their orientation and other variances. But the extreme difference in the binary values may not permit their comparison in hamming space. In the final result, this problem will be reflected with lower true positive matching. Ultimately the problem may encourage to re-project the binary values to a known high dimensional space, and use a relevant metric to quantify their similarities, which contradicts the objective of the research, as it increases the computation needs. A novel technique is proposed to achieve a lower intra-class variance for binary values of image patches, regardless of the variations in similar contents. The proposed technique fine-tunes the pre-trained NIN component of the DNN to achieve this invariance. First, a new target feature map of the NIN component for each of the input is generated. This target feature map is generated by using the generative capability of RBM. The process involved in calculating a representative binary code is explained in the next subsection.

5.4 Calculate representative code

Any of the existing binary descriptors that are based on deep learning uses convolutional features that are obtained by training it for image classification purposes [74], [104], [105]; the method known as learning binary codes from frozen features. However, image matching and classification are two different tasks. Hence, the features should be optimized for a specific purpose. When compared with the existing binary descriptors that depends on deep learning, the proposed NIN-RBM hybrid method learns binary codes by fine-tuning the convolutional features and binarization component alternatively instead relying on pre-computed features extractors. The same backpropagation algorithm that has been used in pre-training phase is used in the fine-tuning. However, in the fine-tuning process, the targets of the input image patches are set in a different way. In the initial training phase that is used for the image classification, the inputs are images while the targets denote the class of the image. The class signifies the content of the image. In this proposed method, the input is an

image, but the target is an intermediate feature map instead of a class annotation. This feature map is generated using the RBM component.

To generate the target feature map for an image, the RBM requires a binary code to input to its hidden unit. The objective of the fine-tuning is to minimize the variation of the codes that are generated for content in an image regardless of visual variances. A binary code that represents all the codes that could be generated by the RBM must be generated to achieve the objective if the same content appeared with variations such as geometrical differences. This specific binary code is called a representative binary code in the proposed method. The process of calculating the representative binary code for input image patches consists of several steps.

1. The input patch is augmented to create a set of input patches with different geometrical variations. The augmentation includes random rotation and cropping of image patches.
2. The augmented image patches are input to the DNN model and their binary codes are record.
3. The representative code is calculated for the set of recorded binary codes using (5.7),

$$H_j = \begin{cases} 1 & \left(\sum h_j^l\right) \geq |L|/2 \\ 0 & otherwise \end{cases} \tag{5.7}$$

where H_j denotes the j^{th} binary value of the representative binary code H of the original input image patch, h_j^l denotes the j^{th} binary value in the binary code that has been output by the RBM for the augmented sample patch l and $|L|$ denotes the quantity of samples created by augmenting the input patch. The procedure mentioned in (5.7) can be seen as a method of finding the centroid of a distribution of binary data points in hamming space. Theoretically the summation of hamming distance from each of the observed point in the distribution to a single point will be the minimum only if this point is located at the centroid of the distribution. Therefore, the computed value of H_j, that is the centroid of all the binary codes generated for the patches from a single image can be a good selection for representing all the binary codes. This H_j is fed to

the RBM from the hidden layer and the visible units are reconstructed. Then the NIN is fine-tuned by forcing it to generate features more likely the reconstructed value at the visible unit. It is hypothesized that if NIN can generate a feature map that is similar to the reconstructed values at the visible unit for an image patch, then the other variations of the same patch should output extremely similar binary codes that is closer to the representative code, such that the centroid. A visualization of a set of reconstructed feature maps are illustrated in APPENDIX F. The next subsection describes the approach followed to fine-tune the feature extraction model.

5.5 Fine tuning NIN

The objective of fine-tuning the features is to minimize the variation of the codes that are obtained for different orientations and other augmentation of an image. The fine-tuning process of the NIN requires a new target per image patch. Although the target of an image was its class label at the initial training of NIN, the new target for the fine-tuning is a feature map. This new target should be generated as it optimizes the above said objective. For example, if the NIN portion of the model takes several inputs that are geometrically different variants of an image patch, the outputs must be closely similar. These similar outputs will be the inputs of the RBM component; hence it should generate similar binary codes for these inputs. The new target that is the desired feature map f will be generated by giving the representative code H to the hidden unit of the RBM using (5.8),

$$f_k = W_k.H + c_k \tag{5.8}$$

where f_k is the k^{th} element of the feature map f. It is observed that the $\tilde{x}$ in (5.4), (5.5) and (5.6) is calculated using Gibbs sampling, which is similar to the equation (5.8). In Gibbs sampling, instead of the H, the generated binary code is given. In this proposed process of feature map generation, the H is obtained using (5.7), which indicates that the optimization terminates if $\tilde{x} = x$. Practically, it can be further seen that if an input that is similar to a reconstructed map is fed to a well-trained RBM, then it will produce a binary code that is similar to the one which has been fed for the reconstruction.

The RBM component of the model is temporarily detached from the compound DNN. The fine-tuning is done similar to the training using backpropagation. The only difference is that it uses the feature map *f* that is generated using (5.8) as the intended output for the different versions of a patches that are computed by applying various image augmentations. The optimization uses the SGD with L2-loss, i.e., the mean square error. An extremely small learning rate that is about $1/10^{th}$ of the last step learning rate at the training phase was used. This fine-tuning tends the various augmented inputs of an image patch to output nearly analogous feature maps. These are then used to obtain the codes that are closer in hamming space using the RBM.

After the fine-tuning of the NIN is converged, the RBM is attached to the NIN. Then RBM is fine-tuned a similar way to the training of RBM but using CD-5 instead of CD-1 to generate the negative samples. CD-5 can be obtained by repeatedly applying single step Gibbs sampling on the output of generated input at the previous iteration, for five times. The training process of both NIN and RBM components are computationally expensive. Hence updating the parameters of the NIN and RBM repetitively impractical. However, empirical termination of this proposed learning process works in practical situations, which will be discussed in Chapter 7. The proposed learning process is terminated soon after it is converged with CD-5. At this stage both the components have been trained and fine-tuned at least once. The specific parameters that have been used in this proposed NIN-RBM hybrid model and the design of experiments that were conducted for the evaluation are described in Chapter 6. The proposed binary descriptor will be referred as NIN-RBM in the evaluation, due to its strong dependency on NIN and RBM networks.

5.6 Summary

This chapter has presented the proposed deep learned binary descriptor for object detection by correspondence matching. First, the overall deep neural network model has been explained by emphasizing the importance of a convolutional layer implemented with multi-layer perceptrons for learning invariant features. Then the importance of Restricted Boltzmann Machine for unsupervised learning of feature to binary code mapping is elaborated. Finally, the learning approach, along with novel finetuning technique, has been proposed for learning an invariant yet highly

discriminative binary descriptor for correspondence matching tasks. The next chapter presents the validation methods, metrics, experimental setup that are used for evaluating the proposed works.

CHAPTER 6

EXPERIMENTAL SETUP AND VALIDATION METHODS

This chapter explains the design of the experiments, the datasets and evaluation metrics used for assessing the different aspects of the proposed methods for object recognition, video segmentation, and specific object detection by correspondence matching.

6.1 Evaluation of object recognition

The first two versions, i.e., the SDPF and SDPF-DDD, of the proposed feature descriptors have been compared separately due to the use of different datasets in the original experiment. The third version, i.e., the Hessian based SDPF (HSDPF) has been assessed comparing with the first two versions and the other state-of-the-art techniques. The main concern of the evaluation is the classification and computational performance; hence several suitable metrics have been used.

6.1.1 Datasets

Since object recognition can be applied to individual frames without considering the temporal information, the proposed methods were evaluated on static images.

SDPF and SDPF-DDD: The experiments on the initial version of the SDPF and the SDPF-DDD were conducted on Corel-1000 and Caltech dataset. Corel-1000 dataset consists of natural images, which are classified into several classes; each consists of 100 images. Caltech dataset, which has been used in many studies regarding object detection and classification, has also been employed to verify the results. For the experiment, 10 image concept categories were selected from the Corel-1000 dataset and six categories from Caltech dataset. In Corel-1000 dataset, each category contains 100 images, whereas Caltech dataset contains 200 images per category. The training split contained 40% of data from each class where the remaining used as the test split. SVM is used to evaluate the classification performance. The datasets are made more challenging by modifying the images in the test dataset, randomly changing the orientation (by 0^o, 90^o,180^o and 270^o) and scale (by 0.5, 1, 1.5 and 2) for assessing the

correct classification rates of SDPF and SDPF-DDD descriptor under drastic variances in orientation and scale of visual contents.

Hessian-based SDPF (HSDPF): HSDPF were evaluated using two openly available datasets. The first dataset is coil-100. It contains exactly 7200 images. All these images contain non-natural objects. The dataset contains 72 versions of a single object, captured from different view angles around the vertical axis of the object. These view angles range from 0 to 360 degrees. The objects in the images are uniformly illuminated and tightly cropped around. The background consists of a solid colour. The robustness of the HSPDF to different view angles of the objects is evaluated with this dataset.

The second dataset is ALOI that contains of total of 96000 images. Out of these 96000, a set of 24000 images have non-uniform illumination. This subset of the dataset is referred as ALOI-ill in this thesis. The remaining 72000 contains objects that were captured in different view angles. This subset of the dataset is referred as ALOI-View in this thesis. Both ALOI-ill and ALOI-View consists of 1000 object classes. These two datasets were used for the purpose of evaluating the invariance of HSDPF in classifying images with non-uniform illuminations. Moreover. The ALOI-View permits to evaluate how well the HSDPF response in classification objects that have been captured from different angles of view. Furthermore, since this is very large dataset, it permits to evaluate how HSDPF behave when applied to classify images among large number of object classes. The objects appeared in the images from both the datasets are in their upright position. However, there may be requirements for recognizing objects which are not in their upright pose. Therefore, some randomly selected images from the ALOI-View dataset were augmented by rotating them to four different angles.

6.1.2 Evaluation metrics

Two different metrics have been proposed to evaluate the performance of the SDPF, SDPF-DDD, and the HSDPF due to the different complexities of the datasets.

Evaluation Metrics for SDPF and SDPF-DDD: The average computation time of feature extraction and description is calculated using a subset of images from the Corel-1000 and Caltech datasets, due to the variations in image dimensions. The spatial dimension of the selected images from Corel-1000 datasets is 384x256, and that of the selected images from Caltech datasets is around 800x600. All the experiments regarding the SDPF and SDPF-DDD were conducted in a machine with a Core i7 3.4 GHz 64-bit platform. Precision, recall, and F1 score are used to evaluate the two descriptors, and class-wise performance has also been mentioned in the following subsection. The smaller dataset permits doing in-depth class-wise analysis, compared to the difficulties arising with larger datasets.

The precision and recall are calculated using (6.1) and (6.2), respectively,

$$recall\ \ (r) = \frac{|R \cap A|}{|R|} \tag{6.1}$$

$$precision\ \ (p) = \frac{|R \cap A|}{|A|} \tag{6.2}$$

where, *R* is the set of images that are relevant by the ground-truth, and *A* is the answer set.

To compare the weighted average of recall and precision, F1 score given in (6.3) is used. *F1 score* reaches 1 for the best result and 0 for the worst case.

$$F1 = 2\frac{(p.r)}{(p+r)} \tag{6.3}$$

Evaluation Metric for HSDPF:

The classification performance of HSDPF was compared by the precision. Since the objects are from multiple classes, the class average precision is calculated using (6.4),

$$Class\ Average\ Precision\ (AP) = \frac{\sum_{n=1}^{N} \frac{|R_n \cap A_n|}{|A_n|}}{N} \tag{6.4}$$

where R_n is the set of images that is relevant to class n according to the ground-truth, and A_n is the set of images that the classifier found to be relevant to class n.

In addition to validating the improvement over the previous versions of the SDPF implementation, the HSDPF has also been assessed for finding its applicability resource constrained hardware. Currently, Graphic Processing Units (GPU) are commonly used for accelerating similar feature extractors. Moreover, Single Instructions Multiple Data (SIMD) is also commonly used for speeding up the algorithms using all potentials of Central Processing Unit. However, these resources may not be available in all the devices that may be used for some applications. Therefore, none of these accelerations were used for the computation of features as well as the classification process. However, all possible hardware resources, including GPU and SIMD registers, were utilized for the training as it is generally a one-time process and not necessary to be done with in the device that is used to deploy the application. The training of CNN and SVM were conducted in a full-sized desktop computer. The hardware configuration of the computer is given below.

- Processor: Intel i7
- Model: 2600
- Clock rate: 3.4 GHz
- RAM: 8 GB
- GPU: Nvidia GTX770
- Operating System: Ubuntu

The testing of the two were conducted in a Raspberry Pi in addition to the desktop computer given above. The specification of the Raspberry Pi is given below.

- Device: Raspberry Pi 2 Model B
- Processor: ARM Cortex-A7
- Clock rate: 900 MHz (quad-core)
- RAM: 1 GB
- Operating System: Raspbian OS.

It is to be noted that the CNN was implemented in two different methods for the desktop PC and the Raspberry Pi device due to technical limitations. The desktop version used Caffe framework to enable the training on GPU. The version of the CNN that is implemented for the Raspberry pi use OpenCV library. The OpenCV based implementation was not used for training by it utilized the weights and biases that have been trained on the desktop with the Caffe based implementation. The dimension of the images was kept constant at 128 x 128. The average time taken for the computation of features was recorded. This computational performance was assessed using ALOI and Coil-100 datasets. The computation time for both training and classification phases were measured.

6.1.3 Assessing the invariant properties of SDPF

SDPF was evaluated for its invariant properties. First, the rotational invariance was assessed. The SDPF descriptor that consists of a two-dimensional histogram was used for this assessment. The assessment has been done by using Corel-1000 dataset. It is a dataset that contains 10 categories, each with 100 images. The dimension of the descriptor kept as 4 distance bins and 12 colour bins, as this dimension gave the best classification accuracy in general case. Similarly, the scale invariance also assessed using the same dataset.

Both rotational and scale invariances are measured using a confidence value that represents the likelihood of being the rotation or scale augmented images predicted to their true category by the SVM.

6.2 Evaluation of segmentation

The proposed segmentation techniques have been evaluated by means of completeness and accuracy. The two proposed methods were compared using different clustering methods. There was no motion clustering-based object segmentation method that exists in the time of the study; hence some well-known conventional motion and structure-based segmentation methods were used for the comparison.

6.2.1 Datasets

The CamVid dataset [96] was used for assess the performance segmentation. The specification of the video contained in the dataset is as given below.

- Duration: 10 minutes
- Frame rate (Video): 30 Hz
- Frame rate (Labelled Image): 1 Hz and 15 Hz
- Number of video sequences: 5
- Quality: High definition
- Number of semantic classes: 32

It is initially created for video frame segmentation, and the ground-truth segments are also provided along with the dataset. This dataset consists of both static and moving objects, and both the global and local motion make it challenging to segment accurately.

6.2.2 Evaluation metrics

The proposed segmentation method is assessed by using the spatial accuracy measure S_{er} [106]. It can be computed by using (6.5),

$$S_{er} = \sum_{n=1}^{N} \left| M^{cl}(n) \oplus M^{ref}(n) \right| \Big/ \sum_{n=1}^{N} \left| M^{ref}(n) \right| \tag{6.5}$$

where, the number of objects is denoted by N, the set of points that has been clustered to the object n by the proposed method is denoted by $M^{cl}(n)$, and the set of points that has been manually clustered to the object n (the ground truth) is denoted by $M^{ref}(n)$. The set operation, "exclusive disjunction" is denoted by the symbol $\oplus$.

The completeness measure (S_{compl})enables measuring how well the segmentation method achieved its objective of the segmentation technique quantitatively. It can be computed using (6.6),

$$S_{compl} = \sum_{n=1}^{N} \left| M^{cl}(n) \cap M^{ref}(n) \right| \Big/ \sum_{n=1}^{N} \left| M^{ref}(n) \right| \tag{6.6}$$

where the symbol ∩ denotes conjunction of the two sets.

6.3 Evaluation of correspondence matching

The performance of the NIN-RBM binary descriptor, NIN-RBM has been assessed by employing it in two applications of correspondence matching, namely, the instance retrieval and colour patch matching tasks. Moreover, the computation cost of the approach has been assessed comparing with other best performed approaches, using several applicable platforms. Since the correspondence matching does not depend on the temporal information, the evaluations have been carried out on images instead of using videos.

6.3.1 Datasets

The initial training of the NIN component was carried out using ImageNet ILSVRC2012 [107] dataset. It includes around a total of 1.4 million natural images. For the remaining phases of the training and the testing for different tasks, including the correspondence matching and instant retrieval, the following datasets were used.

1. Oxford [108]
2. Paris [109]
3. INRIA Holidays [110]
4. Brown [111]
5. HPatches [112]
6. Rome patch [55]

In addition to the above-mentioned datasets, some randomly selected images from CIFAR-10 [113] dataset were also used for evaluating the computational cost of the proposed descriptor, NIN-RBM. The datasets mentioned above are briefly described in this subsection to convey what aspect of NIN-RBM is measured with each of the datasets.

ImageNet ILSVRC2012 [107] is an extensive dataset that consists of natural images. The typical training split includes around 1.2 million images, whereas the validation and testing splits contain 50000 and 100000 images, respectively. The dataset provides an annotation of the images among 1000 different object classes. It is a commonly used dataset that has been introduced as the benchmark in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC). It is very beneficial to use this vast, manually annotated image dataset, as it enables learning a set of generic visual features from the NIN component.

Brown [111] dataset consists of a total of 1200000 image patches. These local image patches are extracted from three different landmarks, namely, Liberty, Notre dame, and Yosemite. The images extracted from these three sets are significantly different, as they were captured at view of extremely different angles relative to the landmark. There are around 400000 grayscale images that are included in each of the subsets. The training split per subset includes around 200000 training pairs, whereas the test split consists of 100000 pairs. Out of all the pairs used for the training and testing, only half of it is selected as the correctly matching pairs, whereas the remaining half is kept as non-matching pairs. The recent studies related to visual descriptors that describe local patches have commonly used Brown dataset. The images from different sets are at different levels of difficulty in the matching task. To evaluate the robustness of the descriptors that are trained with images from different datasets, the dataset was prepared to form all the combinations of cross-category training and testing configurations. All these combinations were used in the evaluation of NIN-RBM. It enables determining how independent NIN-RBM is compared to the dataset that has been used for the fine-tuning. This independence is one of the main objectives that is expected from an unsupervised/self-supervised DNN. Moreover, highly data-dependent methods are very limited in practical applications without retraining using domain-specific data.

All the image patches that are included in Brown dataset are grayscale. Hence the dataset cannot be used to measure the performance of the descriptors when colour information is present. This problem was solved using RomePatches [55] dataset.

There are 20000 colour image patches found in RomePatches dataset. These local image patches have been extracted from several landmarks that are situated in Rome. Both the training and testing splits include 1000 patches in each split. The patches in any of the splits were extracted from 10 different views. The effective utilization of colour for local patch matching can be evaluated using This dataset. Moreover, since the images in RomePatches were taken from 10 different viewpoints, the performance can be compared in the presence of both colour and geometric variances.

Oxford [108] dataset consists of 5062 images of Oxford landmarks. These images were captured from a total of 11 different landmarks in Oxford. Paris [109] dataset consists of 6412 image patches that were captured from 11 different landmarks in Paris. The scale and viewpoint variation exist in the images of both Oxford and Paris datasets. Therefore, the two datasets enable measuring the performance of the descriptors in correspondence matching tasks, with the influence of extreme geometric variances. The comparison is made with the Oxford by selecting five random images as the query, per each landmark. The same comparison was made with the Paris dataset as well by using 55 randomly selected images as the query from each landmark.

Another dataset called INRIA Holidays [110] was used in the evaluation to test the performance of correspondence matching of images, that consist either a distinct scene or an object. The dataset includes a total of 1491 images, which have been annotated with 500 labels of groups. The images within the group differ by illumination and the amount of blurring, in addition to the scale and viewpoint variations, enables comparing the performance in correspondence matching in more challenging cases. The evaluation uses 500 query images.

The HPatches [112] contains around 1.5 million patches. The purpose of using this dataset is to reduce the inconsistencies among the methods used to evaluate the existing local descriptors. The evaluation protocol of HPatches dataset includes three benchmark tasks, each of which should be evaluated according to a strict mechanism on the data. The dataset contains image patches three categories namely *easy*, *hard*, and *tough*. The names if thus categories imply the geometric noise level presents in the data. Hence, the usage of HPatches makes it possible to compare the performance

of NIN-RBM when applied to patches with known levels of noise. These datasets were used for assessing the performance of instance retrieval tasks, by computing the similarity of the local patches that are extracted from each of the images.

CIFAR-10 [113] is commonly used for evaluating computational performance in many related studies. CIFAR-10 contains images with a fixed image dimension, which leads to a consistent aspect ratio. These consistencies enable performing a non-biased evaluation of the computational efficiency of NIN-RBM. Furthermore, it consists of natural images that contain information from all colour channels. The dataset consists of 60000 colour images, which are annotated with 10 different labels of object classes. In this study, the speed of encoding the image patches to their binary codes is evaluate using CIFAR-10 dataset.

Some samples from each data set used above are available in APPENDIX H.

6.3.2 Evaluation metrics

This study assesses NIN-RBM in matching image patches given in the datasets that contains image patches. The correspondence is explored for retrieving similar instances to the query patch from the dataset.

The datasets that contain image patches shipped with the ground truths. Therefore, the precision and recall metrics can be used for the comparison. The 95% error rate is derived from the precision and recall results. It denotes the rate of false-positive in patch similarity matching when it achieves a 95% true-positive rate. The false-positive and true-positive rates can be varied by gradually increasing the Hamming radius at pairwise similarity matching. The radius getting higher typically results in more robust similarity matching with more patches found. This improves the true-positive rate as well as the false-positive rate. Lower radius results in strict similarity matching, which causes low false-positive but, at the same time, low true-positive rates. The objective is to keep good true-positive rate while at a lower false-positive rate. Hence, the 95% error rate implies that, if it is low, the descriptor is better in similarity matching, in terms of excellent handling of both true-positive and false-positive pairs. Moreover,

the 95% error rate has been widely used in related studies for measuring the performance of visual descriptors.

Since the 95% error rate does not show how the precision and recall is varying for the rest of the Hamming radius, a set of Receiver Operating Characteristic (ROC) curves are also presented. The limits in the discriminability and robustness of NIN-RBM will be evaluated comparing with many other best performing binary descriptors.

Moreover, the mean average precision (mAP) is also used in the evaluation. Calculating mAP does not require obtaining the results repetitively by changing the Hamming radius. Instead, the similarity can be computed once per each descriptor, keeping the radius at a value which gives 100% recall. In the process of evaluation, several matching patches are retrieved and rank them according to the confident value of matching. Then top k patches are evaluated be true or false positive by comparing with the ground truths. It is noteworthy that within this k predictions all the true positives are contained hence the recall is 100%. Then mAP can be calculated by using (6.7),

$$mAP = \frac{1}{N}\sum_{n=1}^{N} AP_k(n) \tag{6.7}$$

$$AP_k(n) = \sum_{i=1}^{k} p_n(i)\Delta r_n(i) \tag{6.8}$$

where, N is number of query patches, $AP_k(n)$ is the average precision for class n considering top k predictions considered that are ranked according to the confident value, $p_n(i)$ is the precision of to the i^{th} prediction for class n and $\Delta r_n(i)$ is the difference of recall i.e., $[r_n(i) - r_n(i-1)]$. The $r_n(0)$ can be set to 0.

The $p_n(i)$ can be calculated using (6.2) whereas the $r_n(i)$ can be calculated using (6.1). The mAP percentage is used for the evaluation of the descriptors with RomePatches dataset. The specific metric with the remaining parameters that has been recommended for RomePatches dataset can be found in [55]. The method of obtaining the confident

value differs according to the method of classification. It is the hamming distance for the approach proposed in this study.

