Graz University of Technology
Institute of Electrical Measurement
and Measurement Signal Processing
A-8010 Graz, Austria

Master's Thesis

Dynamic Scene Recognition with Oriented Spacetime Energies

Christoph Feichtenhofer

November 5, 2013

Thesis Supervisors:

Prof. Dr. Axel Pinz

Graz University of Technology, Austria

Prof. Dr. Richard P. Wildes

York University, Toronto, Canada

Abstract

This work presents two main contributions for the computational understanding of dynamic scenes. The first contribution are spacetime forests defined over complementary spatial and temporal features for naturally occurring dynamic scene recognition. The approach is focused on fast execution rates for online classification of natural scenes. There are three key novelties in this work. The first is the introduction of a novel descriptor that exploits the complementary nature of spatial and temporal information. Second, a forest-based classifier is used to learn a complementary multi-class representation of the feature distributions. This aspect supports high class discrimination with great computational efficiency. Third, the video is processed in temporal slices with scale matched preferentially to scene dynamics over camera motion. Slicing allows for efficient, incremental processing by evaluating the input sequence online, with increasing confidence for longer temporal inputs. Further, slicing enables handling temporal alignment as latent information in the classifier, which leverages the high temporal diversity in the spacetime texture patterns.

The second contribution is the Bags of Spacetime Energies (BoSE), a unified bag of visual word (BoW) framework for dynamic scene recognition. In particular, this part builds on densely sampled features that uniformly capture the spatial and temporal structure of the video with oriented filter energies. Following the recent image classification literature, where BoW approaches are well established to achieve good performance, a number of feature encoding methods are evaluated for BoW-based spatiotemporal scene classification. Next, by building on encoded bags of spacetime energies, it is shown that global motion compensation can improve performance on scenes captured with significant camera motion, though at the cost of decreased performance on scenes captured from a static camera. A novel feature pooling method is introduced to pool the encoded spacetime features based on their temporal energy in the frequency domain. This dynamic pooling approach especially increases performance when camera motion is present, but does not compromise performance when camera motion is absent.

The proposed methods are experimentally validated on two publicly available dynamic scene datasets to document their outstanding performance. A substantial improvement on the previous state-of-the-art is achieved, with an increased robustness to camera motion where previous approaches have experienced difficulty. Specifically, the presented BoSE framework outperforms the previous state-of-the-art by improving the classification accuracy by 20% and 19% on the two respective datasets.

The insights of this thesis can have a substantial influence on the design of dynamic scene classification approaches. More generally, the outstanding performance of the presented spacetime recognition framework suggests application to a variety of other areas, such as event retrieval, video indexing, or object and activity localization.

Kurzfassung

Diese Arbeit präsentiert zwei bedeutende Beiträge für das maschinelle Verstehen von dynamischen Szenen. Der erste Hauptteil der Forschungsarbeit beschreibt Spacetime Forests, definiert über räumliche und zeitliche Bildmerkmale, zur Erkennung von natürlich auftretenden dynamischen Szenen. Der Fokus dieses Ansatzes ist auf das Erzielen schneller Laufzeiten gerichtet. Es werden drei wichtige Neuerungen vorgestellt: Erstens wird eine neuartige Beschreibung eingeführt, welche das komplementäre Wesen von räumlichen und zeitlichen Bildmerkmalen ausnützt. Zweitens wird ein spezieller Random Forest Klassifikator verwendet um eine komplementäre Mehrfachklassen-Repräsentation aus den Bildmerkmalen zu lernen. Dies erlaubt eine effektive Diskriminierung der Klassen bei gleichzeitig hoher Effizienz. Drittens werden die Bildmerkmale aus zeitlichen Teilen der Eingangssequenzen extrahiert um mit den zeitlichen Merkmalen vorzugsweise die dynamischen Aspekte der Szene, anstelle der Kamerabewegung, zu erfassen. Außerdem ermöglicht das Aufspalten in zeitliche Teile eine inkrementelle Klassifizierung der Eingangssequenz mit steigender Genauigkeit über die Zeit. Ein weiterer Vorteil ist die Möglichkeit die Vielfältigkeit der Raum-Zeit Texturen aus den Sequenzen zu erlernen.