The objective of the instance retrieval task is to fetch similar patches to the given instance of the query patch. The fetching challenges as there are many similar and dissimilar local patches in the dataset. These local patches are grabbed surrounding the key points that have been detected by an automatic key-point detector. To assess the performance of NIN-RBM, in instance retrieval task, mAP metric, which has been employed in many other related studies as well, has been used. It is promoting that mAP can be calculated for mini batches instead of waiting for evaluating the whole dataset. Mini-batch processing is ideal for processing an extensive dataset in devices with limited memory and processing power. Furthermore, concurrent computation of descriptors over multiple patches is more efficient than serial computation in general cases.

One aspect of the complexity of a DNN can be represented in terms of the quantity of parameters used within the model. The quantity of parameters is typically proportional to the number of neurons. Moreover, it directly affects the computation time in both training and testing phases. The number of DNN parameters is reported per each descriptor for the comparison in this study. When it is combined with the precision of parameters, the total storage requirement can also be obtained. The total storage requirement of the DNN is essential not only because it costs hardware, but also it costs more time in loading the model at the beginning. Hence the storage requirements of the uncompressed models are also reported in megabytes to be used in the comparison. Furthermore, the specific encoding time utilized by the NIN-RBM and the other baseline methods is also reported in the comparison. Since the computation time depends on the platform, the specification is also reported along with the encoding time.

6.4 Summary

This chapter has reported the experimental setup and the validation methods for evaluating the proposed low dimensional visual feature descriptor in object

classification, the novel feature clustering approach for object segmentation and the proposed deep learned binary descriptor for known object detection via correspondence matching. The datasets used for each of the experiments have been described relating to the aspects of performance assessed. Then the evaluation metrics of each experiment have been presented. The next chapter presents the results and discussion of the experiments reported in this section to support the conclusions made in this thesis.

CHAPTER 7

RESULTS AND DISCUSSION

This chapter discuss of the results of all the experiments carried out in the study. Firstly, the results of evaluating the proposed object recognition methods have been discussed. Under the object recognition, the initially proposed salient dither pattern feature (SDPF), the second version which combines the initial method SDPF with dither density descriptor (SDPF-DDD) and the third version Hessian base SDPF (HSDPF) have been assessed. Then the two proposed segmentation methods have been discussed. Finally, the evaluation results of the proposed local descriptor for correspondence matching have been discussed.

7.1 Invariant properties of SDPF

Before assessing the object recognition capabilities of SDPF, a separate study was conducted to assess its invariant properties. In this study, both the rotational and scale invariance of SDPF were assessed.

7.1.1 Rotational invariance assessment

The rotational invariance was assessed using a set of sample images augmented to 360 different versions. The source images used for the assessment were augmented by rotating them over the step of 1^0 until the last rotated image is rotated by 360^0 compared to the original image. An SVM was trained by giving six random samples labelled as class 1, which have been manually created by rotating an image to different angles, along with 99 other images that contain the object of same type as the class 2. For an example, an image from the set of images of buses (as shown in Figure 7.1) were selected and create six sample images rotating by random angles and annotate them as from class 1. The remaining images from the same image set that contains buses were labelled as class 2. Therefore, the SVM get images from the two classes yet all the images contain buses. The intended result is that the SVM classifies the input to the exact image that is in different orientation. This approach makes it difficult to discriminate between the two derived classes (one class contains a different orientation of a specific image, while the other contains different images consisting of

similar objects to the other class), as both classes contain images from the same semantic class.

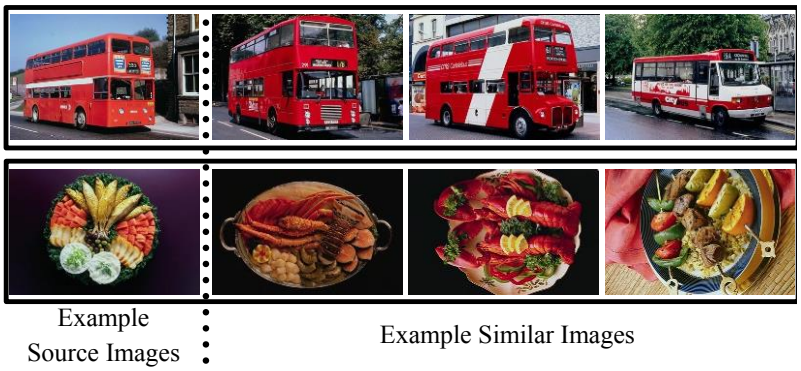


Figure 7.1: A sample of images that were used to evaluate the invariant properties

The probability of being an image from a specific class was obtained using SVM model classifying all rotated versions of images. Given an image indicates a probability in the prediction higher than 0.5 for a class, it was considered that the image matched to the class. The process was repeated for two image categories by randomly picking the query images. Both Figure 7.2 and Figure 7.3 indicate the probability of matching the augmented images of two samples of the randomly selected source images.

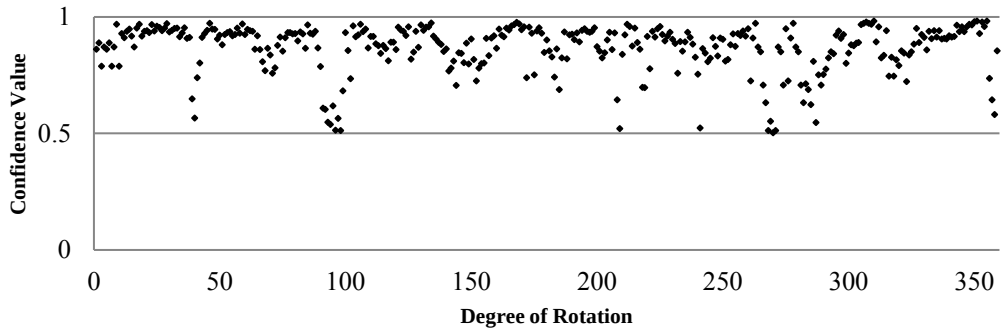


Figure 7.2: Probability of rotated images classified in their true image class, obtained for example image 1

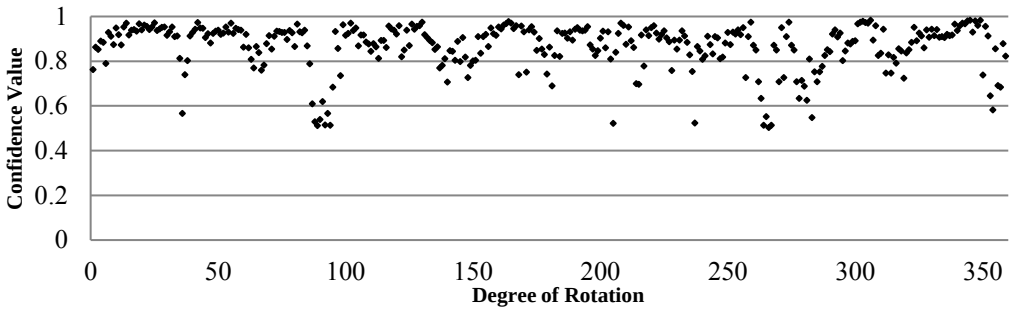


Figure 7.3: Probability of rotated images classified in their true image class, obtained for example image 2

The graph in Figure 7.4 presents the mean probabilities calculated for the set of augmented images from the two source images.

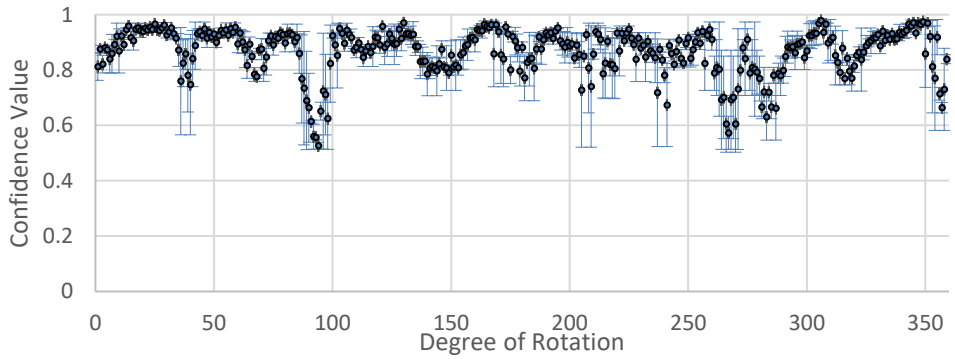


Figure 7.4: Mean probability of rotated images classified in their true image classes with the standard deviation

Any image that has been augmented by rotation was classified to its ground truth image class by the SVM with the probability above the 0.5 hence according to the criteria defined in the experiment, 100% of the augmented images are found to be a true positive. The mean probability has been computed and the result is 0.87 when consider all the source images. The result implies a higher likelihood of the SDPF descriptor computed for the rotationally augmented inputs classified to its ground-truth class. It is noteworthy that only 2% of images were used for the training and the images for the negative class is selected from the same object type. The probabilities of true prediction can be observed to be significantly smaller compared to the calculated mean, for certain angles of rotation. Both Figure 7.2 and Figure 7.3 exhibit the said fluctuation in the output probabilities. Precisely, 100^0, 300^0, and 360^0 indicate

significantly lower probabilities. This nature of the fluctuations in probabilities over different orientations might have been caused by the box-shaped dither patterns, which are considered in the preliminary levels. The result of assessing the scale invariance property of the SDPF is presented in the following subsection.

7.1.2 Scale invariance assessment

The scale invariance is one of the crucial measurements of any feature descriptor to assess the performance of its robustness over different sizes of the objects being identified. The scale invariance was assessed by creating eight augmented images by resizing an input by the factors 2, 1.75, 1.5, 1.25, 0.75, 0.5, and 0.25 in both horizontal and vertical axis. The effect of resizing with different factors is illustrated in Figure 7.5.

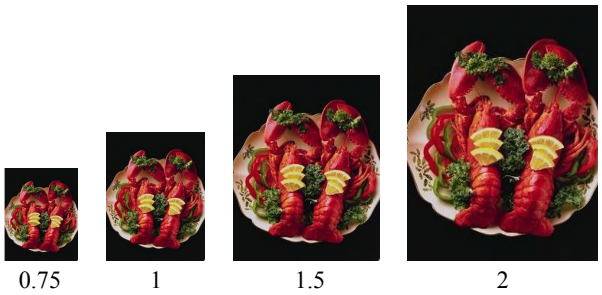


Figure 7.5: Resized versions of an input with the scaling factor

The graph in Figure 7.6 indicates the average of probabilities of predicting the class of the input to its ground truth classes for different scaling factors mentioned above.

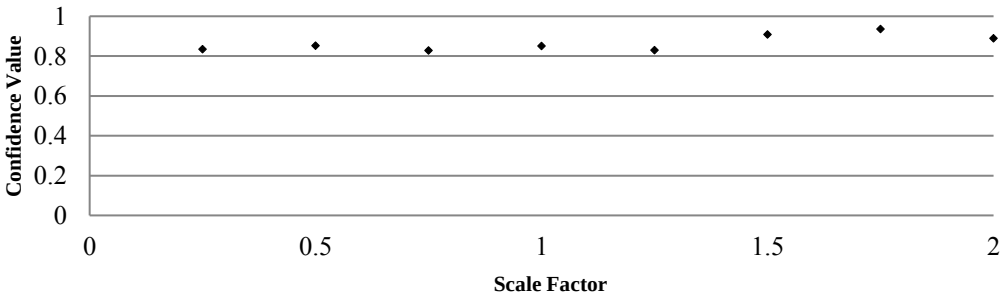


Figure 7.6: Mean probability of predicting an input to its ground truth

The probability of any prediction of the input to the ground truth is always above 0.5. Hence it achieved 100% of true positive rate for the prediction of augmented inputs to their ground truths. The mean probability of predicting the eight augmented inputs to their true class is 0.87 by coincidently being similar to the result of the evaluation of rotational invariance. The result suggest that the feature description exhibits a higher likelihood of matching an image to its ground truth, regardless of dramatically scaling, yet trained using only one out of the eight images to the true class and 99 other similar images to its counter class.

7.2 Assessing the performance of SDPF and SDPF-DDD

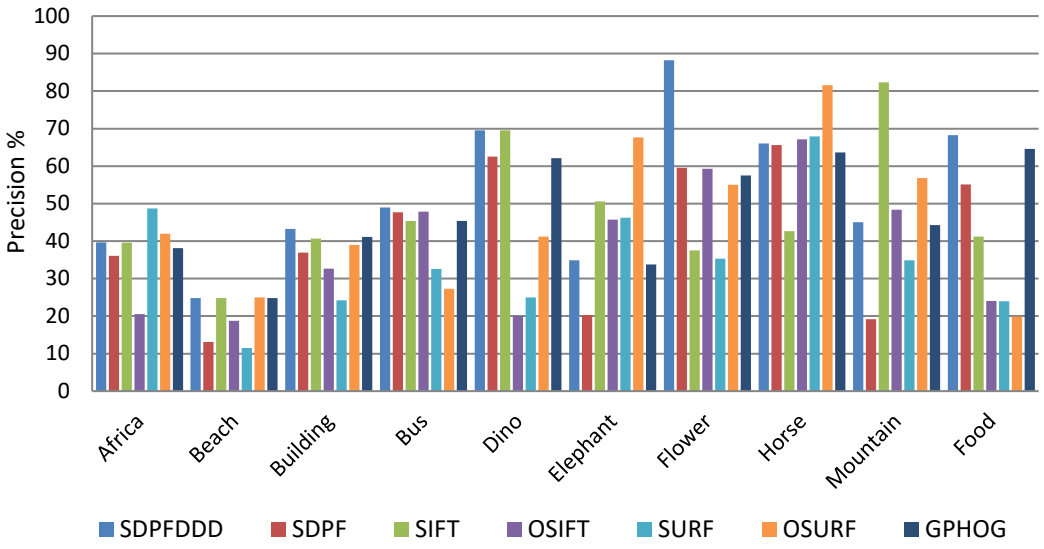
Table 7.1 displays the mean time of computation and the dimensionality of the comparable descriptors. The data indicates that the fusion of DDD with the SDPF increases the dimensionality and the computation time of the SDPF. However, the SDPF-DDD still maintains a much lower computation time and a dimensionality, compared to the other descriptors. Particularly the computation time of SDPF-DDD is around 10% of the computation time of BoF-SURF, which is the next lowest.

Table 7.1. Dimension and average extraction time of feature descriptors

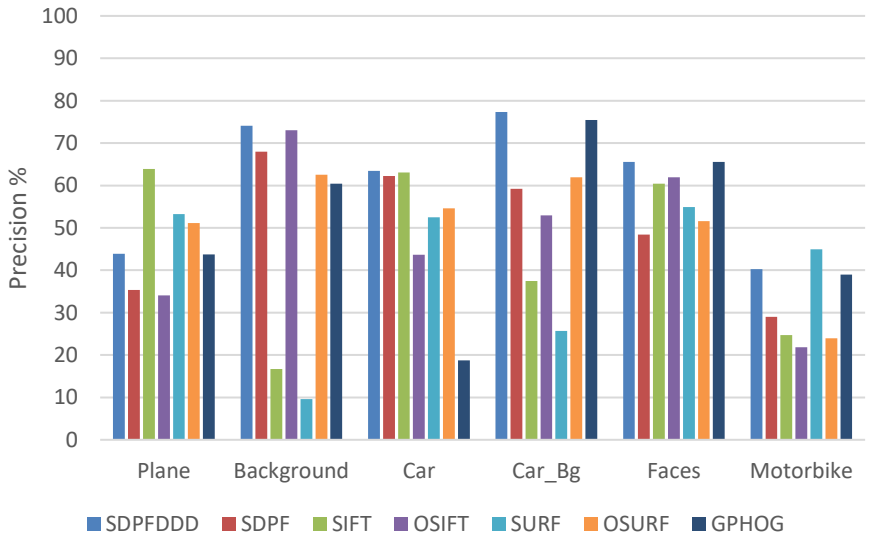
Descriptor	Average Time (ms)		Dimension
	Corel-1000	Caltech	
SDPF-DDD	24	65	128
SDPF	9	15	48
BoF-SIFT	267	720	200
BoF-SURF	251	670	200
BoF-OSIFT	723	2230	600
BoF-OSURF	622	1679	600
FC-GPHOG	3200	9100	1200

Figure 7.7 presents the performance of seven descriptors with Corel-1000 dataset and Caltech dataset in terms of precision. The result clearly indicated that the SDPF-DDD improves upon SDPF by means of precision. Besides the improvement, the novel fused descriptor indicates competitive precision values for several visual categories, including beach, building, bus, dino, flower, food, background, car, car_bg, and faces.

Further, this result proves the consistency of the precision of SDPF-DDD over different visual concepts and even different datasets.



(a)

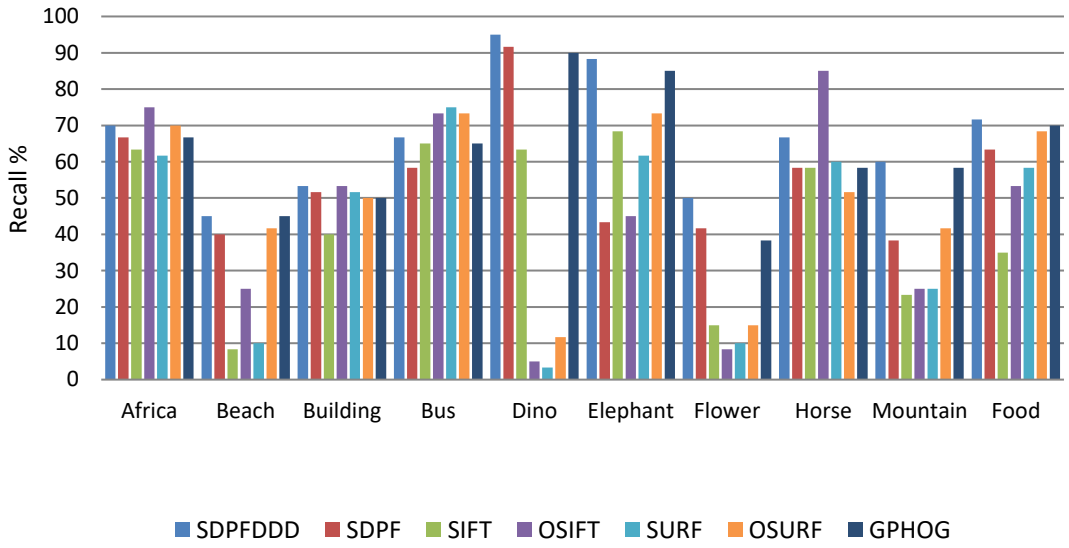


(b)

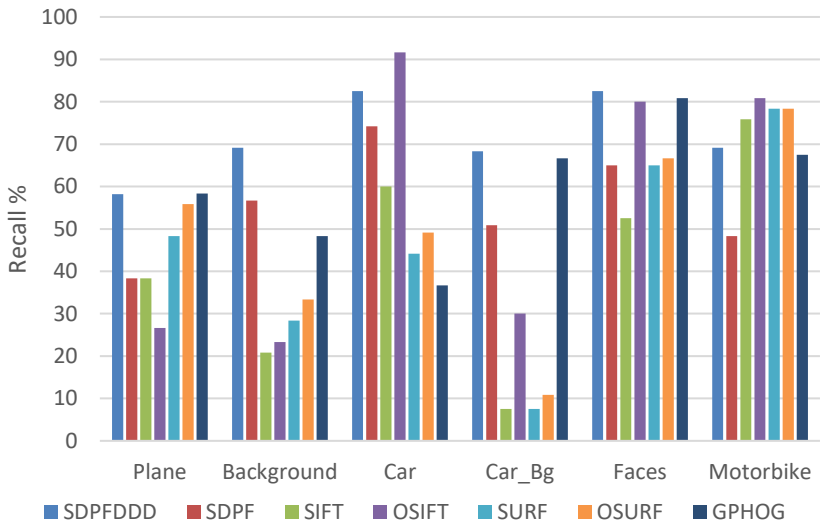
Figure 7.7: Comparison of retrieval precision; (a) ten visual concepts in Corel-1000 dataset, (b) six visual concepts in Caltech dataset.

The graph in Figure 7.8 displays the recall values exhibited by the seven descriptors with Corel-1000 and Caltech dataset. The graph illustrates that the recall power of the SDPF descriptor has been improved with the fusion of innovative DDD descriptor.

The SDPF-DDD descriptor shows competitive recall power for several visual categories, including beach, building, dino, elephant, flower, mountain, food, plane, background, car_bg, and faces. It also indicates a stable recall power over all visual categories, even though the SDPF-DDD descriptor does not achieve the best recall for all the categories.



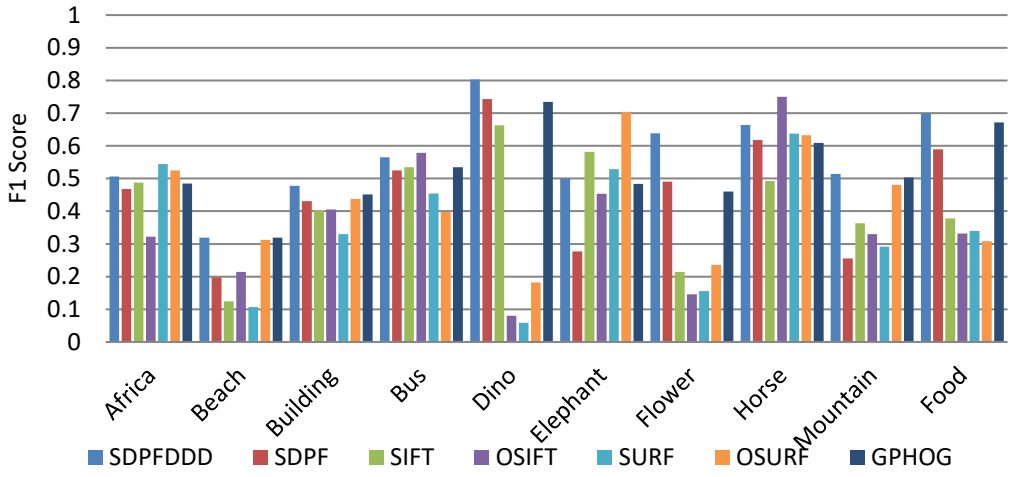
(a)



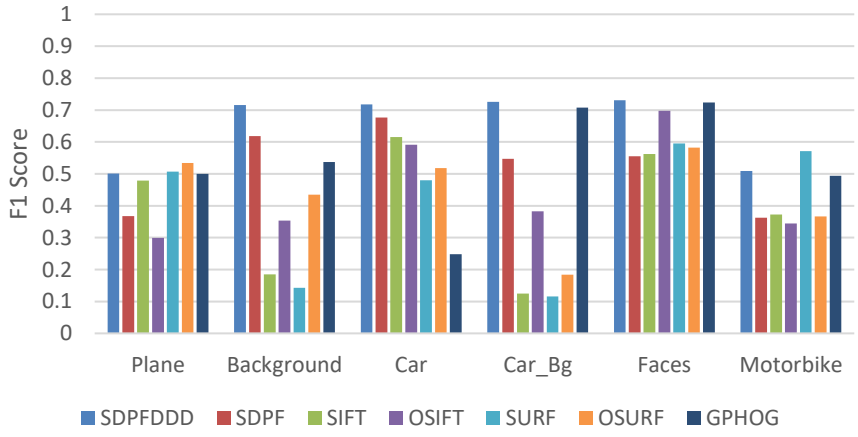
(b)

Figure 7.8: Comparison of recall; (a) ten visual concepts in Corel-1000 dataset, (b) six visual concepts in Caltech dataset.

Figure 7.9 illustrates the calculated F1 scores for the precision and recall values obtained for the seven descriptors with Corel-1000 and Caltech database. Although the SDPF-DDD descriptor does not show a significant F1 score for individual categories in the Corel-1000 dataset, it shows a significantly better F1 score for four out of six categories in the Caltech dataset. However, the average F1 scores of SDPF-DDD descriptor for both datasets achieves the highest, as displayed in Table 7.2, which implies the consistency of classification performance of the SDPF-DDD descriptor.



(a)



(b)

Figure 7.9: Comparison of F1 Scores; (a) ten visual concepts in Corel-1000 dataset, (b) six visual concepts in Caltech dataset.

Table 7.2 displays the averaged results obtained from seven experimental setups that were conducted using Corel-1000 and Caltech datasets. It is evident that the SDPF-DDD descriptor have indicated the best average result for all the criteria that are considered in the experiments. Although the Corel-1000 dataset includes many images that show a weak visual correlation within the visual category, the SDPF-DDD achieves around 66% of average recall. The images in Caltech dataset indicate a high homogeneity within the category; hence all the descriptors show a better classification performance compared to Corel-1000 dataset. The FC-GPHOG descriptor secure the second place in the list of best performing descriptors.

The results of the experiment show that SDPF-DDD not only improves upon the classification performance of SDPF but also outperforms or indicates a competitive advantage over the other popular visual descriptors while significantly solving many other challenging problems, such as the dimensionality of the descriptor, computation time and robustness to geometrical variances. Although the FC-GPHOG has achieved better and state-of-art classification performance with Caltech dataset in its original study, it is observed that the classification performance of FC-GPHOG decreases when the orientation, scale, or the spatial location of the visual content is drastically changed. When considering the dimensionality of the descriptor, the FC-GPHOG displays an order of magnitude higher than the dimensionality of SDPF-DDD, which implies the competitive advantage in the classification time.

Table 7.2: Averaged results obtained from the experimental setups of seven visual descriptors with the two datasets

Descriptor	Caltech			Corel-1000		
	p	r	F1	p	r	F1
SDPF-DDD	60.78	71.64	0.65	52.85	66.67	0.57
SDPF	50.38	55.56	0.52	41.59	55.33	0.46
BoF-SIFT	44.38	42.50	0.39	47.42	44.00	0.42
BoF-SURF	40.15	45.28	0.40	35.04	41.67	0.34
BoF-OSIFT	47.90	55.42	0.44	38.44	44.83	0.36
BoF-OSURF	50.95	49.03	0.44	45.55	49.67	0.42
FC-GPHOG	50.47	59.72	0.54	47.52	62.67	0.53

7.2.1 Assessing the classification performance of HSDPF

In this subsection, the results obtained by assessing the performance of the Hessian based SDPF (HSDPF) are compared with the existing best performing version of SDPF namely the SDPF-DDD, and other best performing methods. The results are presented separately due to the use of different datasets, which allows validating the performance of the descriptor with an extensive dataset. Further, the suitability of this improved descriptor has been validated for use in resource-constrained devices as well.

Figure 7.10 presents the performance of the descriptors by carrying out the classification task using Coil-100 and ALOI-View datasets in terms of average precision. Figure 7.11 depicts the performance recognizing the objects of the images from ALOI-Ill dataset. These images are differently illuminated, and the performance is measured by means of precision.

The experimental results in both the graphs in Figure 7.10 shows that HSDPF performs better than all the SIFT and SURF methods. Moreover, it even outperforms FC-GPHOG while SPDF-DDD shows closer results. None of the methods outperform CNN. The difference of the performance of HSDPF and CNN obtained for ALOI-View dataset is around 3.2%. However, the difference is negligible when the comparison of HSDPF and CNN was done with Coil-100 dataset as shown in Figure 7.10 (b). Coil-100 is less challenging dataset. Unlike ALOI-View, the objects in Coil-100 are fully visible without any covering or disturbance. This might be the reason for achieving slightly different results for Coil-100 compared to the results obtained with ALOI-View.

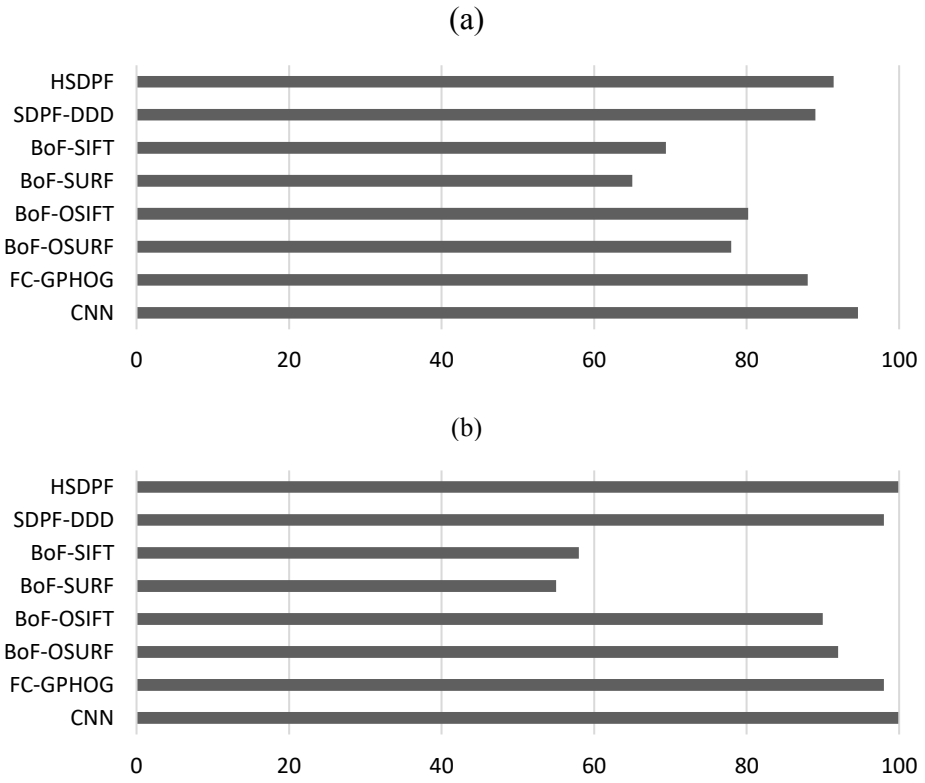


Figure 7.10**:** The performance of recognizing objects; (a) dataset: ALOI-View, (b) dataset: Coil-100 dataset, in terms of class average precision

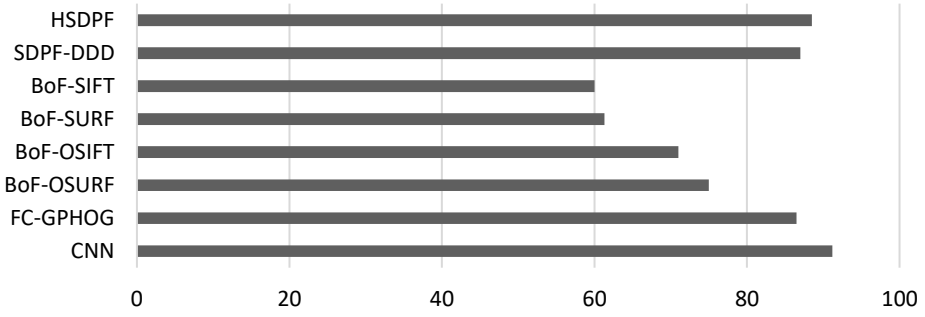
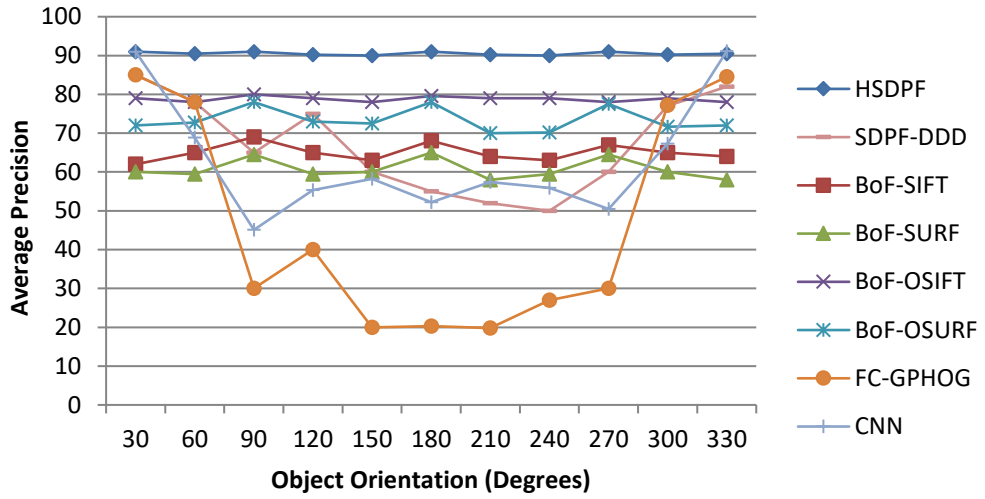


Figure 7.11**:** The performance of recognizing objects in ALOI-ill in terms of class average precision

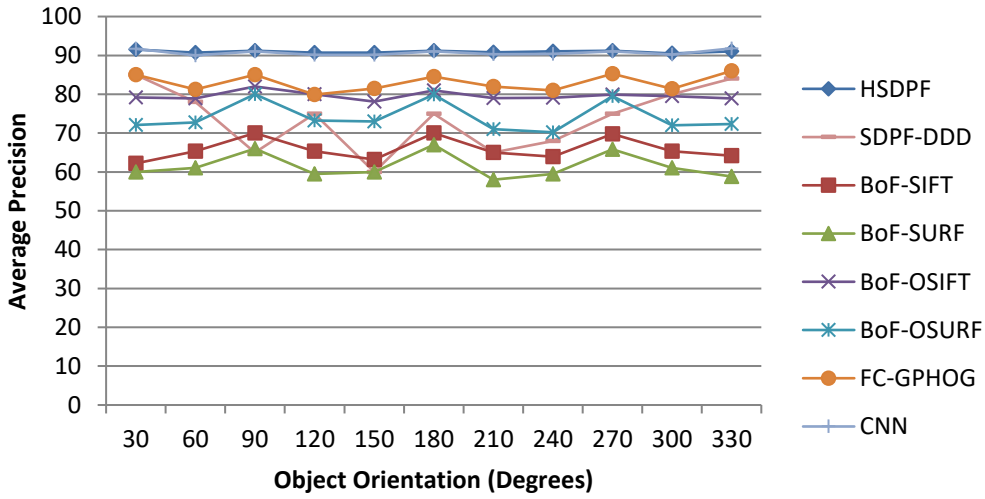
The illumination invariance of HSDPF was also assessed by using ALOi-ill dataset. Note that the original version of the SDPF has not been included as the second version SDPF-DDD already include the feature descriptor of the first version. Figure 7.11 shows that HSDPF outperforms the all SIFT, and SURF based descriptors, the previous version of SDPF which is SDPF-DDD and FC-GPHOG. The difference between the performance of HSDPF and CNN is around 2.7%. The overall results are

lower than the results obtained with the other two datasets. This might be due to the existence of many partially visible images in this dataset. Some examples of such images are shown in Figure 7.12. Although HSDPF does not outperform the CNN the difference is insignificant when the overall result is considered. Furthermore, HSDPF outperforms all the other methods which were known the be best performed handcrafted feature descriptors. Nevertheless, HSDPF is advantageous in many other ways compared to the CNN, which will be discussed in the next sub section. As per the current results, HSDPF is a highly discriminative descriptor with strong robustness to many different variations that are common in images such as different view angles and illumination. Furthermore, the chromatic and highly detailed and normalized shape description may be contributed to this achievement as the other descriptors are lack of having all the visual details together.

Figure 7.12: Sample images from five object categories in ALOI-ill dataset. Each column contains two images from a single object category, captured under different illumination conditions.



(a)



(b)

Figure 7.13**:** Performance of recognizing objects that are in different orientation in ALOI-View. (a) without data augmentation (b) with data augmentation

Figure 7.13(a) presents the performance of difference descriptors when the classifier is trained with non-augmented data. It is expected, the classification will be failed if the classifier has not seen many variations of the data in the training phase. However, Figure 7.13(a) proves that both SDPF-DDD and HSDPF are not affected by the unseen variations of the objects. Moreover, it clearly shows that the unseen variations of the objects significantly affected the performance of FC-GPHOG. The performance of CNN too is affected with the same fact. FC-GPHOG is based on the HOG descriptor,

which is typically robust to a few degrees of rotation in the object; hence the result is predictable. A robust set of convolutional features cannot be learned if the training data does not fairly represent the whole population of data. However, the SIFT and SURF based descriptors are not affected by these unseen variations of data. This might be due to both SIFT and SURF uses orientation and illumination normalization inside their feature description algorithms.

Figure 7.13(b) presents the performance of the classifier when trained with augmented data. The augmentation has been done by rotating the images by 0^o, 90^o, 180^o, and 270^o. The results reveal FC-GPHOG performed better when the classifier has been trained with augmented data. The results of CNN are also similar to the FC-GPHOG, whereas the other methods have not been affected significantly. However, the size of the dataset is increased with any kind of data augmentation. The size of the dataset adversely affects the training time. Therefore FC-GPHOG and CNN like methods are not favourable where a highly robust and computationally inexpensive performance is required. Moreover, HSDPF perform better than any other descriptor when the classifier has been weakly supervised. This implies that the HSDPF is more favourable where a highly robust and computationally inexpensive performance is required. The training of HSDPF will be quicker yet it will be robust to many unseen variations of the visual content.

7.2.2 Assessment of the cost of HSDPF

Table 7.3 shows the average computation performances of the descriptors and their selected dimensionalities. The effect of the dimensionality of the descriptors to the computational time is illustrated in the table below.

Table 7.3 Computational cost of the descriptors

Method	Dimension	Desktop PC			RPi	
		Extraction (ms)	Classification (ms)	Training (minutes)	Extraction (ms)	Classification (ms)
HSDPF	256	**10**	78.1	**45**	**15**	80.3
SDPF-DDD	128	12	77.1	**45**	16	**79.1**
BoF-SURF	**200**	380	77.2	**45**	530	79.8
BoF-SIFT	**200**	470	77.2	**45**	710	79.8
BoF-OSURF	600	1110	83.2	52	1530	90.7
BoF-OSIFT	600	1120	83.2	52	1610	90.7
FC-GPHOG	12000	2590	7772	122	3820	10230
CNN	253440	-	1362	335	-	38200

The results show that HSDPF is the fastest feature extractor regardless of the platform. The time spent for feature extraction by HSDPF on RPi is slightly higher than the time spent on the desktop PC. This gap is insignificant compared to the other comparable methods. Moreover, this insignificant gap can be considered as a solid proof of that HSDPF utilizes the hardware resources better than the other methods. The time taken by the classifier depends on many factors including the dimension of the descriptor. The results clearly show this dependency. Furthermore, the fact resolves the question, why HSDPF descriptor required more time in classification compared to the SDPF-DDD, BoF-SURF and BoF-SIFT. Nevertheless, HSDPF achieved the best overall time hence it is computationally advantageous compared to any other method.

It is noteworthy to mention that the dimensionality reduction method in HSDPF does not sacrifice the feature extraction efficiency as well. SIFT and SURF based descriptors can encode only the geometric/spatial information to the final descriptor. Although HSDPF descriptor is slightly higher in dimensionality compared to SIFT and SURF based descriptors, it includes the chromatic information in addition to the geometric/spatial information. The CNN used in this comparison consists of around 60 million of parameters. All these parameters must be optimized through the process

of training hence it is computationally expensive. Even the inference stage requires number of arithmetic operations not less than the number of parameters hence the prediction also computationally expensive. The result has proved it with the computation time spent on both the platforms. Moreover, the huge number of parameters make the model requires a huge space. Consequently, CNN spent additional 20 seconds to read and load the CNN model to the RAM prior to start the classification on RPi. The size of the model is getting increased with the complexity of the deep network. However, it has been explored that, for a better image classifier, CNN should maintain a sufficient complexity as well.

The dimension of the descriptor and the overall time of computation evident the superiority of HSDPF. The multiple advantages of HSDPF includes the ability of use with large dataset, real-time performance, and ability to run on resource constrained devices with similar performance to a typical desktop PC. Careful inspection over the algorithms of other descriptors, the strength of HSDPF has been inherited from the unconventional dimensionality reduction technique that is based on colour dithering, proposed in this study. An extensive measurement of the computational cost of every step in the HSDPF algorithm was conducted to identify the computationally expensive operations. This measurement can be helpful for further enhancement of the HSDPF algorithm for future applications.

Table 7.4. Computational details of HSDPF.

Description	Arithmetic operations per pixel/pattern						Execution Time (ms)
	add	mul	div	sqrt	cmp	tri	
ED-Dithering	3	9	0	0	3	0	3.51
Colour sorting	0	0	0	0	6	0	0.35
Calculate Hessian	6	3	0	0	8	0	1.87
Analyse Hessian	7	0	0	0	10	0	1.50
Non-max suppression	0	0	0	0	8	0	0.23
Centroid	2	0	0	0	0	0	0.17
Centroid distance	3	2	0	0	0	0	0.22
Distance bin ranges	0	0	0	0	1	0	0.01
Dominant orientation	5	2	0	0	0	0	0.28
Resolving upside down	6	1	1	0	1	1	0.85
SDPF angles	3	0	1	0	1	1	0.71
Descriptor construction	0	0	1	0	4	0	0.21

Table 7.4 presents the computational details against the algorithmic breakdown of HSDPF where *add* refers to addition/subtraction, *mul* refers multiplication, *div* refers to division, *sqrt* refers to square root, *cmp* refers logical comparison and *tri* refers to inverse trigonometric function. A trial of computing 100 HSDPF descriptors was used to obtain the execution time. The trial is run on the desktop PC using 100 images with the dimension 128 x 128. The precision of the operands used with the each of the operator is not mentioned in Table 7.4 though the majority is of single precision. Not all the operations are operated on every pixel. Precisely, any operation after obtaining the patters should operates on pattern level, which is an order of magnitude less on quantity compared to the number of pixels. The detailed analysis revealed that the computation found in any of the sub-operations has linear time complexity. Hence the overall algorithm also works in *O(n)* complexity, which helps maintain the speed of computation for images with different resolution consistently. Furthermore, HSDPF algorithm safely avoids many some arithmetic that are known to computationally expensive. This optimization helps the algorithm to work in real-time for images with average resolution. The detailed computation time analysis of shown in Table 7.4 revealed that the most computationally critical operation of HSDPF is the dithering

process regardless of using a faster implementation mentioned in [94]. The process can be further improved by implementing the same to compatible with parallel processing. However, the optimization must be verified through a proper experimental setup, as any such implementation only approximate the ED dithering that may not give the exact output of the sequential algorithm.

7.3 Experimental results of object segmentation

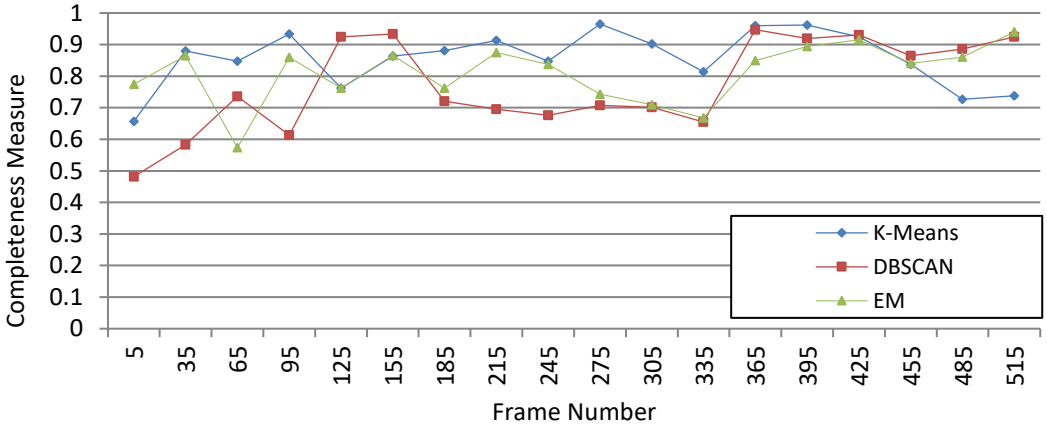
As the proposed object segmentation approach evolved through two major versions, both were evaluated separately, and then a comparison between them was made at the end. The next subsection discusses the results of evaluating the initial version, in which the object segmentation was conducted using the number of objects that have been predicted based on the quantity of SDPF points.

7.3.1 Segmentation with an estimated number of objects

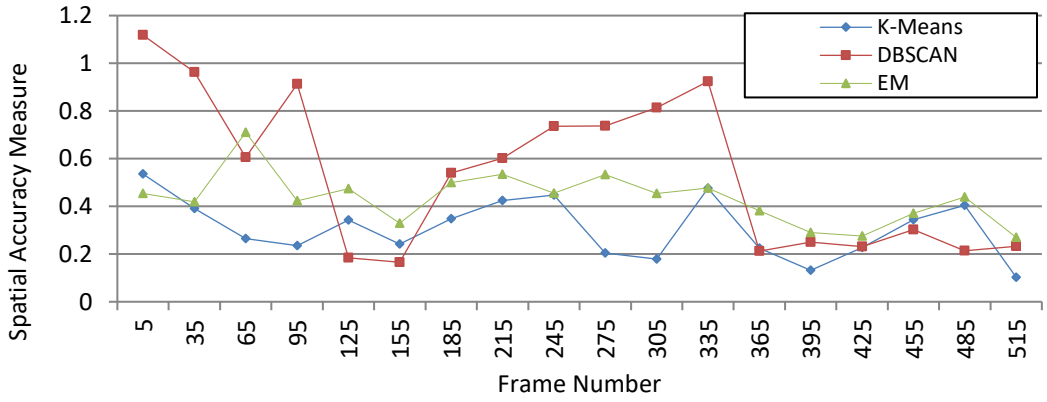
The experiment is setup to evaluate the performance of the segmentation with the estimation of the quantity of dominant objects in a frame using a video that contains both fixed and moving objects. These objects were selected from seven random categories provided in the dataset. Initially, the SDPF points are extracted. Then each of the points is clustered by using the selected clustering method. The distribution of SDPF points in a cluster is considered as a region of a single object. The cluster centres are computed and used them as the location of the corresponding object.

Figure 7.14 shows the performance of the three clustering algorithms evaluated by first few frames of the video sequence 16E5 that can be found in the CamVid dataset. According to the graph of Figure 7.14 (a), the K-means algorithm outperformed the other two for most of the frames in terms of completeness. However, the fluctuation of the completeness measurement of both EM and K-means methods shows a great analogy. Occasionally, DBSCAN also shows the highest completeness. However, the consistency of the performance is also important for using it as a reliable segmentation method. Nevertheless, DBSCAN failed over the other two in terms of spatial accuracy error, presented in the graph of Figure 7.14 (b). The lowest error can be seen with K-means at most of the frames. Although EM and K-means tend to perform closely, EM

should be trained repetitively if the number of clusters is changed, hence K-means is favourable over EM in terms of computation time. When the results are summarized K-means has achieved 86% as the mean completeness and 31% as the mean spatial accuracy for all the frames. The performance of the EM clustering is slightly less than the K-means clustering. EM clustering shows 43% of average spatial accuracy measurement and of 81% of average completeness. DBSCAN has achieved the worst result, with 77% as the mean completeness and 54% as the mean spatial accuracy. The overall results revealed that K-means is a better selection compared to EM and DBSCAN for clustering-based image segmentation when associated with SDPF points and the motion details in terms of completeness and spatial accuracy.