Der zweite Schwerpunkt dieser Arbeit ist die Entwicklung eines einheitlichen bag-of-visual-word (BoW) Systems zur Erkennung dynamischer Szenen. Dieser Teil baut auf dicht extrahierte lokale Bildmerkmale, welche mithilfe orientierter Filter die zeitliche und räumliche Energie der Videos modellieren. Die primitiven Bildmerkmale werden in eine, für dynamische Szenenerkennung effektive, Zwischenrepräsentation codiert, welche anhand einer systematischen Evaluierung populärer Codierungsmethoden bestimmt wird. Darüber hinaus wird gezeigt, dass die Verwendung von Kamera-Stabilisierungsmethoden die Erkennungsrate für Videos mit Kamerabewegungen zwar verbessert, beim Erkennen von statisch aufgenommenen Szenen jedoch zu einer Verschlechterung führt. Abschließend wird ein neuartiges Konzept zur Sammlung der codierten Raum-Zeit Merkmale, basierend auf der zeitlichen Energie im Frequenzbereich, eingeführt. Dieser Ansatz erhöht die Erkennungsrate besonders bei dynamischer Kameraführung, beeinträchtigt die Resultate aber nicht falls keine Kamerabewegung vorliegt.

Die vorgestellten Methoden werden auf zwei öffentlichen Datensätzen experimentell validiert. Hierbei wird eine erhebliche Verbesserung der Erkennungsrate, verglichen mit dem derzeitigen Stand der Technik, demonstriert. Im Detail beträgt die absolute Steigerung der Genauigkeit auf den beiden Datensätzen 20% bzw. 19%.

Die Erkenntnisse dieser Arbeit können einen erheblichen Einfluss für das Entwerfen von Algorithmen zur dynamischen Szenenerkennung haben. Darüber hinaus schlagen die herausragenden Resultate der präsentierten Ansätze zur Raum-Zeit Erkennung weitere verwandte Anwendungen vor, wie etwa das Erkennen von Ereignissen, Objekten oder Aktivitäten.

Deutsche Fassung:
Beschluss der Curricula-Kommission für Bachelor-, Master- und Diplomstudien vom 10.11.2008
Genehmigung des Senates am 1.12.2008

EIDESSTATTLICHE ERKLÄRUNG

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Graz, am …….. ……..
(Unterschrift)

Englische Fassung:

STATUTORY DECLARATION

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

…….. ……..
date (signature)

Contents

List of Figures

Acknowledgements

I would like to express my deep gratitude to my advisor Axel Pinz for his invaluable guidance and support. In our fruitful discussions he always gave me new insights when I was struggling to pursue my ideas and with his deep and broad knowledge of science he taught me how to write a paper and give a clear presentation. I am also grateful for his fundamental support that made my research stay at the York University Toronto possible. Thanks to Richard P. Wildes for hosting me in the Vision Lab at York University. I am very fortunate to have him as my co-advisor. His boundless enthusiasm for computer vision research and his ability to keep a clear overview, even for very complex problems, were a key factor for my progress during my time in Toronto. Special thanks to all my friends and colleagues for making my time as a graduate student very enjoyable and memorable. Finally, I want to profoundly thank my parents who made all this possible with their unconditional support.

1 Introduction

For humans it seems like a trivial task to recognize and interpret the rich visual world where they find themselves. For a computer, however, the automatic categorization of complex natural scene types is a fundamental research problem. Computer vision researchers have developed algorithms for representing and learning the complex source of visual information in natural scene images for several decades. Still, no extant automated system rivals the level of performance achieved by humans.

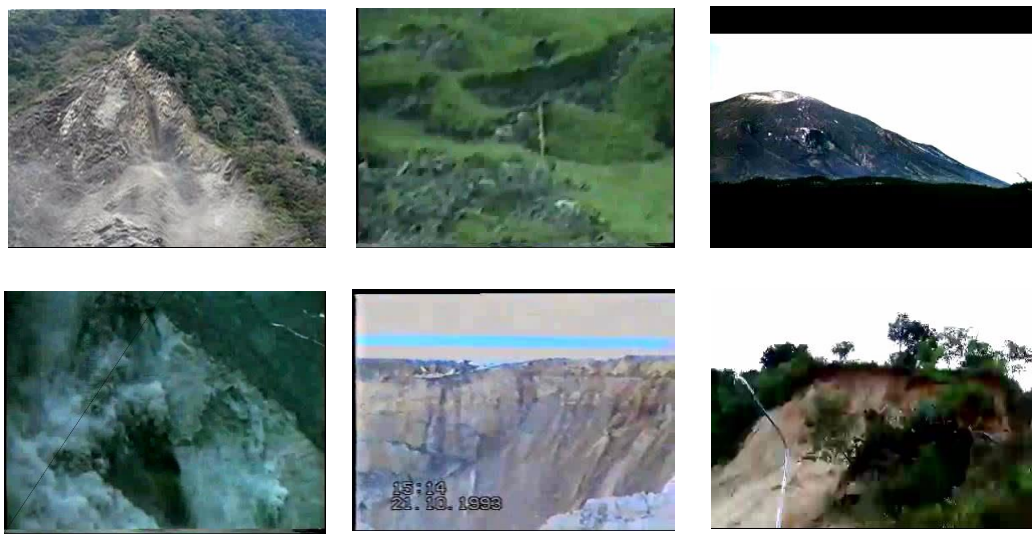
The task of scene categorization is to find the categories (*e.g.* beach, city, river) to which the input sequence belongs. Humans are able to perform this task with speed and accuracy [82, 84] and with little attention to the objects present in the scene [61]. Such a holistic understanding of the scene is also pursued by popular representations and algorithms for scene categorization [32, 58, 73], where local features are used to describe a complex scene straightforwardly, without intermediately extracting semantics of the objects in the scene.