(a)



(b)

Figure 7.14: Comparison of the three algorithms with the sequence 16E5. (a) Completeness measure. (b) Spatial accuracy measure.

Figure 7.15 presents a sample frame that has been segmented by clustering SDPF points. The optical flow of SDPF options has also been visualized in Figure 7.15. It clearly shows the correspondence between the distribution of flow vectors and the SDPF point clusters. This correspondence proves that the proposed method takes advantage of temporal information of SDPF. In the example shown in Figure 7.15, the points on the road and fence are not properly clustered. It can also be seen that the colour of the road as well as the fence is not uniform and unique compared to their surroundings. The proposed method seems failing in the absence of discriminative chromatic information. Nevertheless, all the other objects have been properly segmented. The strength of the proposed method can further be seen by observing how it discriminates between the two car objects with the same colour yet at different depth in the scene.

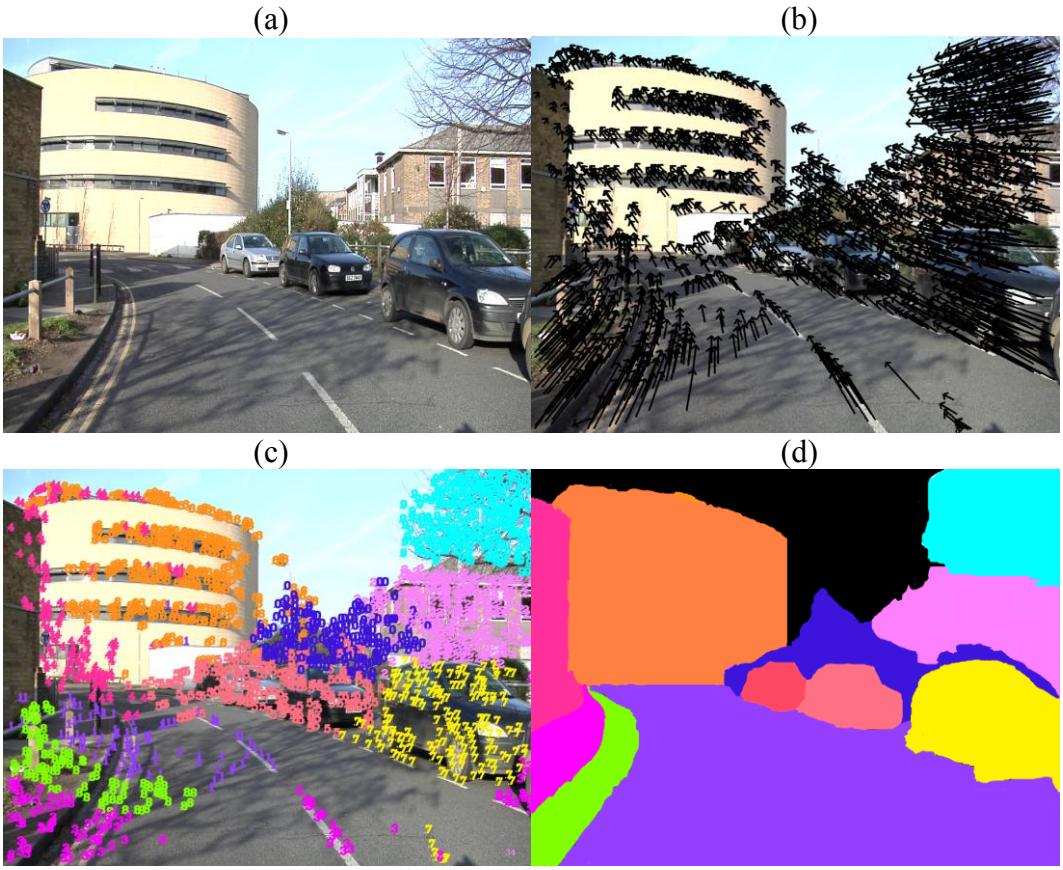
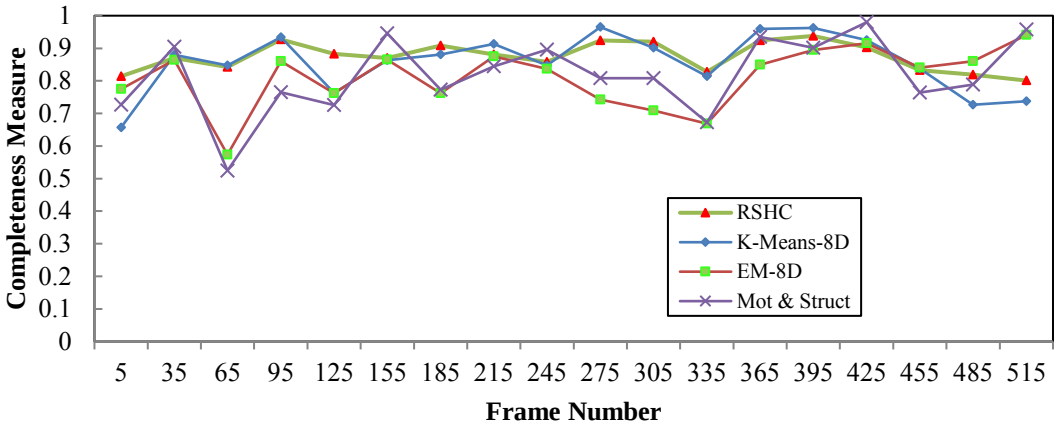


Figure 7.15: Clustering SDPF points in a video frame using K-means. (a) source frame (b) optical flow of SDPF (c) clustered SDPF (d) ground truth segments

7.3.2 Segmentation without estimating the number of objects

The same dataset and evaluation metrics were used for evaluating the performance of the second version of the proposed segmentation technique. The comparison has been made with the initially proposed segmentation technique, using the two best performing clustering techniques, i.e., the K-means and Expectation Maximization. Since the clustering of SDPF points was carried out a vector of eight elements (i.e., x, y coordinates, four colours in SDPF, flow magnitude and flow angle) the name K-means-8D is used to identify the K-means based approach, while the EM-8D is used to identify the Expectation Maximization based clustering approach. DBSCAN was not used in the comprehensive comparison due to its unacceptable performance. The second approach of segmentation, that uses Refinement Superpixels using HOOF and Colour cue, is referred by RSHC. Figure 7.16 presents Performance of RSHC, K-means-8D, EM and the method presented in [114], over consecutive frames in the sequence 16E5. The method presented in [114], is the only study based on motion and structure for video segmentation found so far. Since the method in [114] uses motion and structure information to generate point cloud before the segmentation, it is referred as Mot & Struct in the results and discussion.

(a)



(b)

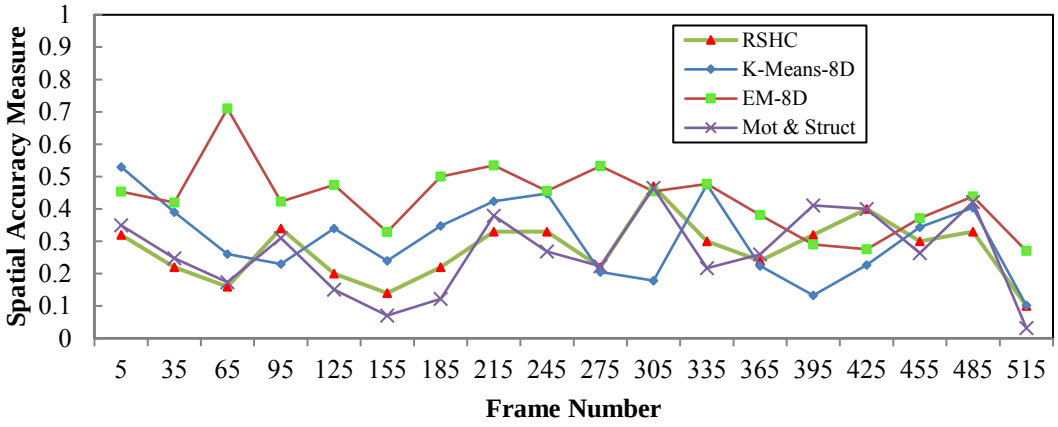


Figure 7.16: Performance of RSHC, K-means-8D and EM over consecutive frames in the sequence 16E5. (a) Completeness measure. (b) Spatial accuracy measure.

Figure 7.16 shows that RSHC is better than the previously proposed methods and Mot & Struct method in terms of overall completeness while closely compete with spatial accuracy error of Mat & Struct. The average results can be found in Table 7.5. The result shows that the improvement brought by the proposed method in terms of average completeness is insignificant. However, as Figure 7.16 indicates, RSHC performs consistently over consecutive frames. Although Mot & Struct method outperforms the other methods in terms of spatial accuracy measurement, the completeness is much lower compared to RSHC. The average completeness and the consistency together prove the superiority of the RSHC. The lower consistency of the other methods that

have been proposed previously might be caused by the weaknesses in the estimation of the quantity of dominant objects. In contrast, RSHC requires no prior knowledge of the frame. Hence a greater consistency can be expected. The result of clustering SDPF points in an example frame using the proposed methods is presented in Figure 7.17. to emphasize the quality of the performance of the new approach. Figure 7.17 shows that lack of information, such as the number of objects, leads to finding excessive number of false clusters by the previously proposed approach.

The lowest spatial accuracy makes the RSHC segments the objects with better precision. The improvement in the spatial accuracy error of RSHC can be attributed to the ability of accurate localization of the object boundaries by SLIC algorithm. Furthermore, the consistency of SLIC algorithm can be attributed to the use of weighted CIELAB colours, instead of utilizing equal responses from all three-colour components.

Table 7.5: The average completeness and the average spatial accuracy error of the segmentation with unknown number of objects

Method	**Average Completeness**	**Average Spatial Accuracy Error**
RSHC	87%	27%
K-means-8D	86%	31%
EM-8D	81%	43%
Mot & Struct [114]	82%	26%

(a) (b)

(c) (d)

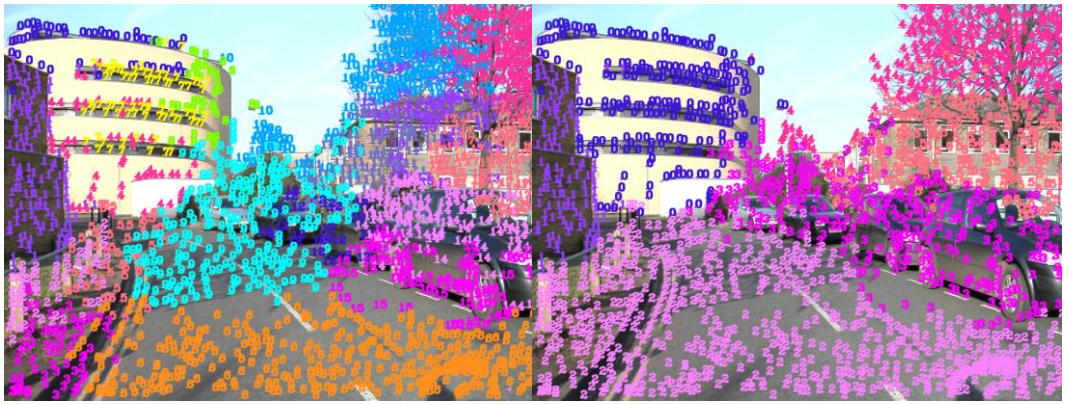


Figure 7.17: Clustering SDPF points in a video frame using the proposed methods. (a) Source frame. (b) manual segmentation (c) K-means-8D clustered (d) clustered with RSHC.

Once the objects are segmented with RSHC method, the individual clusters of SDPF can be used to obtain HSDPF descriptor to classify them among the object types. Since the HSDPF has already been evaluated with the individual objects, the experiment on recognizing individual objects over RSHC clustered HSDPF is not elaborated. However, the classification has been conducted to verify the performance of RSHC and HSDPF integration, and the results have been tabulated in APPENDIX E.

7.4 Evaluation of correspondence matching of NIN-RBM

The Browns dataset was used for evaluating the proposed local descriptor (NIN-RBM) in correspondence matching tasks. It comprises of local patches obtained from three image categories, namely Yosemite, Liberty and Notre Dame [111]. The evaluation

has been conducted comparing with the best performing methods from both handcrafted and learned descriptor categories. Both the real and binary-valued descriptors are selected from the two categories to compare trade-off in addition to the performance of the proposed descriptor, NIN-RBM. Table 7.6 illustrates the comparison of performance in terms of 95% error rate, which includes all combinations of pairs of categories that can be used for training and testing. In Table 7.6 *N* denotes Notre Dame, *L* denotes Liberty and *Y* denotes Yosemite where a pair of labels denote the trained set and test set, respectively. For an example *YN* denotes that "trained on Yosemite and tested on Notre Dame". Furthermore, Table 7.6 includes the dimension of all the descriptors to compare the matching time with the help of the details in remarks.

Table 7.6 includes the performance of descriptors from the categories given below.

1. **Real-valued-supervised:** the values in the descriptor are floating-point numeric, and they are learned using a supervised approach. The descriptors are typically assumed to be in Euclidean space.
2. **Real-valued-handcrafted:** the values in the descriptor are floating-point numeric and they describe handcrafted features. The descriptors are typically assumed to be in Euclidean space.
3. **Binary-valued-handcrafted:** the values in the descriptor are binary numeric, and they describe handcrafted features. The descriptors are typically assumed to be in Hamming space.
4. **Binary-valued-supervised:** the values in the descriptor are binary numeric, and they are learned using a supervised approach. The descriptors are typically assumed to be in Hamming space.
5. **Binary-valued-self-supervised:** the values in the descriptor are binary numeric, and they are learned using a self-supervised approach. The descriptors are typically assumed to be in Hamming space.

Table 7.6. Comparison of patch matching performance of NIN-RBM.

Method	Size	NL	YL	NY	LY	YN	LN	Average	Remarks
SIFT[37]	128	36.27	36.27	29.15	29.15	28.09	28.09	31.17	Real-valued, handcrafted
MatchNet[69]	4096	6.9	10.77	8.39	10.88	5.67	3.87	7.74	Real-valued, use supervised learning, high dimensional
DeepCompare[70]	256	4.85	7.20	4.10	5.00	2.11	1.90	4.19	
Binary L2-Net[83]	256	4.01	6.65	4.04	5.61	2.51	1.90	4.12	Binary-valued, use supervised learning
RFD[77]	598	19.35	19.40	14.50	16.99	11.68	13.23	15.86	
BinBoost[79]	64	20.49	21.67	18.96	22.88	14.54	16.90	19.24	
D-BRIEF[76]	32	51.30	53.39	46.22	47.29	43.96	43.10	47.54	
LDAHash[75]	128	49.66	49.66	52.95	52.95	51.58	51.58	51.40	
BRISK[72]	512	79.36	79.36	73.21	73.21	74.88	74.88	75.81	Binary-valued, handcrafted
BRIEF[71]	256	59.15	59.15	54.96	54.96	54.57	54.57	56.23	
ORB[25]	256	59.15	59.15	54.96	54.96	54.57	54.57	56.23	
DeepBit[57]	256	32.06	34.41	63.68	57.61	29.60	26.66	40.67	Binary-valued, self-supervised: no ground truths needed
DBD-MQ[104]	256	31.10	33.11	57.24	57.15	27.20	25.78	38.59	
NIN-RBM	256	30.10	32.95	54.25	54.12	27.35	22.19	**36.83**	

SIFT [37], is a key-point detector and a descriptor that falls into the *real-valued-handcrafted* category. It detects blob-like features by computing difference of Gaussian (DoG). Then the surrounding to the detected features is described by using histogram of oriented gradients (HOG). The HOG in SIFT typically contains 16 histograms, each with eight orientation bins. Hence a single descriptor contains a total of 128 floating-point values. Although this specific number is used in the original implementation of SIFT [37], it can be further elevated based on the application. HOG descriptors can be compared using Euclidean distance. However, calculating Euclidean distance in floating-point precision requires an exceedingly long time in computation, which worsens when there is an excessive number of key points, as the similarity should be calculated exhaustively. MatchNet [69] falls into the *real-valued-supervised* category. It is a deep neural network that consists of four convolution layers, then a bottleneck layer and, at the end, three fully connected layers. The first four layers learn the features out of the given dataset. The dimensionality of the feature responses that are output from the 4^{th} convolution layer can be reduced by using the bottleneck layer. The fully connected layers attached to the last feature extraction

layers of the network directly output the pairwise patch similarity. The first layer of the last three fully connected layers in MatchNet uses an extremely high dimensional vector that has been output from its previous layer. As a result, the computation in the last three layers requires enormous computation power. Not to mention that these operations should be done in floating-point precision as well. Moreover, MatchNet consists of a model that is built according to a Siamese-like architecture [115]. Siamese network consists of two towers of network; typically, two instances created from a single network. Generally, a network with Siamese like architecture should be trained with supervision, as it is necessary for MatchNet as well.

DeepCompare [70] is much similar to the architecture presented in MatchNet. The presence of an additional decision network makes it different from the MatchNet. However, the number of parameters used in the two models is also slightly different, although the decision network makes DeepCompare outstand in terms of 95% error rate. Furthermore, Table 7.6 shows that DeepCompare is significantly better than all other methods except for Binary-L2-Net. The other advantage of the DeepCompare is the lower dimensionality of the final descriptor. The dimensionality of the output descriptor profoundly affects the similarity matching speed. However, DeepCompare is also a DNN-based approach, and the computation within the inner layers of the network is expensive. Moreover, it may be limited using real-world applications, due to the inherent problems from supervised learning, such as the lack of generalizability and unavailability of annotated data. The decision network in DeepCompare also computes the output by using the real-valued descriptors. Hence the patch similarity calculation will be computationally much expensive.

The third section of the rows in Table 7.6 use supervised learning. The results from the LDAHash [75] and D-BRIEF [76] descriptors are less favourable contrasted to the other descriptors in the same section. Both the descriptors project a real-valued handcrafted feature vector that resides in Euclidean space into a binary vector that resides in Hamming space. Both methods learn the projection using a sample dataset. However, they inherit the adverse performance from their real-valued handcrafted features. BinBoost, RFD, and Binary L2-Net [83] achieved better performance

compared to all self-supervised methods. Hence those methods outperform the proposed approach, NIN-RBM. The supervision used in these methods causes their superiority in describing the patches using binary codes. The supervision introduces the domain dependency; hence the applications are extremely limited in case of lack of annotated data. Although NIN-RBM shows poor performance compared to Binary-L2, it significantly outperforms any of the handcrafted descriptors such as ORB and BRIEF that are successfully used in real world applications to date. Hence, the performance of NIN-RBM is still sufficient for real world application and it will further improve the quality of the output as well. On this ground, NIN-RBM is more beneficial than these supervised learned binary descriptors when no annotated data exists.

The last two sections of rows in Table 7.6 consists of unsupervised or self-supervised binary descriptors. Out of these descriptors, BRIEF [71], BRISK [72], and ORB [25] are unsupervised and handcrafted. The remaining two descriptors, i.e. DeepBit and DBD-MQ [104] are trained by using self-supervised techniques. The simple intensity comparison approaches used in BRIEF, BRISK, and ORB do not outperform most of the methods used for comparison, including the other methods in the same section. DBD-MQ relies on multiple auto encoders, to learn features and a function for the binarization. DeepBit [57], in contrast, utilizes a rigid sign function without considering the data distribution, to compute the binary descriptor. The proposed method, NIN-RBM significantly outperforms both DBD-MQ and DeepBit in terms of 95% error rate. It could be further seen that NIN-RBM performs well at the majority of train-test cross-category configuration compared to the other two in the same section.

The proposed method outperforms the self-supervised binary descriptors that have been compared for most of the train-test cross-category configurations. However, it is noteworthy to mention that DBD-MQ shows an exception with a slightly better result for testing images from the Notre dame subset when the model is trained using the images from Yosemite. Nevertheless, NIN-RBM signifies its better performance for all the other cross-category combinations.

The proposed descriptor, NIN-RBM can reduce the quantization error of the hidden unit of the RBM through the contrastive divergence algorithm. This approach preserves and forwards the power of discrimination in the real-valued features, obtained from the NIN to the binary-valued descriptor through the RBM. This is the main difference between NIN-RBM and the remaining descriptors in the same section of Table 7.6. Furthermore, the proposed method has a fine-tuning step that directs the proposed NIN-RBM hybrid network to train with invariant features, without freezing the pre-trained parameters of the NIN component. This step of the proposed learning method can cope with various forms of geometric variations. DBD-MQ and DeepBit also attempt learning geometrically robust encoding schemes. However, this objective has been set at the optimization algorithm that is utilized for learning the function for mapping the features to binary codes through the RBM network. Hence the deep learned convolutional features from the VGG network are never optimized for the local patch matching task. Moreover, standard convolution features are less robust to geometrical variations such as rotation [57]. Therefore, the features that are learnt through a generic learning algorithm through a conventional CNN may have contributed to the greater error rates shown with DeepBit and DBD-MQ. This difference makes the NIN-RBM response better to the patches with abrupt geometrical variations where the DBD-MQ and DeepBit might be failing.

The results of experiments that have been conducted using Brown dataset are presented in Figure 7.18. The proposed method, NIN-RBM has detected all the pairs of images in Figure 7.18 to be similar. However, the pair at the last column is an incorrect match as described in the dataset. The correctly matched samples revealed the robustness of NIN-RBM under several variations, as mentioned below.

1. First column: robustness under blurring
2. Second column: robustness under slight illumination variations
3. Third column: robustness under drastic variance in viewpoint

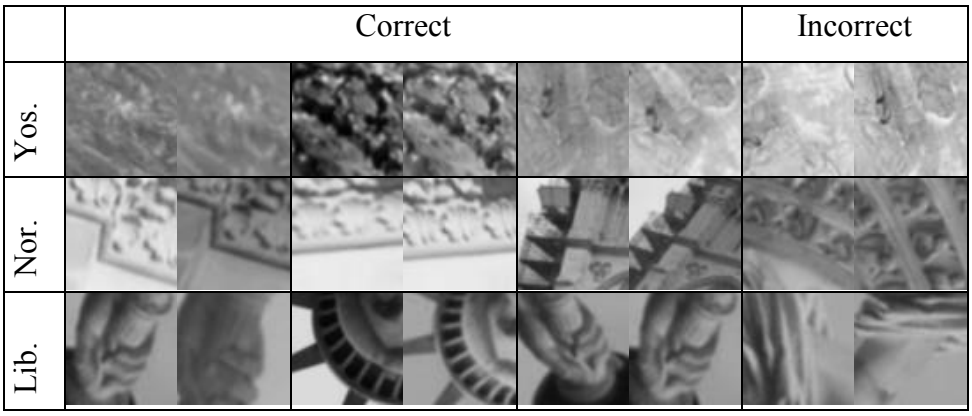


Figure 7.18: Correct and incorrect output of NIN-RBM for Brown Dataset. First row shows four sample pairs of matched patches from Yosemite, the second one from Notre Dame and the last row of is from liberty subsets.

In the last column, the first pair of images share similar visual content. However, the content seems different due to extreme geometrical differences such as translation and rotation. The dataset has annotated them as a mismatching pair, regardless of the existence of a greater intersection of visual similarity. This problem can be observed in the second mismatching pair as well. The two images have some architectural similarities, although the content appeared to be oriented in different directions. The two images in the last row are also not different from the first two, as it also has a bit of an intersection in visual content. The only visible difference can be noted as the contents are on slightly different scales, but with a significantly different orientation.

The problem persists with the false matching pairs shown in Figure 7.18 seems to be a side-effect of the fine-tuning process used in NIN-RBM. The purpose of the fine-tuning phase is to abstract further the features obtained from the pre-trained session to make them robust to many geometrical variances. However, it is an undeniable fact that overly abstracting the visual data reduces the discriminative power of the descriptor. The fine-tuning has been carried out using samples with heavily augmented inputs. This process may overly abstract the proposed descriptor, as it diminishes the discriminative power required by some pairs of the images in this specific dataset.

Figure 7.19 presents the receiver operating characteristic (ROC) curves of the binary-valued descriptors used for the comparison. This ROC curve was obtained by using the all cross-category training-testing configurations of the patches from the Brown dataset. The proposed binary descriptor exhibits an excellent performance consistently

over all the cases. As per Figure 7.19, it is evident that NIN-RBM is better in balancing the robustness and discriminative power throughout the different configurations in the dataset. Most of the patches that are annotated as dissimilar shares very weak visual similarities, in both Notre dame and Liberty datasets. This nature of the two datasets makes it less challenging for most of the descriptors in terms of discriminability. Yosemite dataset, in contrast, include many image patches that are annotated as dissimilar yet share a strong visual resemblance. Hence Yosemite dataset seems challenging for all the binary descriptors, including NIN-RBM. This can be clearly observed in Figure 7.19, as all methods show comparatively higher error rates where the Yosemite dataset has been used. Moreover, the majority performs better with the remaining two datasets. Furthermore, Figure 7.19 denotes that most of the handcrafted binary descriptors could maintain very low false-positive rates compared to the learned binary descriptors, including NIN-RBM. This implies that learning-based descriptors, including the proposed approach, are better with smaller Hamming radius used in similarity matching.

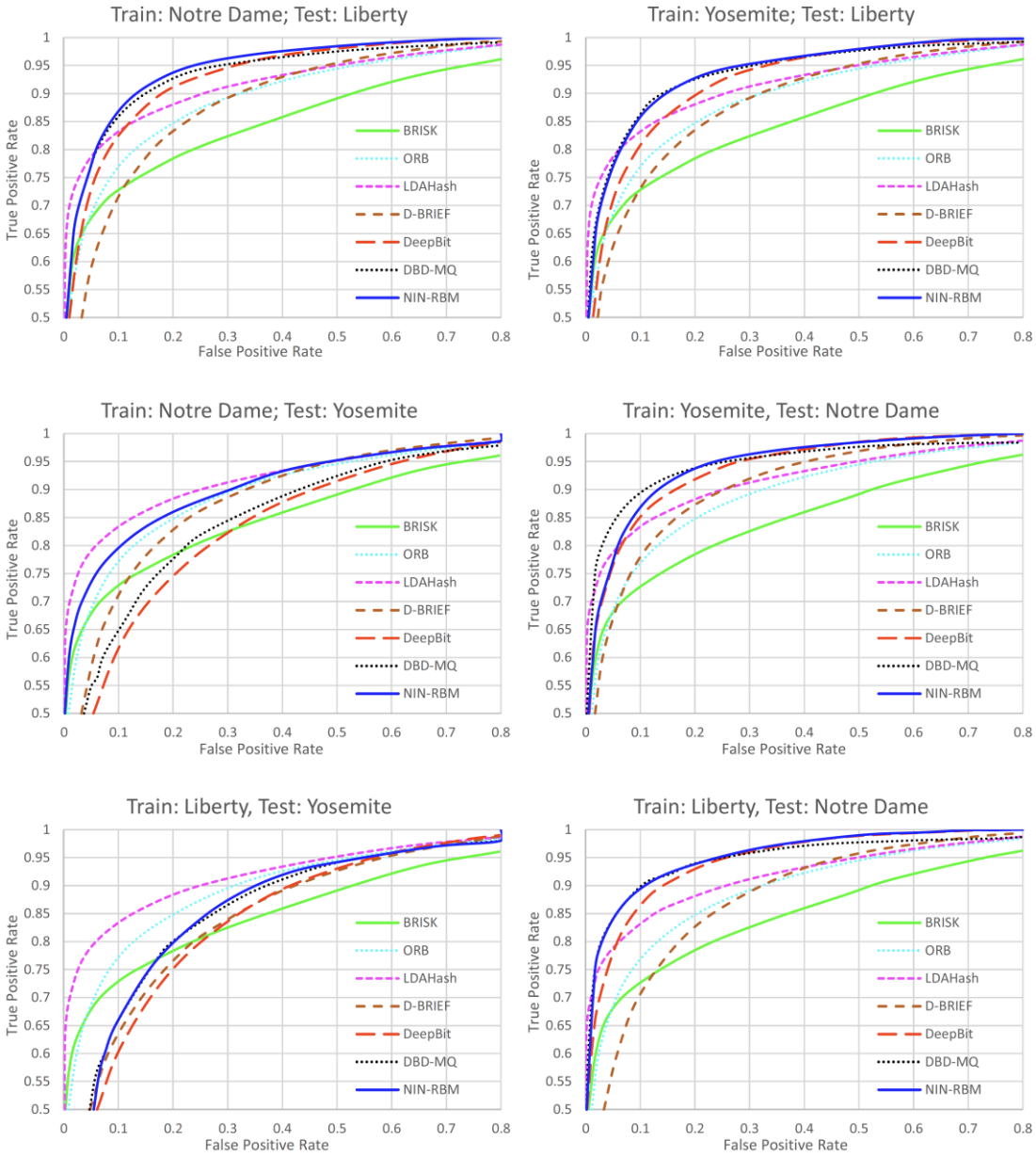


Figure 7.19: ROC of binary-valued descriptors with all combinations of all cross-category training and testing configurations with Browns dataset

In brief, the results presented in Table 7.6 demonstrate that the proposed method outperforms all binary descriptors that are in the self-supervised section in terms of 95% error rate. Figure 7.19 further reveals that NIN-RBM performs consistently over different datasets and is better in similarity matching when a smaller Hamming radius is used.

The performance of NIN-RBM applying for correspondence matching when colour information exists is assessed using RomePatches dataset. Table 7.7 presents the

results that are obtained by performing the experiment as specified in [55]. In this experiment, 1000 feature points were used for querying target features among 9000 feature points. The comparison of the proposed binary descriptor has been made with both the binary-valued and real-valued descriptors in terms of mAP.

AlexNet-conv5 [54] and CKN-grad [55] are real-valued descriptors. These methods use learning techniques to obtain their feature set. CKN-grad and SIFT outperform all the other descriptors used for the comparison. This outstanding performance can be attributed to the descriptors being real valued, that is known to be able to encode extremely discriminative information. Both CKN-grad and AlexNet-conv3 depend on convolution features. However, CKN-grad creates an extremely higher dimensional descriptor compared to the output of AlexNet-conv5. Generally, more discriminative information can be encoded with higher dimensional descriptors. This may be the reason behind the better performance of CKN-grad compared to AlexNet-conv5.

FREAK [73] and BRISK [72] are binary descriptors that are based on handcrafted features. As both the descriptors are not capable of encoding colour information, the colour information that exists in this specific dataset should not affect the performance of the two descriptors. Both FREAK and BRISK do not handle variances in the orientation of the content. However, ORB [25] uses a reorientation technique for both FAST detector and BRIEF descriptor that results in achieving better orientation invariance. On the other hand, the RomePatches includes local patches extracted out of images that were captured from drastically different viewpoints. These facts made ORB outperforms all the handcrafted binary-valued descriptor for this dataset. It is noticeable that ORB performs better on a ground that it inherently cannot utilize the colour information provided in the dataset.

The performance of DeepBit is better than all the handcrafted binary-valued descriptors. However, all the real-valued descriptors have performed better than DeepBit. The generic features that are obtained for natural image classification are used in DeepBit to obtain the binary descriptor. Hence it is limited in handling the geometric variances through its binary description that were optimized for the geometric invariance purpose. The patches with dramatic geometric variance found in

the dataset, together with the unoptimized visual features, do not compete with majority of the other descriptors. In contrast, NIN-RBM relies on features that are further optimized for binary description, together with the feature to binary mapping. This factor is reflected adequately in the results presented in Table 7.7, where it outperforms all the other methods. It is noteworthy that NIN-RBM outputs a binary descriptor of 256 bits (32-byte) yet performs better than existing methods, either with the same or double the binary code lengths. Furthermore, the proposed descriptor outperforms AlexNet-conv5 descriptor, although it cannot compete with the remaining real-valued descriptors. However, the intention of proposing NIN-RBM is to be used in applications that requires extremely fast correspondence matching, hence the poor performance of NIN-RBM compared to the real-valued descriptors does not limit its applicability. As many of the binary-valued descriptors are currently used in real-world applications, replacing them with NIN-RBM will significantly improve the quality of the output. Furthermore, NIN-RBM is advantageous in matching colour images, due to its extremely lower dimensionality and being a binary descriptor that shows its significance in the patch matching process.

Table 7.7 Results of correspondence matching task with RomePatches.

Descriptor	Bytes	mAP	Remarks
CKN-grad[55]	1024	88.10	Real-valued and extremely high dimension
SIFT[37]	128	87.90	
AlexNet-conv5[54]	256	49.60	
FREAK[73]	64	23.26	Binary-valued and very low dimension
BRISK[72]	64	31.95	
ORB[25]	32	43.83	
DeepBit[74]	32	46.97	
NIN-RBM	32	**50.50**	

7.4.1 Evaluation of instance retrieval

In this section the assessment of NIN-RBM by applying it in instance retrieval task, is elaborated. Three datasets namely, Holidays, Paris and Oxford datasets were used in this evaluation.

Initially, a set of key-points were detected using FAST [116] key-point detector for each of the images in four different scales. Then a local image patch was obtained for each key-point from the image. This process enables extracting salient patches from the images and recording the true-similar pair by using the correspondence information that was provided with the dataset. Then the binary codes for each patch are obtained using the binary description method. The similarity measurement was done by measuring the hamming distance. Brute-force matching was used to obtain the best similarity between the pairs of images. The proposed descriptor was prepared for this experiment by obtaining the pre-training parameters utilizing ImageNet ILSVRC2012 dataset. However, the RBM component was trained using the landmark dataset. Furthermore, the fine-tuning process of the feature extraction component was also carried out by using the same set of images from the Landmark dataset that were taken for the training of the RBM component. These images were not used for testing. Table 7.8 presents the result of NIN-RBM comparing with two real-valued descriptors that are based on SIFT and six deep learned descriptors. The second-row section includes three real-valued deep learned descriptors. The third-row section includes several different versions of three binary-valued deep learned descriptors.

SIFT-BOW [109] is a bag of word-based methods that initiate the process with SIFT key points and descriptors. SIFT-IFV [117] is also based on SIFT key points, but it utilizes a fisher model in preparation for the descriptor. When comparing the two variants, SIFT-IFV [117] generally outperforms the bag-of-word (BOW) based descriptors [117]. The fact is shown in the first two rows of Table 7.8. However, the two methods do not outperform the proposed descriptor for any of the subsets in the dataset.

The second section of the rows in Table 7.8 includes several real-valued descriptors. CNN+aug+ss [118] is a descriptor that has been formed using CNN with augmentation; the “ss” stands for “spatial search,” which is a technique used for obtaining local patches. Out of the three descriptors, CNN+aug+ss rarely outperforms the remaining two. It is even comparable with the proposed method by slightly outperforming for some subsets of the dataset. DF.FC1 [119] is based on deep learned

features. It consists of a deep model and several fully connected layers to generate the descriptor. It has been extended by a refinement technique and has been proposed as ReDSL.FC1 [119]. Nevertheless, both descriptors are extremely large in size due to the excessive number of parameters that are contained in their deep neural mode.

The third section of the rows in Table 7.8 shows the performance of binary descriptors. The first six descriptors are different variants from three deep learned binary descriptors, namely ITQ [120], [121], neural code (NC) based principal component analysis PCA (NC-PCA) [105], and DeepBit. ITQ uses an iteratively quantizing technique to obtain the binary descriptor from deep learned real-valued feature descriptor. Furthermore, it is based on an optimization technique that thoroughly depends on data, to mitigate the error in quantization that is performed in the process of binarization. Nevertheless, it does not outperform NIN-RBM, although it shows some advantages over the handcrafted real-valued descriptors. NC, with PCA, outperforms the ITQ method. The binarization is done by applying PCA compression over neural codes. Both ITQ and NC-PCA are based on learned features using a CNN similar to the process in DeepBit. However, Table 7.8 illustrates that DeepBit outperforms the other two, which can be a result of the usage of a more successful CNN model called VGG and the three objective functions used in the optimization of DeepBit binarization that balances the even distribution, constructability, and robustness. However, the lack of the ability to learn geometrical invariant properties has been inherited to DeepBit from CNN. As observed in Table 7.8, NIN-RBM outperforms DeepBit for most of the datasets. Although the proposed method depends on a deep learned feature, the mlpconv-based feature learning technique, along with the fine-tuning approach, could improve the robustness of the features.

Table 7.8 shows that NIN-RBM on the FAST-based patches significantly outperforms all binary descriptors that are based on the deep learned feature. Furthermore, binary codes of 256-bit length perform better than the other versions of the proposed descriptor. NIN-RBM outperforms CNN-aug-ss with Paris dataset in terms of mAP. The evaluation of DF.FC1 and ReDSL.FC1 [119] with Oxford and Paris dataset shows significantly better performance over the entire remaining methods. This is mainly because of the two descriptors are being real valued. Moreover, their data dependency

also may positively contribute to the performance. However, the binary descriptors, including the proposed method, are still advantageous in the matching phase due to quicker calculation of Hamming distance, in contrast to the calculation of Euclidean distance for real-valued descriptors.

Table 7.8 Performance over instance retrieval task with Holidays, Oxford, and Paris in terms of mAP%.

Method	Paris	Oxford	Holidays	Remarks
SIFT-BOW[109]	46.0	36.4	54.0	Real-valued high dimensional
SIFT-IFV[117]	-	41.8	62.6	
CNN+aug+ss[118]	79.5	68.0	84.3	Real-valued high dimensional and larger models
DF.FC1[119]	86.8	46.5	-	
ReDSL.FC1[119]	94.7	78.3	-	
ITQ 256 [120], [121]	66.3	48.9	67.1	Binary-valued descriptors low dimensional
ITQ 512 [120], [121]	66.8	50.8	3.9	
NC+PCA 256 [105]	-	55.7	78.9	
NC+PCA 512 [105]	-	55.7	78.9	
DeepBit 256 [74]	82.5	60.3	81.8	
DeepBit 512 [74]	82.9	62.7	82.7	
NIN-RBM+FAST 128	79.5	61.2	80.0	
NIN-RBM+FAST 256	**83.2**	**64.5**	**83.8**	
NIN-RBM+FAST 512	**83.3**	**64.7**	**83.8**	

In summary, real-valued handcrafted descriptors do not outperform any of the real-valued learned descriptors and most of the learned binary-valued descriptors, such as NIN-RBM, ITQ, DeepBit and NC-PCA have achieved comparable results but they have not outperformed NIN-RBM. The difference between the above three descriptors and the proposed method can be summarized as follows.

- The three descriptors and NIN-RBM rely on features learned for general image classification.
- The features of the three descriptors are learned through convolutional kernels.
- The features of NIN-RBM are learned through mlpconv layers.
- None of the three descriptors optimize the features but only concerns optimizing the binary codes by freezing the learned features.

- NIN-RBM optimizes both the binary coding and features simultaneously.

These differences may support gaining better performance over the patches that have more considerable geometrical differences. A significant improvement can be obtained by increasing the dimension of NIN-RBM descriptor from 128-bit to 256-bit. The 256-bit version of NIN-RBM indicates its best performance where the performance improvement by higher dimensions is not significant. Hence 256-bits is the optimal dimension for NIN-RBM.

7.4.2 Tasks from the HPatches benchmark

It has been observed that there may be certain inconsistencies in the results obtained for the dataset with many different evaluation criteria and metrics [112]. In order to assess the performance consistently without any bias from the datasets and evaluation metrics, evaluation protocols given in [112] were used. The protocols have been defined with a dataset called HPatches [112]. Hence, NIN-RBM was benchmarked with the same dataset. The protocols provided with HPatches benchmark scheme are well defined and strict to evaluate the performance correctly. The comparison has been made by comparing the performance of the proposed descriptor with several state-of-the-art self-supervised and handcrafted binary descriptors. In addition to the binary descriptors, several best performing real-valued descriptors were also used to evaluate the trade-off of NIN-RBM as it is a low-dimensional binary descriptor. HPatches benchmark is carried out on three different tasks as mentioned in [112]:

1. Matching
2. Retrieval
3. Verification

The first task uses a reference image and then measures the performance by matching it with the target image.

The discriminative power of a descriptor can be measured by the verification task, in which a descriptor is used to discriminate between positive and negative pairs of patches. This task consists of the following two sub-experiments.

1. Intra-sequence verification: the verification is carried out using pairs of patches from the same patch sequence.
2. Inter-sequence verification: the verification is carried out using pairs of patches from different patch sequences.

HPatches dataset consists of many sequences of image patches. There is an excellent correlation among the patches within the sequence, as they have been extracted from a single image; as a result, it is typically less challenging in the inter-sequence verification task.

The retrieval task focused on measuring how well a descriptor retrieves image patches, similar to a query image patch from an extensive collection. The performance measurements of all three tasks, including the two subtasks, are recorded in terms of mAP percentage. None of the descriptors that describe learned features have utilized the patches from the HPatches dataset for the training purpose; hence all the testing images were never seen by any of these models.

Figure 7.20 shows the results obtained for all the tasks by the nine binary descriptors, including NIN-RBM and four real-valued descriptors. BRIEF [71] and ORB [25] are handcrafted binary descriptors where DeepBit [74] and DBD-MQ [104] are deep learned binary descriptors. According to Figure 7.20, NIN-RBM outperforms all the binary descriptors in all three tasks. Moreover, it outperforms SIFT [37], which is a real-valued handcrafted descriptor in the patch verification task. Images in the *tough* category generally include extreme geometrical noise. The results indicate that the proposed descriptor and the other binary descriptors could not perform well with the *tough* category, compared to the other real-valued descriptors. However, NIN-RBM outperforms the remaining binary descriptors in the *hard* category, which includes patches that consist of a significant level of geometric and illumination variations. The three real-valued learned descriptors, namely, SKAR [122], SOSNet [123] and GeoDesc [124], outperform both the real-valued and binary-valued descriptors on HPatches dataset.

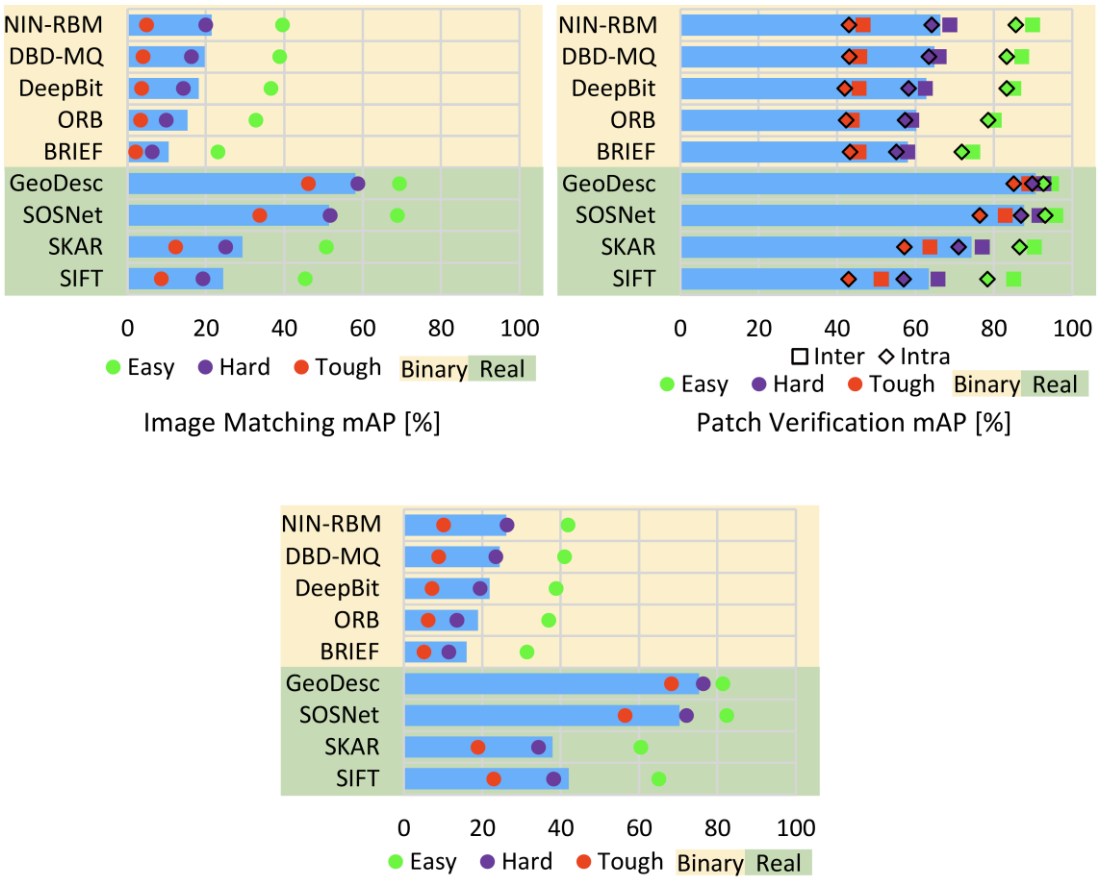


Figure 7.20 Performance over the three tasks namely retrieval, verification and matching. The coloured circular bullets indicates the performance over the three different challenging levels found in HPatches dataset. binary-valued or real-valued categories are denoted with different backgound colours.

As per Figure 7.20, these three descriptors perform better with data in all three noise levels. It is a substantial benefit over the remaining binary descriptors. The results obtained for the patch verification tasks differ from the remaining experiments due to the binary descriptors showing comparable results to several best performing real-valued descriptors. The proposed method even outperforms SKAR [122] for the *easy* category. Moreover, the performance seems consistent over both the cases where the patches belong to the same or different sequence. This result implies that binary descriptors are equally favourable in applications which requires verification of correspondence matching where no dramatic geometric and illumination variances exist.

7.4.3 Evaluation of the cost of computation

The computational efficiency of NIN-RBM has been evaluated in terms of model size, and the time is taken for encoding the input patch to its descriptor. The model size was compared in terms of the number of parameters that are included in the deep neural network and the actual size the model acquires in a disk. The computational cost of matching two descriptors partially affected by the precision of elements, i.e., real-valued or binary, the length or the dimensional of the descriptor and, finally, the choice of similarity metric. In this comparison, all the binary descriptors typically use Hamming distance, where the real-valued descriptors use Euclidean distance-based similarity measurement. Since the measuring metric is similar to the category (real-valued or binary-valued), the speed of similarity matching is proportional to the length/dimensionality of the descriptors. Therefore, the matching speed of the proposed binary descriptor was not compared. Instead, the lengths of the binary codes and other real-valued descriptors are reported. Table 7.9 presents the storage requirements of the feature extractor components used in this experiment.

Table 7.9 Model size and number of parameters of the base models.

	AlexNet	VGG16	NIN
Model size	228 MB	537 MB	29 MB
Number of parameters	57 M	134 M	7.3 M

Table 7.9 establishes that the NIN component used in NIN-RBM is exceptionally lightweight, contrasted to the AlexNet and VGG16 that are commonly used in other methods. VGG16 is one of the most used models due to its exceptional natural image classification capabilities without considering the complexity of the model. Although the closest model to the NIN, in terms of the model size, is AlexNet, the number of parameters in NIN is around one-eighth of the number of parameters in AlexNet. This is a huge benefit in the case of hosting the proposed model in a resource-constrained device. Further, the small storage space requirement typically affects the time taken for loading of the model to memory at the initial launching as well. In addition to the NIN component, the proposed method uses an RBM which is a single layer network.