The modeling of visual objects in space and time is a crucial component for a wide range of applications including the representation of dynamic scenes. Beyond the scientific interest in dynamic scene classification, many useful applications have emerged. With the fast expansion of video data present in the Internet (*e.g.*, YouTube) and the increasing amount of video-enabled devices (*e.g.*, smartphones), a huge amount of data is generated daily. Therefore, human inspection becomes impossible in these situations. Hence, automatic systems for recognition and organization of video are in high demand.

Such systems could further be helpful in surveillance or safety applications; *e.g.* cameras monitoring spacetime events such as forest fires or avalanches. Although this increasing demand has triggered recent research activity, state-of-the-art classification frameworks are still far from human recognition performance. The amount of information and variability present in images of diverse natural scenes calls for an approach that is able to handle multiple classes, scales and temporal variations, yet still be efficient in training and recognition.

Several aspects can make the automatic recognition of dynamic scenes a difficult task. One can imagine that videos of the same scene may look very different when captured from different camera viewpoints, variations in illumination, (motion) clutter, etc. Furthermore, the variability in visual appearance and dynamics of scenes from the same category may be significant. Figure 1.1 illustrates such intra and inter-class variations taken from the Maryland [90] and YUPENN [27] dynamic scene datasets. Sequences from three different classes are depicted in Figure 1.1(a), where very little difference in spatial appearance can be observed. Hence, a distinction between these classes, based on spatial appearance only, becomes very challenging. Further sequences, showing different instances of a landslide class, are illustrated in Figure 1.1(b). Due to the large variations within the class, learning a model that represents common, distinctive properties of these sequences is a very difficult task on its own.

(a) Images from three different classes with similar appearance.



(b) Images from landslide sequences with large differences in appearance.

Figure 1.1: Examples for low inter class differences (a) and large intra class variations (b) from the YUPENN (a) and the Maryland (b) datasets.

The goal of this thesis is the analysis and modeling of spatiotemporal image structures for recognition of natural world sequences. Scenes are recognized on the basis of their image spacetime appearance, *e.g.*, as forest fire vs. beach vs. city. The thesis is organized as follows. In the next chapter, related work on the representation and classification of dynamic natural scenes is given. Subsequently, the thesis introduces two different representations for dynamic scenes. A novel method for dynamic scene recognition with complementary features used by spacetime forests is presented in Chapter 3. This representation describes each temporal subset (*i.e.* slice) of a video with a single feature vector, capturing complementary histograms of spatial and temporal filter responses, as well colour distributions. The focus of this approach is on fast, online processing of video. Efficiently extracted features are directly pooled in a vector representation and classified using a decision tree classifier. Chapter 4 introduces a codebook-based approach for dynamic scene recognition. Local spacetime regions are represented by single feature vectors and decomposed into visual codewords to provide a large degree of invariance to intra-class variations in appearance. A sequence is subsequently represented by spatiotemporal aggregation of the mid-level feature codes and classified using discriminative one-vs-rest classifiers. Conclusions and ideas for further improvement are given in Chapter 5.

Interestingly, a striking result of this work is that only a very small amount of temporal information is necessary to achieve state-of-the-art performance in dynamic scene classification.