The RBM component used in this method consists of a fully connected layer compared to the global average pooling layer that can be found in the existing implementation of NIN. This addition introduces only around eight hundred thousand more parameters that are roughly equal to 3 megabytes. Therefore, the effect from the NIN-RBM overall model to the existing space complexity of the NIN is negligible. The number of mathematical operations that have to be performed in inferring is proportional to the number of parameters. Hence Table 7.9 further implies that the process of inference of the proposed method utilizes fewer computer resources due to an exceptionally smaller number of parameters. Hence, NIN-RBM is significantly less expensive, compared to the other descriptors in terms of cost of feature extraction and description.

Figure 7.21 portrays the time taken for encoding the input image patches to their descriptors. 1000 randomly selected images from CIFAR-10 dataset were used. The experiment has been conducted in both the CPU and CUDA enabled GPU separately. The experiments were conducted on a computer with the following specification.

- CPU: Intel Core i7 6700HQ
    - 2.6 GHz of a base frequency (no overclocking done during the experiments)
    - 4 physical cores
    - 2 Logical cores per physical core
    - 6 MB cache
    - 8 GT/s bus speed
- GPU: Nvidia GTX970M
    - Nvidia Maxwell architecture
    - 1280 CUDA cores
    - Base clock 924 MHz (no overclocking done during the experiments)
    - 3 GB 2500 MHz GDDR5 memory with 120 GB/sec bandwidth
    - 192-bit memory interface width

In this experiment, both CPU and CUDA implementations of SIFT were used in the comparison. The CUDA based GPU version of SIFT (GPU-SIFT) was implemented based on the specification given in [125]. PCA-ITQ was obtained for images using the

descriptors that were extracted using CPU based implementation of SIFT. Both ORB and BRIEF were also used in the comparison. These two descriptors were used for representing a typical handcrafted binary descriptor. VGG16 has been used as the foundation of DBD-MQ, and DeepBit. Both the descriptors represent the self-supervised binary-valued descriptor category.

NIN-RBM significantly outperforms both DeepBit and PCA-ITQ in terms of encoding time. Moreover, this significant difference does not get affected by the running platforms such that the CPU and GPU. Although the CPU based implementation of SIFT is extremely slow, it is noteworthy that the GPU based implementation is extremely faster, as it is second only to the BRIEF. The GPU-SIFT even outperforms the proposed descriptor that is executed in the same GPU, in terms of encoding time. It is even not comparable with the CPU version of SIFT, when NIN-RBM is executed using the CPU. Nevertheless, the encoding time difference of GPU-SIFT and NIN-RBM is not significant when executed on the GPU. Furthermore, both the GPU and CPU based implementations of SIFT output a real-valued descriptor with the dimension, 128. Comparing a pair of binary-valued descriptors requires a few bitwise operations in contrast to many arithmetic operations that are performed with floating-point precision for comparing real-valued descriptors. Hence the 256-bit binary descriptor of the proposed method is still beneficial in the patch matching phase. Moreover, this similarity measurement phase requires a significant amount of time due to the exhaustive comparison of patches.

The computational efficiency of DeepBit depends on the model that is used to extract the features. In these experiments, DeepBit is used with VGG16, as it is recommended in the original study. The extreme encoding time of DeepBit is inherited from the VGG16 model. PCA-ITQ includes the extraction of SIFT feature descriptors and PCA projection. These two steps make PCA-ITQ significantly slower compared to the other descriptors. NIN-RBM, in contrast to the DeepBit, uses the NIN model that consists of only four mlpconv layers with the single layer RBM component. This ultra-small model makes the encoding time significantly lower compared to the other deep learning-based descriptors. This number of parameters presented in Table 7.9 properly aligns with the encoding time of the proposed descriptor, NIN-RBM shown in Figure

7.21. Moreover, the GPU based implementation of NIN-RBM is comparable to the ORB that is executed in the CPU. It is noteworthy that the size of the descriptor does not affect the encoding time significantly when executed in GPU.

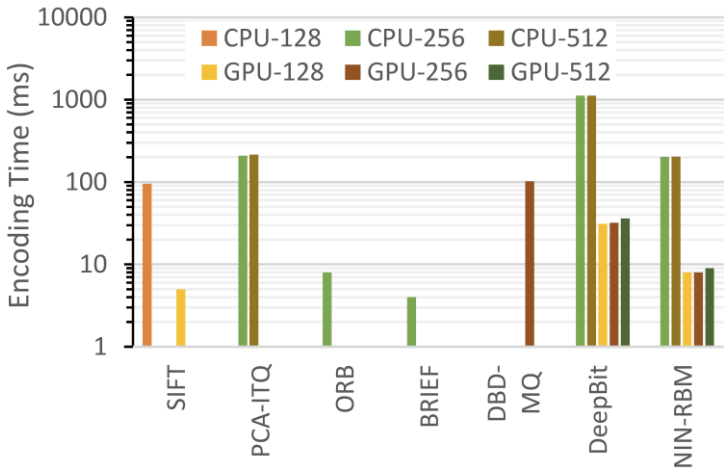


Figure 7.21: Mean time taken for feature encoding. Dataset: CIFAR-10

7.5 Summary

In this chapter, the results of six experiments have been discussed. Initially, the evaluation results of the three different versions of SDPF descriptor, namely the original SDPF, SDPF-DDD, and HSDPF, have been discussed. First, the invariant properties of the original SDPF were evaluated, and the results revealed that it is both rotational and scale-invariant with very high confidence. Then the classification performance of SDPF-DDD was compared with many handcrafted feature descriptors and the original SDPF. The results revealed that SDPF-DDD not only improves the SDPF but also closely competes with the best-performed descriptor, namely FC-GPHOG. The third version of SDPF, i.e., the HSDPF was compared with many handcrafted and deep-learned feature descriptors. The results revealed the HSDPF is advantageous over all the other methods, in terms of training efficiency, due to not requiring data augmentation. Further, the results revealed that the computation performance of HSDPF is consistent in resource-constrained devices like Raspberry Pi.

Then the results of two experiments that were carried out for evaluating the video segmentation were discussed. The first method that uses the clustering method achieved the best result with K-means clustering with an estimated K. The second version of the segmentation method based on HOOF and superpixel outperformed the previous version and also the other state-of-the-art methods used in the comparison.

Finally, the results of the correspondence matching have been discussed. A set of experimental results covering the reliability, robustness, consistency, and computational efficiency have been discussed in detail. Results revealed that NIN-RBM, the proposed binary descriptor outperforms all the self-supervised binary-valued descriptors taken into the comparison, in terms of the precision of correspondence matching and computational efficiency.

CHAPTER 8

CONCLUSION AND RECOMMENDATIONS

This study has introduced a visual data depiction method, focusing on depicting digital video, that can be applied in several different scenarios of retrieval applications. It has been identified that the low computational efficiency of visual feature description is a significant problem in video depiction. Hence, the study focused on finding methods to efficiently depict the content of video frames, targeting the retrieval applications.

The study included three major parts, namely, recognizing visual data presence in video frames or images, segmenting objects in video frames using spatial and temporal information and visual correspondence matching for retrieving visual queries from videos. The recognition of visual data has been evolved over three closely related versions, and the improvement has been reported through a set of extensive experiments. The segmentation method has also been evolved in two versions. Finally, a novel correspondence matching technique has been introduced and evaluated for the matching performance and computational efficiency. In this chapter, the conclusions that have been derived from the results and discussion presented in Chapter 7, are reported separately depending on each of the three major parts of the study.

Efficient Visual Data Abstraction with Salient Dither Pattern Feature (SDPF)

Geometrical invariance is a necessary property that any visual descriptor should have. Since SDPF is based on an analysis of square-shaped windows of pixels, both the rotational and scale invariance were evaluated. Results revealed that SDPF could achieve high rotational and scale invariance with a very high confidence value. The results supported the conclusion that colour dithering operation that is based on a square-shaped window of pixels does not significantly affect the geometrical invariance of the final spatial chromatic histogram descriptor of the SDPF. However, the classification performance, obtained with the original SDPF, could not outperform some state-of-the-art descriptors, although it is successful in rotational invariance and computational efficiency. Hence, the second version was introduced, incorporating more details to the descriptor. This additional detail, named “dither density descriptor”,

integrates the regional colour distribution in terms of dither colour density. The evaluation results show that the improved version, i.e., the SDPF-DDD, outperforms the SDPF and closely competes with the state-of-the-art handcrafted descriptors. It can be concluded that SDPF-DDD is equally favourable to the state-of-the-art for object recognition in visual data, with the support of these results. Moreover, SDPF-DDD is more favourable for recognizing objects in video frames, due to the computational efficiency and the extremely low dimension of the descriptor.

Both the versions of SDPF could not compete with the deep-learned model in terms of the classification performance though they are advantageous in terms of computational efficiency. Furthermore, deep learning-based methods known to perform any handcrafted descriptor for large datasets; hence SDPF was further improved and invented HSDPF. The results revealed that HSDPF is more favourable than the state-of-the-art handcrafted descriptors and deep-learned methods, in terms of computational efficiency and robustness with minimum supervision, as it performed better than all other methods without any data augmentation. These results lead to the conclusion that the Hessian based SDPF descriptor that has been orientation-normalized using the estimated dominant orientation object from the least square method is more favourable for highly invariant object detection. Moreover, the results prove that HSDPF can extract features and describe them solely inside resource-constrained devices such as Raspberry Pi. Furthermore, the low complexity of the algorithm that is presented in Chapter 7 supports the conclusion that HSDPF can maintain its reputation as an efficient computational descriptor, even in processing high-resolution visual data. This is a demanding feature for a video depicter, as the applied video resolution is being improved day by day.

In summation, SDPF has been evolved through three different versions, and the results of the last version support the following conclusions.

1. Colour dithering can be used as a dimensionality reduction technique for visual data.

2. Hessian based salient point detection is more stable compared to the simple dither pattern matching based saliency detection that has been used in the first two versions.
3. The spatial-chromatic histogram, that has undergone an orientation normalization technique, has made the descriptor invariant to geometrical differences; hence no data augmentation is required.
4. The feature extraction and description are extremely fast, even in resource-constrained devices, due to algorithmic optimization that has been described in Chapter 3.
5. The complexity of the algorithm is linear; hence it can reasonably handle inputs in higher resolutions.

Video Frame Segmentation for Separating Individual Objects

The proposed object segmentation technique has been evolved through two different versions. The results of the version revealed that K-means based clustering of SDPF features with their colour, flow angle and flow magnitude can achieve segmentation results in terms of completeness and spatial accuracy error, better than using density-based scanning and expectation maximization-based clustering. Moreover, the results revealed that the result is not consistent over different frames due to the propagation of the error in the estimation of the number of objects. As a result, the second version that utilizes the histogram of oriented optical flow and superpixels with a refinement technique were introduced. The evaluation results revealed that the second version outperforms the K-means based first version and other motion and structure-based techniques. It further establishes the consistency over the frames in the test videos. This result supports the conclusion that removing the dependency of the number of object estimation with the superpixel based feature clustering and the time series of HOOFs based refinement technique can improve the segmentation results.

It is noteworthy to mention that the proposed segmentation method is limited in using videos that have no global or local motion. Nevertheless, a slight motion is still sufficient, due to the fact that the algorithm uses a very shallow history of frames to calculate the time-series of HOOFs.

This proposed segmentation can be used for individual object detection, combined with SDPF, with a minimum overhead due to both object detection and segmentation that can utilize the same SDPF point.

Correspondence Matching with the Binary Descriptor for Retrieving Visual Data with a Visual Query

The proposed binary descriptor, NIN-RBM with the novel learning approach has been extensively evaluated with many datasets by applying it in correspondence matching. It consists of a lightweight deep neural network (DNN), which is a hybrid of two well-known models called NIN and RBM. The proposed binary descriptor, NIN-RBM outperformed all the deep learned binary descriptors for most of the datasets. Furthermore, it closely competes with the deep learned real-valued descriptors as well. The results revealed that the NIN model is more capable of learning invariant features, as the proposed method has achieved better results with the dataset that consists of many geometrically different pairs of patches. Further, the results support the conclusion that RBM can generate extremely discriminative binary descriptors from NIN based features, while minimizing reconstruction loss by using contrastive divergence learning. Moreover, it can be concluded that the simultaneous optimization of both NIN and RBM components can optimize the features already learned for general image classification to the task of correspondence matching. The optimization of mlpconv features in NIN was carried out using a fine-tuning approach based on the reconstructed features from a representative binary code. The results revealed that this approach has successfully reduced the intra-class variations by giving better performance for the dataset that needs greater robustness of the descriptors in correspondence matching. As NIN-RBM outperforms DeepBit, which is based on a better deep model called VGG, further it can be concluded that the generic convolutional features can be further optimized for other tasks, such as correspondence matching, with local image patches. NIN-RBM is more favourable for practical applications due to the following reasons:

- Smaller model size
    - Low storage space requirement
    - Quick loading of model parameters

- An extremely smaller number of parameters
    - Less time taken by the training phase of NIN
    - Lower encoding time.
- The lightweight architecture model permits the usage in resource-constrained devices.
- Self-supervised learning
    - Can easily be adapted to any application, as no annotated data is required for training
- Binary-valued descriptor
    - Extremely fast in matching due to the ability to use Hamming distance to quantify the similarity
    - Can be used in real-time applications
    - Requires less storage space
- Robust descriptor
    - The features learned out of the mlpconv layers in NIN have more potency in learning invariant features with the support from the proposed learning approach.

The proposed binary descriptor requires manual intervention, as it consists of several steps. Further, it is difficult to define a termination criterion for the training approach; hence the algorithm must be stopped empirically. Future works may direct their attention to finding a proper termination criterion for this proposed learning approach. Furthermore, it will be worthy of exploring other possible deep learned features that can be combined with RBM to train with the proposed learning approach. Although this study discusses correspondence matching, targeting the retrieval of information, this binary descriptor has more potential applications, such as real-time object recognition, panoramic stitching, object pose estimation, object tracking, and 3D reconstruction, etc. Out of these applications, pose estimation of objects can be effectively used for improving the semantics retrieved from the visual data because, the pose of an object can significantly affect the context of the video.

To sum up, this study focusses on depicting visual data in terms of the possibility of being used in video retrieval applications, using dimensionality reduction techniques. The contribution from the study includes the interpretation of colour dithering as a dimensionality reduction technique for visual data, the use of colour dithering to derive a novel feature descriptor that is very efficient in computing and using it to segment the objects from video frames in order to recognize them independently. Furthermore, in case of having an example image of an object, the proposed correspondence matching technique can be used to retrieve similar images/frames from a vast archive or a video. This thesis has described the significance of NIN-RBM with sufficient evidence extracted from the experimental results.

Finally, it can be concluded that the object segmentation, and recognition approaches proposed in this thesis can be used for efficiently retrieve semantics from videos with fine details such as the pose and location of the objects with the help of the proposed local binary descriptor using correspondence matching.

REFERENCES

[1] J. Wagemans *et al.*, "A century of Gestalt psychology in visual perception: I. Perceptual grouping and figure–ground organization.," *Psychol. Bull.*, vol. 138, no. 6, p. 1172, 2012.

[2] Z. Kourtzi, L. R. Betts, P. Sarkheil, and A. E. Welchman, "Distributed neural plasticity for shape learning in the human visual cortex," *PLoS Biol.*, vol. 3, no. 7, 2005.

[3] P. J. Kellman and M. E. Arterberry, "Infant visual perception," *Handb. Child Psychol.*, vol. 2, 2007.

[4] V. Ramachandran and S. Anstis, "Extrapolation of motion path in human visual perception.," *Vision Res.*, 1983.

[5] D. Rose and R. Blake, "Motion perception: from phi to omega," *Philos. Trans. R. Soc. Lond. B. Biol. Sci.*, vol. 353, no. 1371, pp. 967–980, 1998.

[6] W. Hu, N. Xie, L. Li, X. Zeng, and S. Maybank, "A survey on visual content-based video indexing and retrieval," *Syst. Man Cybern. Part C Appl. Rev. IEEE Trans. On*, vol. 41, no. 6, pp. 797–819, 2011.

[7] J. R. R. Uijlings, A. W. M. Smeulders, and R. J. H. Scha, "Real-Time Visual Concept Classification," *Multimed. IEEE Trans. On*, vol. 12, no. 7, pp. 665–681, Nov. 2010, doi: 10.1109/TMM.2010.2052027.

[8] Yang Mingqiang, Kpalma K. Idiyo, and Ronsin Joseph, "A Survey of Shape Feature Extraction Techniques," *Pattern Recognit.*, pp. 43–90, Nov. 2008.

[9] A. B. Benitez *et al.*, "Semantics of Multimedia in MPEG-7," in *Proceedings. International Conference on Image Processing*, 2002, vol. 1, p. 1.

[10] L. Rossetto *et al.*, "IMOTION—a content-based video retrieval engine," in *International Conference on Multimedia Modeling*, 2015, pp. 255–260.

[11] J. Lokoč, G. Kovalčík, T. Souček, J. Moravec, and P. Čech, "VIRET: A video retrieval tool for interactive known-item search," in *Proceedings of the 2019 on International Conference on Multimedia Retrieval*, 2019, pp. 177–181.

[12] L. Cinque, G. Ciocca, S. Levialdi, A. Pellicano, and R. Schettini, "Color-based image retrieval using spatial-chromatic histograms," *Image Vis. Comput.*, vol. 19, no. 13, pp. 979–986, 2001.

[13] R. Chakravarti and X. Meng, "A study of colour histogram based image retrieval," in *Information Technology: New Generations, 2009. ITNG'09. Sixth International Conference on*, 2009, pp. 1323–1328.

[14] F. C. Heilbron, W. Barrios, V. Escorcia, and B. Ghanem, "Scc: Semantic context cascade for efficient action detection," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 3175–3184.

[15] Y. Nakajima and H. Saito, "Efficient object-oriented semantic mapping with object detector," *IEEE Access*, vol. 7, pp. 3206–3213, 2018.

[16] Y. Nishizumi *et al.*, "FPGA implementation of object recognition processor for HDTV resolution video using sparse FIND feature," in *2017 IEEE International Workshop on Signal Processing Systems (SiPS)*, 2017, pp. 1–6.

[17] A. Gorave, S. Misra, O. Padir, A. Patil, and K. Ladole, "Suspicious Activity Detection Using Live Video Analysis," in *Proceeding of International Conference on Computational Science and Applications*, 2020, pp. 203–214.

[18] H. Kiani Galoogahi, A. Fagg, C. Huang, D. Ramanan, and S. Lucey, "Need for speed: A benchmark for higher frame rate object tracking," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1125–1134.
[19] X. Yang, T. Zhang, and C. Xu, "A new discriminative coding method for image classification," *Multimed. Syst.*, pp. 133–145, 2015, doi: 10.1007/s00530-014-0376-y.
[20] J. Zhang, Y. Han, and J. Jiang, "Semi-supervised tensor learning for image classification," *Multimed. Syst.*, pp. 1–11, 2014.
[21] A. Sinha, S. Banerji, and C. Liu, "New color GPHOG descriptors for object and scene image classification," *Mach. Vis. Appl.*, vol. 25, no. 2, pp. 361–375, 2014.
[22] N. Singhai and S. K. Shandilya, "A Survey On:'Content Based Image Retrieval Systems,'" *Int. J. Comput. Appl.*, vol. 4, no. 2, pp. 22–26, 2010.
[23] P. Bolettieri *et al.*, "CoPhIR: a test collection for content-based image retrieval," *ArXiv Prepr. ArXiv09054627*, 2009.
[24] T. Deserno, S. Antani, and L. Rodney Long, "Content-based image retrieval for scientific literature access," *Methods Inf. Med.*, vol. 48, no. 4, p. 371, 2009.
[25] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *Proceedings of the IEEE International Conference on Computer Vision*, Nov. 2011, pp. 2564–2571, doi: 10.1109/ICCV.2011.6126544.
[26] G. Takacs, V. Chandrasekhar, S. Tsai, D. Chen, R. Grzeszczuk, and B. Girod, "Unified real-time tracking and recognition with rotation-invariant fast features," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, 2010, pp. 934–941.
[27] D. Debnath and R. Parekh, "Content Based Image Retrieval Using Directional Color Correlograms," *Int. J. Eng. Sci.*, vol. 3, 2011.
[28] C. Taranto, N. Di Mauro, S. Ferilli, and F. Esposito, "Approximate image color correlograms," in *Proceedings of the international conference on Multimedia*, 2010, pp. 1127–1130.
[29] R. Alaoui, S. O. El Alaoui, and M. Meknassi, "Spatial color indexing: An efficient and robust technique for content-based image retrieval," *J. Comput. Sci.*, vol. 5, no. 2, p. 109, 2009.
[30] Y. M. Ro, M. Kim, H. K. Kang, B. Manjunath, and J. Kim, "MPEG-7 homogeneous texture descriptor," *ETRI J.*, vol. 23, no. 2, pp. 41–51, 2001.
[31] K. L. Lee and L. H. Chen, "An efficient computation method for the texture browsing descriptor of MPEG-7," *Image Vis. Comput.*, vol. 23, no. 5, pp. 479–489, 2005.
[32] S. Rahman, S. M. Naim, A. A. Farooq, and M. M. Islam, "Performance of PCA based semi-supervised learning in face recognition using MPEG-7 edge histogram descriptor," *J. Multimed.*, vol. 6, no. 5, pp. 404–415, 2011.
[33] D. Zhang and G. Lu, "Evaluation of MPEG-7 shape descriptors against other shape descriptors," *Multimed. Syst.*, vol. 9, no. 1, pp. 15–30, 2003.
[34] W. Hsu, T.-S. Chua, and H. K. Pung, "An Integrated Color-Spatial Approach to Content-Based Image Retrieval," in *ACM Multimedia*, 1995, pp. 305–313.
[35] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, 2005, vol. 1, pp. 886–893.

[36] A. Bosch, A. Zisserman, and X. Munoz, “Representing shape with a spatial pyramid kernel,” in *Proceedings of the 6th ACM international conference on Image and video retrieval*, 2007, pp. 401–408.
[37] David G. Lowe, “Distinctive Image Features from Scale-Invariant Keypoints,” *Int. J. Comput. Vis.*, Jan. 2004.
[38] W. Yang *et al.*, “Structure from Motion on XSlit Cameras,” *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 1–1, 2019, doi: 10.1109/TPAMI.2019.2957119.
[39] H. Li and G. Hua, “Probabilistic Elastic Part Model: A Pose-Invariant Representation for Real-World Face Verification,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 918–930, 2018, doi: 10.1109/TPAMI.2017.2695183.
[40] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, “Speeded-Up Robust Features (SURF),” *Comput. Vis. Image Underst.*, vol. 110, no. 3, Sep. 2008.
[41] Luo Juan and Oubong Gwun, “A Comparison of SIFT, PCA-SIFT and SURF,” *Int. J. Image Process.*, vol. 3, no. 4, pp. 143–152, 2009.
[42] X. Yuan, J. Yu, Z. Qin, and T. Wan, “A SIFT-LBP image retrieval model based on bag-of features,” in *International Conference on Image Processing (ICIP)*, 2011, pp. 1061–1064.
[43] S. Hwang, “Bag-Of-Visual-Words Approach based on SURF Features to Polyp Detection in Wireless Capsule Endoscopy Videos,” Jul. 2011, vol. 2, pp. 941–944.
[44] F. Yang, H. Lu, W. Zhang, and G. Yang, “Visual tracking via bag of features,” *Image Process. IET*, vol. 6, no. 2, pp. 115–128, 2012, doi: 10.1049/iet-ipr.2010.0127.
[45] M. S. Martin Stommel, “Binarising SIFT-descriptors to reduce the curse of dimensionality in histogram-based object recognition,” *Int. J. Signal Process. Image Process. Pattern Recognit.*, vol. 3, no. 1, pp. 25–36, 2010.
[46] R. Chakraborty, L. Yang, S. Hauberg, and B. Vemuri, “Intrinsic Grassmann Averages for Online Linear, Robust and Nonlinear Subspace Learning,” *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 1–1, 2020, doi: 10.1109/TPAMI.2020.2992392.
[47] Y. Xu, G. Feng, and Y. Zhao, “One improvement to two-dimensional locality preserving projection method for use with face recognition,” *Neurocomputing*, vol. 73, no. 1, pp. 245–249, 2009.
[48] R. Kapoor and R. Gupta, “Non-linear dimensionality reduction using fuzzy lattices,” *Comput. Vis. IET*, vol. 7, no. 3, p., 2013, doi: 10.1049/iet-cvi.2012.0097.
[49] X. Zhou, X. Zhuang, H. Tang, M. Hasegawa-Johnson, and T. S. Huang, “Novel Gaussianized vector representation for improved natural scene categorization,” *Pattern Recognit. Lett.*, vol. 31, no. 8, pp. 702–708, 2010.
[50] Y. Ke and R. Sukthankar, “PCA-SIFT: A more distinctive representation for local image descriptors,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2004, vol. 2, pp. 497–506.
[51] L. Akarun, Y. Yardunci, and A. E. Cetin, “Adaptive methods for dithering color images,” *Image Process. IEEE Trans. On*, vol. 6, no. 7, pp. 950–955, 1997.