2

Related Work

While static scene recognition from single images has been researched extensively (*e.g.* [32, 58, 62, 65, 73, 83, 94, 96, 102]), relatively little research has considered video-based dynamic scene recognition [27, 67, 90], even though the availability of temporal information should provide an additional means for classifing scenes visually. The reason for the relatively small amount of previous work in the dynamic scene recognition area might be that the task of dynamic scene recognition requires new descriptors, since popular temporal features such as optical flow or spatial features such as gradient histograms, are not able to capture the dynamic texture information properly [27, 29]. On the other hand, static scene recognition relies on the large pool of previous work on image descriptors. Another reason for the slow progress in this area might be the lack of a large and well-designed dataset of dynamic scenes; however, this problem has recently been addressed by Derpanis *et al.* in [27], where a large and diverse database of dynamic scene categories is proposed.

2.1 Features for dynamic scene recognition

The problem of dynamic scene recognition was first tackled by Marszalek *et al.* in the context of human action classification [67]. Since, in realistic videos, human actions are often correlated with the scene classes, they show that automatically extracted context of natural dynamic scenes can improve action recognition. Similar, as in their work in [53], Laptev *et al.* attempt to recover the motion from optical flow measurements and consequently use histograms of optical flow to describe human actions and scene dynamics. While these features have proven to be effective for action categorization [53, 81], Derpanis *et al.* [27] have shown that optical flow achieves low performance for modelling the dynamics of natural scenes. The reason is assumed to be that the optical flow constraint, *i.e.* the brightness constancy assumption, does not hold for dynamic patterns exhibiting specularities and flickering lighting, *e.g.*, in textures of water, fire, or lightning.

Doretto *et al.* [30] have used linear dynamical systems to model successfully the stochastic properties of dynamic textures. Limited by the first-order Markov property and linearity assumption, this model has shown poor performance for dynamic scene classification in [90]. Shroff *et al.* [90] propose a method with fuses static and dynamic features in a chaos-theoretic system to classify "in-the-wild" dynamic scenes.

Due to the lack of appropriate for evaluation, Shroff *et al.* [90] also introduced a dataset, consisting of amateur footage from the Internet. All the videos have been collected from Youtube, except for the "Boiling Water" sequences that have been taken from the DynTex database http://projects.cwi.nl/dyntex/index.html. Therefore, these videos include camera motion and even scene cuts. As the videos contain camera movement only for some classes, *e.g.*, tornado or avalanche, while other classes are free from camera movement, *e.g.*, street traffic or fountain, it is not clear if the algorithms capture temporal regularities introduced by camera motion and scene cuts, or the dynamic attributes of the scene.

With the purpose of isolating temporal dynamics of the objects and surfaces in the scenes from the movement induced by the camera, Derpanis *et al.* [27] present a new dataset with stabilized camera settings. Furthermore, they systematically investigate the impact of low-level feature representations on dynamic scene classification. By comparing spatial appearance, temporal dynamics and joint spatial appearance and dynamic features they conclude that using features that jointly model spatial appearance and temporal

dynamics provided overall best performance for recognizing dynamic scenes.

Most recently, Theriault *et al.* [95] proposed an approach based on slow feature analysis (SFA) [109]. They learn motion features from filter responses that are reputed to model primate V1 cortical operations, as they result from local maxima of spatially oriented, multiscale Gabor filters [86]. The slowest varying features among these are identified by computing their temporal derivatives, and are coded using a trained dictionary. After encoding, the slow features are pooled into a feature vector by applying max-pooling to the whole video in spatial pyramidal regions. A linear support vector machine is subsequently used to classify the videos.

The approach proposed in the present thesis is based on local measurements of orientation. This feature type has been used for both static [58, 73, 94] and dynamic [27] scene classification. In using local orientation measurements that have been aggregated into texture patches, these approaches build on research in both static [4] and dynamic [29] texture analysis that use similar primitives. Application of such measurements to dynamic image analysis additionally has been shown useful in a variety of areas, perhaps most related to current concerns are image motion estimation [1, 38, 91] and human action characterization [28]. While their previous application to dynamic scene recognition has led to state-of-the-art performance, it also has shown notable limitations when confronted with significant camera motion [27].

2.2 Datasets for dynamic scene recognition

As mentioned in the previous section, concomitant to the ongoing progress in representations, new challenging datasets, reflecting real-world scenes, have been introduced. These datasets provide sequences, which present scenes with high intra-class variability; *e.g.* various shapes, poses, and appearances, with diverse illumination and foreground clutter.