[52] L. Ranathunga, R. Zainuddin, and N. A. Abdullah, “Compacted Dither Pattern Codes Over Mpeg-7 Dominant Colour Descriptor In Video Visual Depiction,” *Malays. J. Comput. Sci.*, vol. 23, no. 2, pp. 68–84, 2010.

[53] L. Ranathunga, R. Zainuddin, and N. A. Abdullah, “Performance evaluation of the combination of Compacted Dither Pattern Codes with Bhattacharyya classifier in video visual concept depiction,” *Multimed. Tools Appl.*, vol. 54, no. 2, pp. 263–289, 2011.

[54] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[55] M. Paulin, M. Douze, Z. Harchaoui, J. Mairal, F. Perronin, and C. Schmid, “Local convolutional features with unsupervised training for image retrieval,” in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 91–99.

[56] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *ArXiv Prepr. ArXiv14091556*, 2014.

[57] K. Lin, J. Lu, C.-S. Chen, and J. Zhou, “Learning compact binary descriptors with unsupervised deep neural networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1183–1192.

[58] H. Arora, N. Loeff, D. A. Forsyth, and N. Ahuja, “Unsupervised Segmentation of Objects using Efficient Learning,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2007, pp. 1–7, doi: 10.1109/CVPR.2007.383011.

[59] S. Gould, T. Gao, and D. Koller, “Region-based segmentation and object detection,” in *Advances in neural information processing systems*, 2009, pp. 655–663.

[60] A. Vögele, B. Krüger, and R. Klein, “Efficient Unsupervised Temporal Segmentation of Human Motion,” Copenhagen, Denmark, Jul. 2014.

[61] K.- Maninis *et al.*, “Video Object Segmentation without Temporal Information,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 6, pp. 1515–1530, 2019, doi: 10.1109/TPAMI.2018.2838670.

[62] S. W. Oh, J. Lee, N. Xu, and S. J. Kim, “Fast User-Guided Video Object Segmentation by Interaction-And-Propagation Networks,” in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 5242–5251, doi: 10.1109/CVPR.2019.00539.

[63] M. E. Aallaoui and A. Gourch, “Spatio-temporal segmentation with Mumford-Shah functional,” in *Computer Systems and Applications (AICCSA), 2013 ACS International Conference on*, 2013, pp. 1–4.

[64] R. Li, S. Yu, and X. Yang, “Efficient spatio-temporal segmentation for extracting moving objects in video sequences,” *Consum. Electron. IEEE Trans. On*, vol. 53, no. 3, pp. 1161–1167, 2007.

[65] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[66] H. Noh, S. Hong, and B. Han, “Learning deconvolution network for semantic segmentation,” in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1520–1528.
[67] O. A. Abbas, “Comparisons Between Data Clustering Algorithms.,” *Int Arab J Inf Technol*, vol. 5, no. 3, pp. 320–325, 2008.
[68] X. Jin and J. Han, “Expectation Maximization Clustering,” in *Encyclopedia of Machine Learning*, Springer, 2010, pp. 382–383.
[69] X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg, “Matchnet: Unifying feature and metric learning for patch-based matching,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3279–3286.
[70] S. Zagoruyko and N. Komodakis, “Learning to compare image patches via convolutional neural networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 4353–4361.
[71] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, “Brief: Binary robust independent elementary features,” in *European conference on computer vision*, 2010, pp. 778–792.
[72] S. Leutenegger, M. Chli, and R. Y. Siegwart, “BRISK: Binary robust invariant scalable keypoints,” in *2011 International conference on computer vision*, 2011, pp. 2548–2555.
[73] A. Alahi, R. Ortiz, and P. Vandergheynst, “Freak: Fast retina keypoint,” in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 510–517.
[74] K. Lin, J. Lu, C.-S. Chen, J. Zhou, and M.-T. Sun, “Unsupervised Deep Learning of Compact Binary Descriptors,” *IEEE Trans. Pattern Anal. Mach. Intell.*, 2018.
[75] C. Strecha, A. Bronstein, M. Bronstein, and P. Fua, “LDAHash: Improved matching with smaller descriptors,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 1, pp. 66–78, 2012.
[76] T. Trzcinski and V. Lepetit, “Efficient discriminative projections for compact binary descriptors,” in *European Conference on Computer Vision*, 2012, pp. 228–242.
[77] B. Fan, Q. Kong, T. Trzcinski, Z. Wang, C. Pan, and P. Fua, “Receptive fields selection for binary feature description,” *IEEE Trans. Image Process.*, vol. 23, no. 6, pp. 2583–2595, 2014.
[78] X. Yang and K.-T. Cheng, “Local difference binary for ultrafast and distinctive feature description,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 1, pp. 188–194, 2014.
[79] T. Trzcinski, M. Christoudias, P. Fua, and V. Lepetit, “Boosting binary keypoint descriptors,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 2874–2881.
[80] A. Andoni and P. Indyk, “Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions,” in *Foundations of Computer Science, 2006. FOCS’06. 47th Annual IEEE Symposium on*, 2006, pp. 459–468.
[81] R. Salakhutdinov and G. Hinton, “Semantic hashing,” *Int. J. Approx. Reason.*, vol. 50, no. 7, pp. 969–978, 2009.

[82] R. Xia, Y. Pan, H. Lai, C. Liu, and S. Yan, "Supervised hashing for image retrieval via image representation learning.," in *AAAI*, 2014, vol. 1, p. 2.
[83] Y. Tian, B. Fan, and F. Wu, "L2-Net: Deep Learning of Discriminative Patch Descriptor in Euclidean Space.," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, vol. 1, pp. 661–669.
[84] H. Liu, R. Wang, S. Shan, and X. Chen, "Deep supervised hashing for fast image retrieval," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2064–2072.
[85] H. Lai, Y. Pan, Y. Liu, and S. Yan, "Simultaneous feature learning and hash coding with deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3270–3278.
[86] M. Lin, Q. Chen, and S. Yan, "Network in network," *ArXiv Prepr. ArXiv13124400*, 2013.
[87] R. Bandara, L. Ranathunga, and N. A. Abdullah, "Invariant Properties of a Locally Salient Dither Pattern with a Spatial-Chromatic Histogram," Peradeniya, Sri Lanka, Dec. 2013.
[88] "Caltech." http://www.vision.caltech.edu/archive.html.
[89] James Z. Wang, "Corel 1000 and Corel 10000 image database." http://wang.ist.psu.edu/docs/related.shtml.
[90] D. Zhang and G. Lu, "A comparative study on shape retrieval using Fourier descriptors with different shape signatures," in *Proceedings of international conference on intelligent multimedia and distance education*, 2001, pp. 1–9.
[91] D. Zhang and G. Lu, "Content-Based Shape Retrieval Using Different Shape Descriptors: A Comparative Study.," in *IEEE International Conference on Multimedia and Expo*, 2001, pp. 1139–1142, doi: 10.1109/ICME.2001.1237928.
[92] S. Shaila and A. Vadivel, "Content-Based Image Retrieval Using Modified Human Colour Perception Histogram," *ITCS SIP JSE-2012 CS IT*, vol. 4, pp. 229–237, 2012.
[93] G. D. Finlayson and G. Schaefer, "Hue that is invariant to brightness and gamma.," in *BMVC*, 2001, pp. 1–10.
[94] V. Ostromoukhov, "A simple and efficient error-diffusion algorithm," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, 2001, pp. 567–572.
[95] A. Bordes, S. Ertekin, J. Weston, and L. Bottou, "Fast kernel classifiers with online and active learning," *J. Mach. Learn. Res.*, vol. 6, no. Sep, pp. 1579–1619, 2005.
[96] G. J. Brostow, J. Fauqueur, and R. Cipolla, "Semantic Object Classes in Video: A High-Definition Ground Truth Database," *Pattern Recognit. Lett.*, vol. 30, no. 2, pp. 88–97, 2008.
[97] A. Papazoglou and V. Ferrari, "Fast object segmentation in unconstrained video," in *Computer Vision (ICCV), 2013 IEEE International Conference on*, 2013, pp. 1777–1784.
[98] A. Jain, S. Chatterjee, and R. Vidal, "Coarse-to-Fine Semantic Video Segmentation Using Supervoxel Trees," in *IEEE International Conference on Computer Vision*, Dec. 2013, pp. 1865–1872, doi: 10.1109/ICCV.2013.234.

[99] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *Pattern Anal. Mach. Intell. IEEE Trans. On*, vol. 34, no. 11, pp. 2274–2282, 2012.
[100] S. N. Tamgade and V. R. Bora, "Motion vector estimation of video image by pyramidal implementation of Lucas Kanade optical flow," in *International Conference on Emerging Trends in Engineering and Technology*, 2009, pp. 914–917.
[101] R. Chaudhry, A. Ravichandran, G. Hager, and R. Vidal, "Histograms of oriented optical flow and binet-cauchy kernels on nonlinear dynamical systems for the recognition of human actions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2009, pp. 1932–1939.
[102] M. Tkalcic, J. F. Tasic, and others, "Colour spaces: perceptual, historical and applicational background," in *The IEEE Region 8 EUROCON 2003. Computer as a Tool.*, 2003, pp. 304–308.
[103] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Comput.*, vol. 14, no. 8, pp. 1771–1800, 2002.
[104] Y. Duan, J. Lu, Z. Wang, J. Feng, and J. Zhou, "Learning deep binary descriptor with multi-quantization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1183–1192.
[105] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky, "Neural codes for image retrieval," in *European conference on computer vision*, 2014, pp. 584–599.
[106] M. Wollborn and R. Mech, "Refined procedure for objective evaluation of video object generation algorithms," *Doc ISOIEC JTC1SC29WG11 M*, vol. 3448, Mar. 1998.
[107] O. Russakovsky *et al.*, "Imagenet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.
[108] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–8.
[109] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Lost in quantization: Improving particular object retrieval in large scale image databases," 2008.
[110] H. Jegou, M. Douze, and C. Schmid, "Hamming embedding and weak geometric consistency for large scale image search," in *European conference on computer vision*, 2008, pp. 304–317.
[111] M. Brown, G. Hua, and S. Winder, "Discriminative learning of local image descriptors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 1, pp. 43–57, 2011.
[112] V. Balntas, K. Lenc, A. Vedaldi, and K. Mikolajczyk, "HPatches: A benchmark and evaluation of handcrafted and learned local descriptors," 2017.
[113] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," University of Toronto, 2009.
[114] G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla, "Segmentation and recognition using structure from motion point clouds," in *European conference on computer vision*, Springer, 2008, pp. 44–57.
[115] J. Liu, P. L. Rosin, X. Sun, J. Xiao, and Z. Lian, "Image-driven unsupervised 3D model co-segmentation," *Vis. Comput.*, vol. 35, no. 6–8, pp. 909–920, 2019.

[116] E. Rosten and T. Drummond, “Machine learning for high speed corner detection,” in *Euproean Conference on Computer Vision*, 2006, vol. 1, pp. 430–443.
[117] H. Jegou, F. Perronnin, M. Douze, J. Sánchez, P. Perez, and C. Schmid, “Aggregating local image descriptors into compact codes,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 9, pp. 1704–1716, 2012.
[118] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, “CNN features off-the-shelf: an astounding baseline for recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2014, pp. 806–813.
[119] J. Wan *et al.*, “Deep learning for content-based image retrieval: A comprehensive study,” in *Proceedings of the 22nd ACM international conference on Multimedia*, 2014, pp. 157–166.
[120] G. Yunchao, S. Lazebnik, A. Gordo, and F. Perronnin, “Iterative quantization: a procrustean approach to learning binary codes for large-scale image retrieval,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 12, pp. 2916–2929, 2013.
[121] K. Reddy Mopuri and R. Venkatesh Babu, “Object level deep feature pooling for compact image representation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2015, pp. 62–70.
[122] N. Markuš, I. Pandžić, and J. Ahlberg, “Learning Local Descriptors by Optimizing the Keypoint-Correspondence Criterion: Applications to Face Matching, Learning From Unlabeled Videos and 3D-Shape Retrieval,” *IEEE Trans. Image Process.*, vol. 28, no. 1, pp. 279–290, 2018.
[123] Y. Tian, X. Yu, B. Fan, F. Wu, H. Heijnen, and V. Balntas, “SOSNet: Second order similarity regularization for local descriptor learning,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11016–11025.
[124] Z. Luo *et al.*, “Geodesc: Learning local descriptors by integrating geometry constraints,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 168–183.
[125] M. arten Björkman, N. Bergström, and D. Kragic, “Detecting, segmenting and tracking unknown objects using multi-label MRF inference,” *Comput. Vis. Image Underst.*, vol. 118, pp. 111–127, 2014.

APPENDIX A: TABLE OF PRE-CALCULATED ERROR DIFFUSION COEFFICIENT VECTORS

The following table shows the distribution coefficients in form of i : A_{10}, A_{-11}, A_{01} for the input levels i in the interval [0…127]. $d_{10}(i) = A_{10}(i)/M(i)$;$d_{-11}(i) = A_{-11}(i)/M(i)$;$d_{01}(i) = A_{01}(i)/M(i)$; $M(i) = (A_{10}(i) + A_{-11}(i) + A_{01}(i))$, and $D(i) = \{d_{10}(i), d_{-11}(i), d_{01}(i)\} == D(255 – i)$.

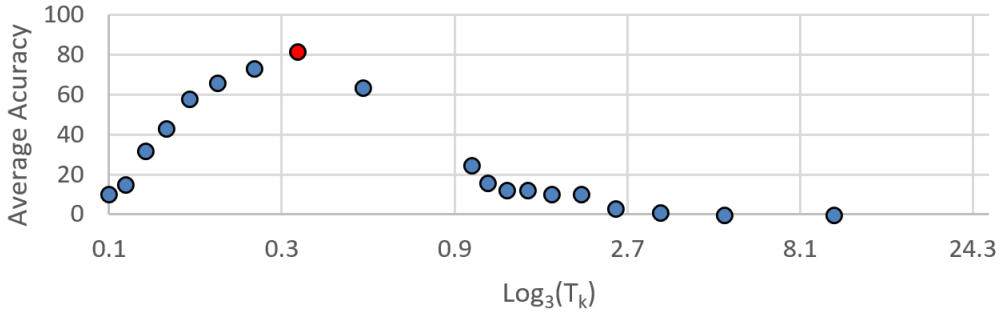
0: 13, 0, 5	32: 20, 10, 19	64: 11, 10, 0	96: 5, 3, 2
1: 13, 0, 5	33: 1937,1000,1767	65: 158, 151, 3	97: 5, 3, 2
2: 21, 0, 10	34: 977, 520, 855	66: 178, 179, 7	98: 5, 3, 2
3: 7, 0, 4	35: 657, 360, 551	67: 1030,1091,63	99: 5, 3, 2
4: 8, 0, 5	36: 71, 40, 57	68: 248, 277, 21	100: 5, 3, 2
5: 47, 3, 28	37: 2005,1160,1539	69: 318, 375, 35	101: 5, 3, 2
6: 23, 3, 13	38: 337, 200, 247	70: 458, 571, 63	102: 5, 3, 2
7: 15, 3, 8	39: 2039,1240,1425	71: 878, 1159,147	103: 5, 3, 2
8: 22, 6, 11	40: 257, 160, 171	72: 5, 7, 1	104: 5, 3, 2
9: 43, 15, 20	41: 691, 440, 437	73: 172, 181, 37	105: 5, 3, 2
10: 7, 3, 3	42: 1045,680, 627	74: 97, 76, 22	106: 5, 3, 2
11: 501, 224, 211	43: 301, 200, 171	75: 72, 41, 17	107: 5, 3, 2
12: 249, 116, 103	44: 177, 120, 95	76: 119, 47, 29	108: 305, 176, 119
13: 165, 80, 67	45: 2141,1480,1083	77: 4, 1, 1	109: 155, 86, 59
14: 123, 62, 49	46: 1079,760, 513	78: 4, 1, 1	110: 105, 56, 39
15: 489, 256, 191	47: 725, 520, 323	79: 4, 1, 1	111: 80, 41, 29
16: 81, 44, 31	48: 137, 100, 57	80: 4, 1, 1	112: 65, 32, 23
17: 483, 272, 181	49: 2209,1640,855	81: 4, 1, 1	113: 55, 26, 19
18: 60, 35, 22	50: 53, 40, 19	82: 4, 1, 1	114: 335, 152, 113
19: 53, 32, 19	51: 2243,1720,741	83: 4, 1, 1	115: 85, 37, 28
20: 237, 148, 83	52: 565, 440, 171	84: 4, 1, 1	116: 115, 48, 37
21: 471, 304, 161	53: 759, 600, 209	85: 4, 1, 1	117: 35, 14, 11
22: 3, 2, 1	54: 1147,920, 285	86: 65, 18, 17	118: 355, 136, 109
23: 481, 314, 185	55: 2311,1880,513	87: 95, 29, 26	119: 30, 11, 9
24: 354, 226, 155	56: 97, 80, 19	88: 185, 62, 53	120: 365, 128, 107
25: 1389,866, 685	57: 335, 280, 57	89: 30, 11, 9	121: 185, 62, 53
26: 227, 138, 125	58: 1181,1000,171	90: 35, 14, 11	122: 25, 8, 7
27: 267, 158, 163	59: 793, 680, 95	91: 85, 37, 28	123: 95, 29, 26
28: 327, 188, 220	60: 599, 520, 57	92: 55, 26, 19	124: 385, 112, 103
29: 61, 34, 45	61: 2413,2120,171	93: 80, 41, 29	125: 65, 18, 17
30: 627, 338, 505	62: 405, 360, 19	94: 155, 86, 59	126: 395, 104, 101
31: 1227,638, 1075	63: 2447,2200,57	95: 5, 3, 2	127: 4, 1, 1

APPENDIX B: ANALYSIS OF T_K, THE FRACTION OF THE AVERAGE DISSIMILARITIES

The following table shows the average classification accuracy of SDPF when pattern threshold T is selected by varying the fraction of the summation of neighbor patterns' colour differences (T_k).

T_k	Average Accuracy
0.1	10
0.111	15
0.125	32
0.1423	43
0.1667	58
0.2	66
0.25	73
0.3333	81
0.5	63
1	25
1.1111	16
1.25	12
1.4286	12
1.6667	10
2	10
2.5	3
3.3333	1
5	0
10	0

The following figure is the plot of the above data, and it shows the peak accuracy that can be found at the T_k is 0.3333, which is approximately equal to 1/3.



APPENDIX C: NEAREST NEIGHBOUR CLASSIFICATION OF SDPF

The following table shows the results obtained for classifying some categories in Caltech dataset by SDPF using nearest neighbor classification in terms of recall and precision.

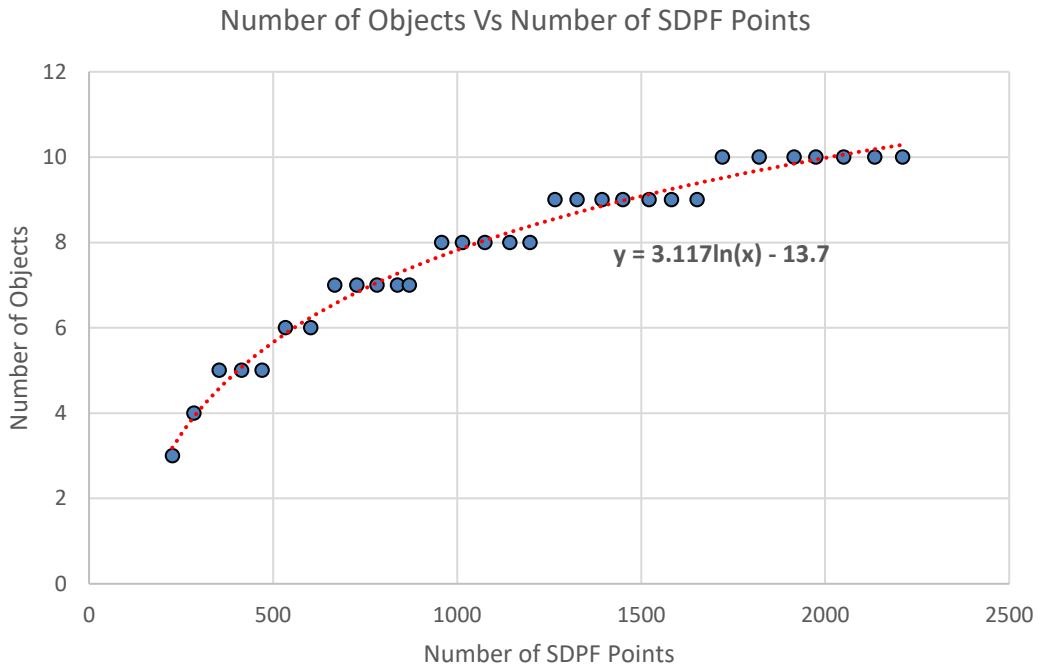
Category	Recall	Precision
face	35	48
Pagoda	34	44
metronome	61	29
accordion	41	40
yinyang	45	24
soccerball	55	14
grand-piano	25	51
headphone	20	45
Ferry	32	62
cougar-face	56	36
Bass	22	17
Ketch	37	35
pyramid	45	27
Rooster	26	50
Laptop	36	30
waterlilly	18	29
Wrench	49	33
strawberry	61	54
Starfish	28	61
ceilingfan	48	51
seahorse	10	34
Stapler	29	47
stop-sign	60	39
Zebra	67	63
Face	54	41
Pagoda	32	65
metronome	56	28
accordion	37	48
yinyang	52	12
soccerball	23	33
grand-piano	44	21
headphone	36	45
Ferry	38	33

APPENDIX D: NUMBER OF OBJECTS VS NUMBER OF SDPF POINTS

The following table shows the data collected from CamVid dataset in order to obtain an equation to predict the number of objects in a frame using the number of SDPF points detected for the same frame. The data is sorted based on the number of SDPF points.

# of SDPF	# of Objects	# of SDPF	# of Objects	# of SDPF	# of Objects	# of SDPF	# of Objects	# of SDPF	# of Objects
226	3	602	6	958	8	1326	9	1720	10
285	4	667	7	1014	8	1393	9	1820	10
353	5	727	7	1075	8	1450	9	1915	10
414	5	782	7	1143	8	1521	9	1974	10
470	5	838	7	1198	8	1582	9	2050	10
533	6	870	7	1266	9	1652	9	2134	10

The following figure shows the plot of above data with the trend line fitted to obtain the equation for the prediction. The dotted line denotes the logarithmic trend of the data.



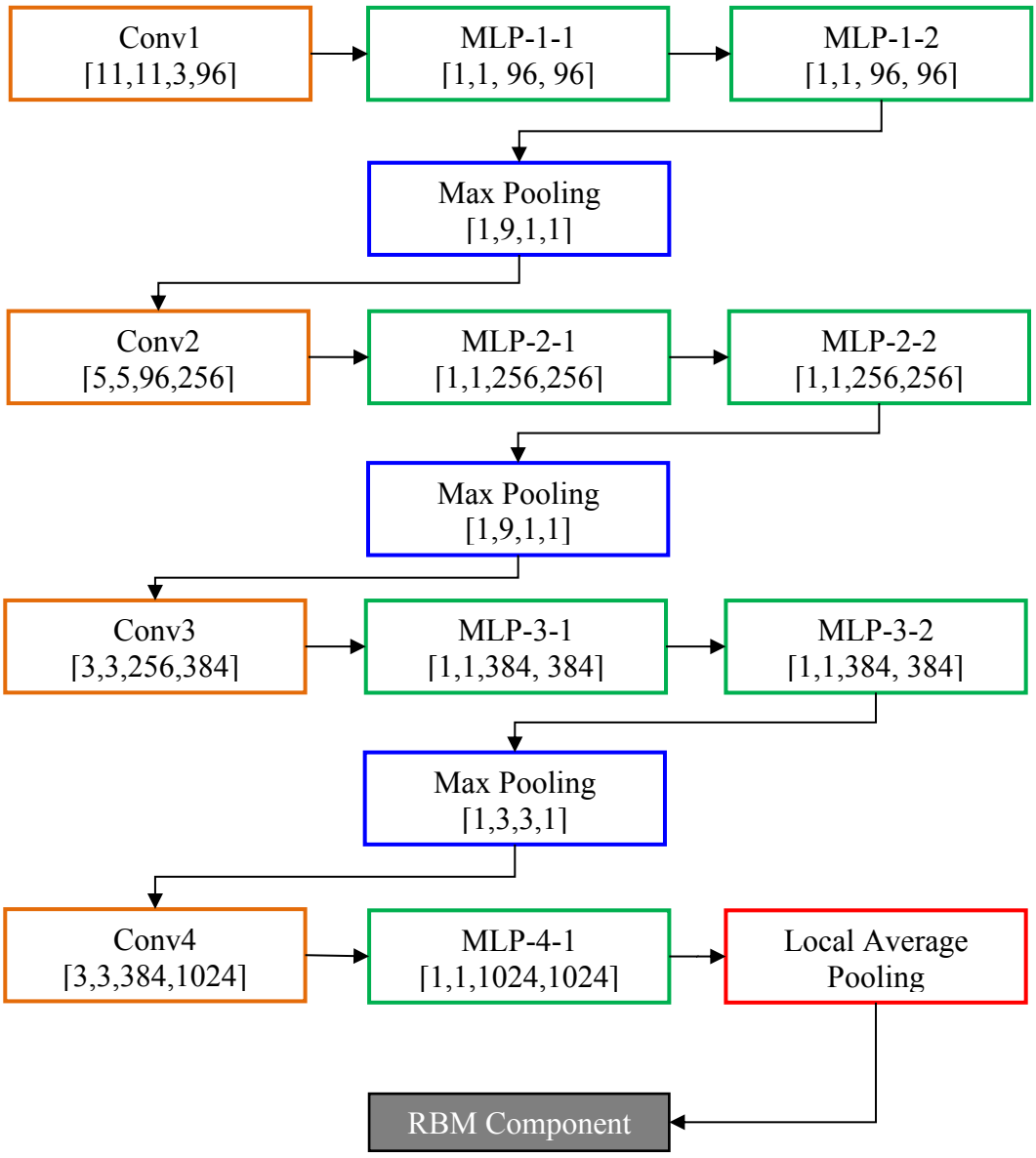
APPENDIX E: PERFORMANCE OF HSDPF ON SEGMENTED OBJECTS

The following table shows the mean average precision (mAP) of recognizing objects after segmenting videos using the proposed RSHC method. The videos were taken from all the sequences from CamVid dataset. 600 manually selected frames were used for this experiment by assigning exactly 50% of frames were used to training the SVM, and the other 50% of frames were used for obtaining this result. Note that this result includes the error that is propagated from the segmentation approach.

Category	**Number of instances found as a percentage (Ground-truth)**	**mAP%**
Road	27	75
Building	23.4	80
Sky	18.7	60
Tree	11	88
Sidewalk	7	78
Car	5.4	98
Column-Pole	0.8	55
Sign-Symbol	1.2	98
Fence	1	88
Pedestrian	0.5	97
Bicyclist	0.3	92
Void	3.7	35

APPENDIX F: MODIFIED NETWORK IN NETWORK MODEL

The following illustration shows the details of the Network in Network model used in this study. Each box denotes a layer with the dimension of the filter in a way that *[width, height, channels, number of filters]*. "Conv" denotes convolution layer where "MLP-X-Y" denotes the Y^{th} Multilayer Perceptron after X^{th} convolution layer. The dimensions of the filters have been decided assuming the size of input image is 224x224. These dimensions should be changed according to the other sizes of inputs.



APPENDIX G: SAMPLE IMAGES OF INPUTS AND INTERNAL RESPONSE OF NIN-RBM

The following are sample input images from HPatches dataset that were used for visualizing the responses from internal layers of the NIN-RBM model, the generated binary codes, and the reconstructed features from the generated binary codes. Note that images have been upscaled.

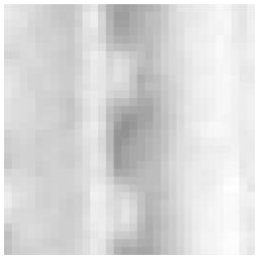


Image 1

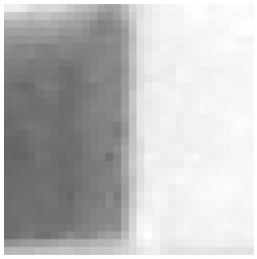


Image 2

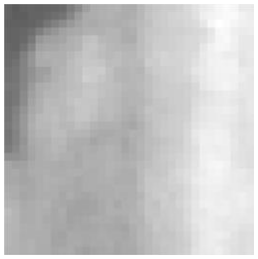


Image 3

Image 4

Image 5

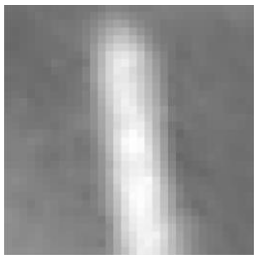


Image 6

Image 7

Image 8

Image 9

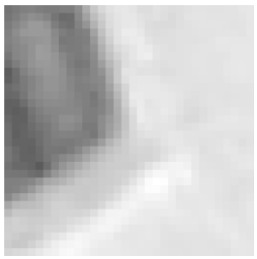


Image 10

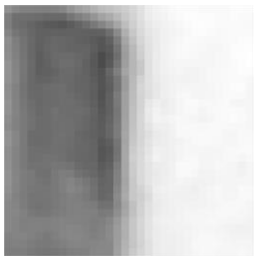


Image 11

Image 12

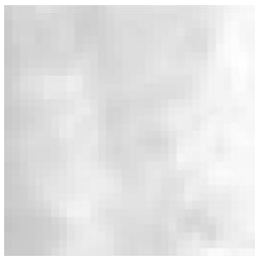


Image 13

Image 14

Image 15

The following images show the feature maps output from the global average layer of the NIN component of the proposed DNN that corresponds to the above-given images.

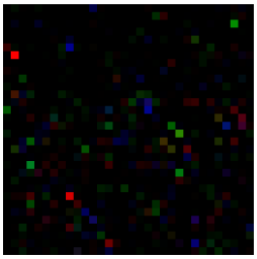


Image 1

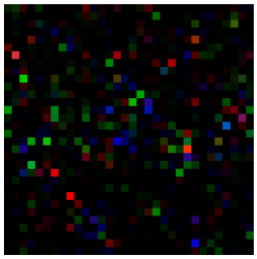


Image 2

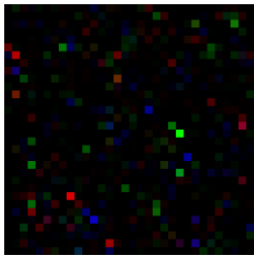


Image 3

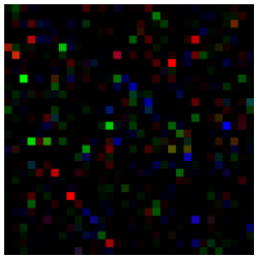


Image 4

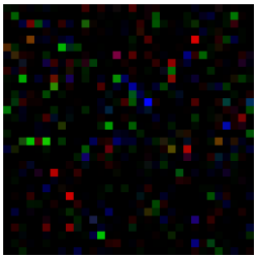


Image 5

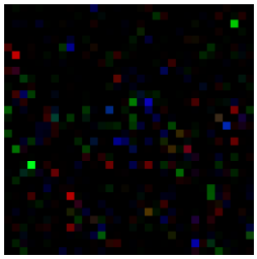


Image 6

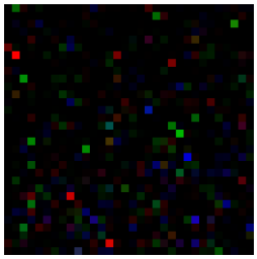


Image 7

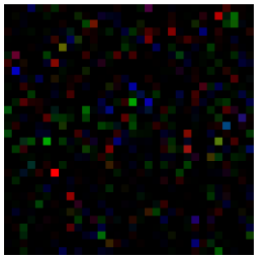


Image 8

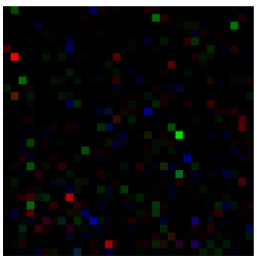


Image 9

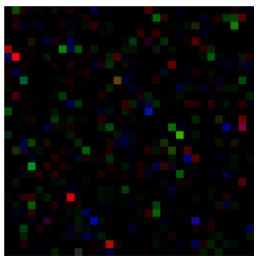


Image 10

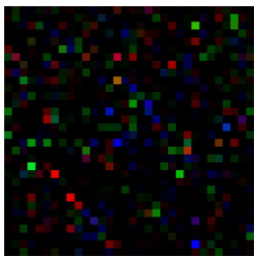


Image 11

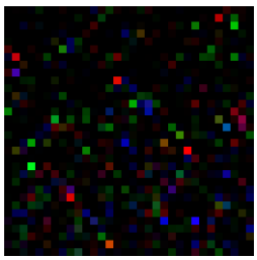


Image 12

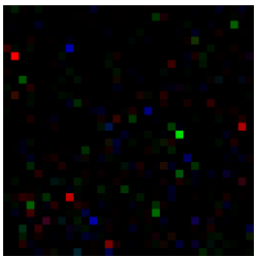


Image 13

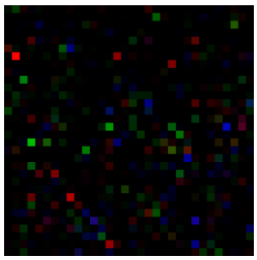


Image 14

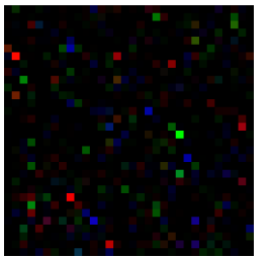


Image 15

The following images show the generated feature maps from the RBM component of the proposed method, for the representative binary codes that have been generated by augmenting the images.

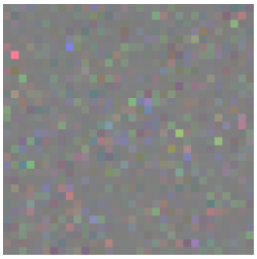


Image 1

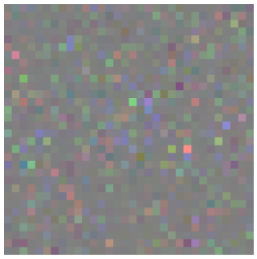


Image 2

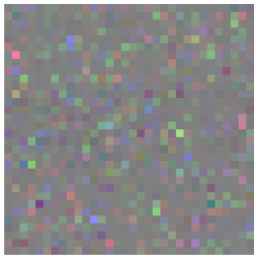


Image 3

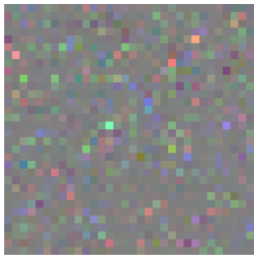


Image 4

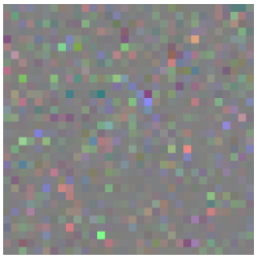


Image 5

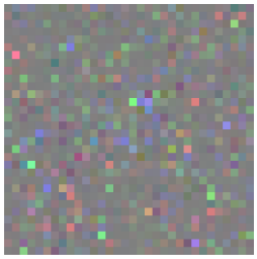


Image 6

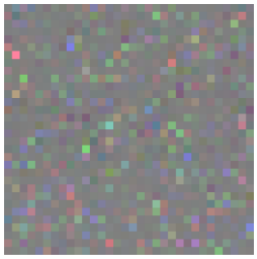


Image 7

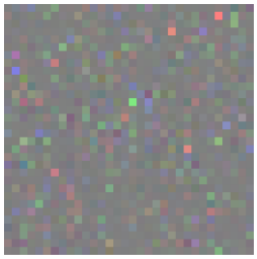


Image 8

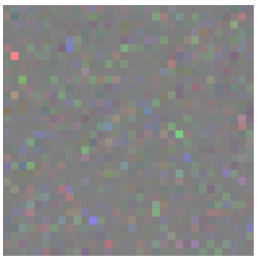


Image 9

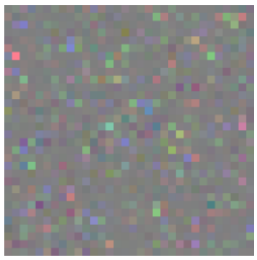


Image 10

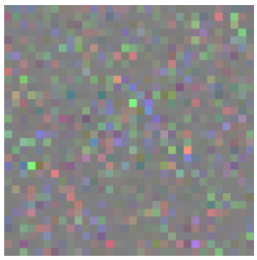


Image 11

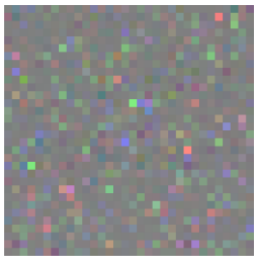


Image 12

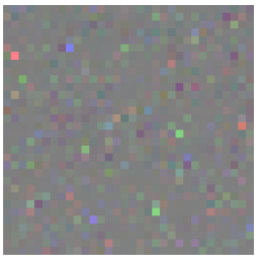


Image 13

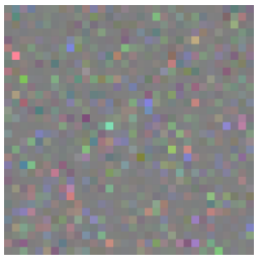


Image 14

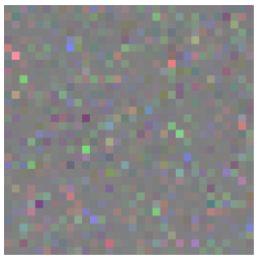


Image 15

The following images show the generated binary codes from the RBM component of the proposed method for each of the input images. Note that this is the direct response from the sigmoid activation without rounding the values; hence, some values are denoted with values in between 0 and 1 (grey pixels). The values are rounded to get the binary code.

Image 1

Image 2

Image 3

Image 4

Image 5

Image 6

Image 7

Image 8

Image 9

Image 10

Image 11

Image 12

Image 13

Image 14

Image 15

The following images visualize the first most convolution filters learned by the NIN component of the proposed DNN. Note that the filters are upscaled for visualization purposes. This is from the convolutional layer that placed before the multi-layer perceptron.


Filter 1

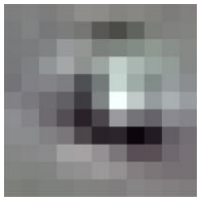

Filter 2

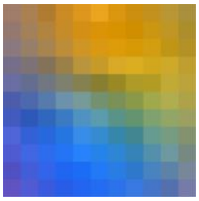

Filter 3

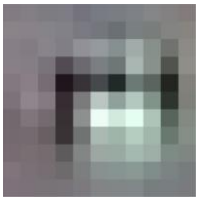

Filter 4

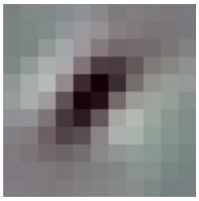

Filter 5


Filter 6


Filter 7


Filter 8


Filter 9


Filter 10

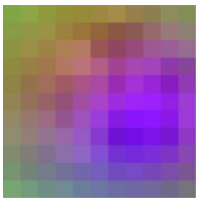

Filter 11

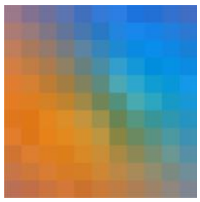

Filter 12


Filter 13

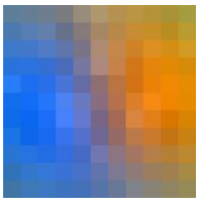

Filter 14


Filter 15


Filter 16


Filter 17

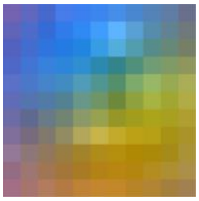

Filter 18


Filter 19


Filter 20


Filter 21

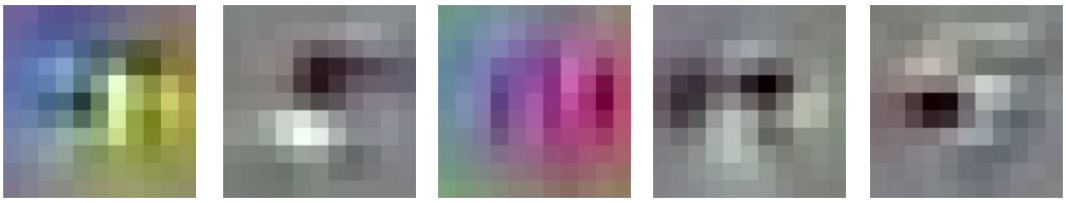

Filter 22


Filter 23


Filter 24

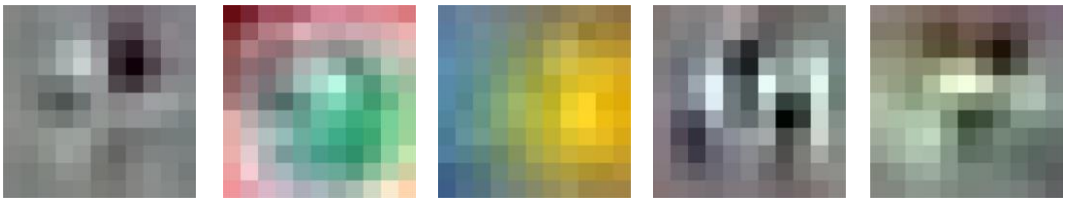

Filter 25

Filter 26 Filter 27 Filter 28 Filter 29 Filter 30

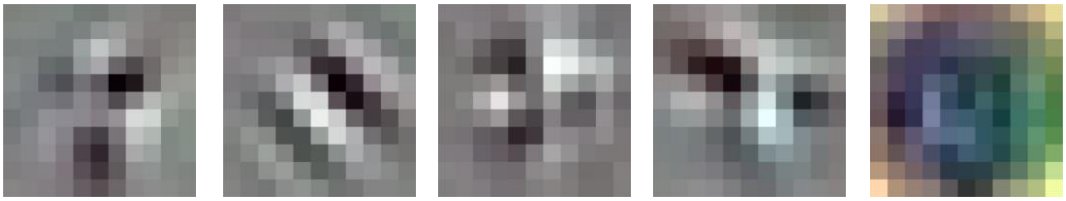


Filter 31 Filter 32 Filter 33 Filter 34 Filter 35

Filter 36 Filter 37 Filter 38 Filter 39 Filter 40



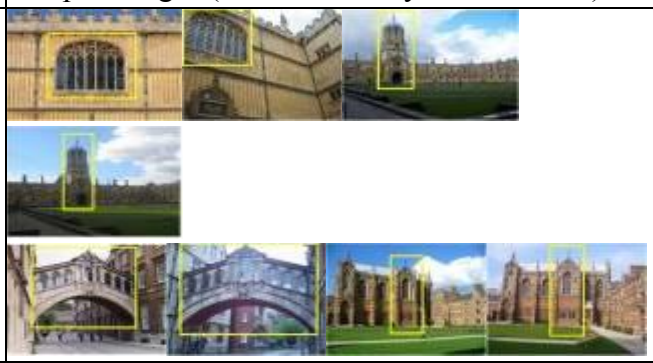
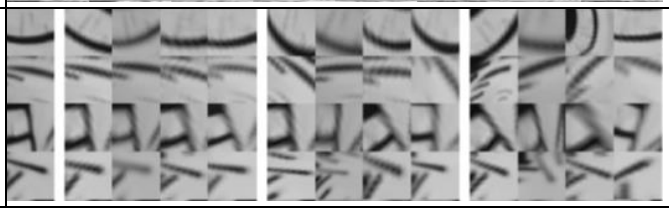
Filter 41 Filter 42 Filter 43 Filter 44 Filter 45



Filter 46 Filter 47 Filter 48 Filter 49 Filter 50

APPENDIX H: SAMPLE IMAGES FROM THE DATASETS

The following table shows sample images of each of the datasets used for evaluating the proposed local binary feature. Some datasets include full images where the others include only the local patches. The datasets which provide full images has their own protocols to select the local patches from the full images.

Dataset	Sample images (resized for easy visualization)	Remarks
Oxford (landmarks)		Full images with key-point centroids and the geometry of patches are provided. The samples are provided as pairs that belong to the same image group.
Paris (landmarks)		
INRIA Holidays		
Brown		Local patches are provided. Consists of 3 subsets 1st row: Liberty, 2nd: Notre dame, 3rd: Yosemite
HPatches		Local patches are provided. First column consists of query patches of each row.

Rome patch		Consists of colour images. key-point centroids and the geometry of patches are provided.

APPENDIX I: PUBLICATIONS BASED ON THIS RESEARCH STUDY

Peer-Reviewed Journal Article

Bandara AMRR, Ranathunga L, Abdullah NA (2020) "Deep Learned Compact Binary Descriptor with a Lightweight Network in Network Architecture for Visual Description". The Visual Computer (TVCJ). https://doi.org/10.1007/s00371-020-01798-5

Bandara AMRR, Ranathunga L., Abdullah NA (2019) "Nature Inspired Dimensional Reduction Technique for Fast and Invariant Visual Feature Extraction". International Journal of Advanced Trends in Computer Science and Engineering (IJATCSE) 8:696–706. https://doi.org/10.30534/ijatcse/2019/57832019

Book Chapter

Bandara AMRR, Ranathunga L, Abdullah NA (2014) "Visual Feature Clustering Using Temporal, Colour and Spatial Information". In: Electronics, Communications and Networks IV. CRC Press, Beijing, China, pp 677–682

IEEE Indexed Conference Publications

Bandara AMRR, Ranathunga L, Abdullah NA (2015) "A feature clustering approach based on Histogram of Oriented Optical Flow and superpixels". In: 2015 IEEE 10th International Conference on Industrial and Information Systems (ICIIS). pp 480–484

Bandara AMRR, Ranathunga L, Abdullah N (2013) "Invariant properties of a locally salient dither pattern with a spatial-chromatic histogram". In: Industrial and Information Systems (ICIIS), 2013 8th IEEE International Conference on. IEEE, pp 304–308