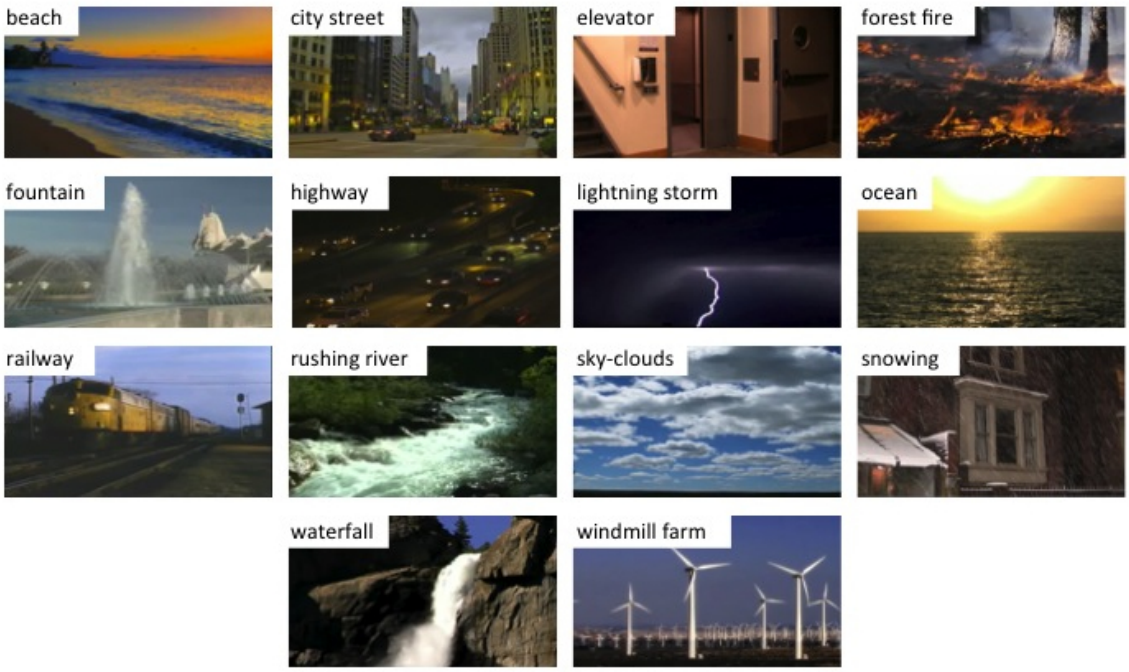
Currently there exist two publicly available datasets for natural dynamic scene classification. The "Maryland in-the-wild" [90] dataset consists of amateur videos from the Internet and therefore is confounded with unconstrained camera movement. On the other hand, the "YUPENN Dynamic Scenes data set" [27] consists of scenes recorded from a stabilized camera setting. The algorithms proposed in this thesis are evaluated on both of these datasets. Figure 2.1 shows examples of the two datasets and Table 2.1 further compares the datasets in various aspects.

	Maryland "In-The-Wild"	YUPENN Dynamic Scenes
Number of classes	13	14
Number of videos per class	10	30
Camera movement	unconstrained	stabilized
Scene cuts	yes	no
Average resolution	308×417 pixels	250×370 pixels
Average duration	617 frames	145 frames
Standard dev. of duration	531 frames	21 frames

Table 2.1: Properties of the Dynamic scene datasets used in this thesis.

It is also notable that the Maryland dataset contains large variations in video duration, while all videos in the YUPENN dataset have approximately the same duration. Both datasets contain large intra-class variability, with the Maryland dataset exhibiting extreme intra-class differences for some specific classes only, *e.g.*, avalanche and landslide. Moreover, since the videos in the datasets are collected from various sources, they vary in terms of resolution, framerates, scale, illumination, and camera viewpoint. Such circumstances additionally challenge the recognition algorithms.

(a) Maryland "In-The-Wild" [90]



(b) YUPENN Dynamic Scenes data set [27]

Figure 2.1: Sample scenes from the Maryland (a) and YUPENN (b) datasets.

2.3 Bag of visual word representations

Previous work, *e.g.*, [25, 93], has shown that the use of intermediate representations improves performance in classification tasks. Bag-of-word (BoW) methods were initially proposed for text retrieval systems [47] and later on adopted for visual classification methods [60, 93]. These approaches first build a dictionary of visual words (*i.e.* codewords), which represents a visual vocabulary to describe the local appearance of objects, images or image sequences. To classify unknown inputs, the appearance of a new query is described by the visual words and recognized by using a trained classifier on the visual vocabulary. In scene categorization, the idea is motivated by the success of similar techniques in classifying image textures as distributions of so-called textons. Intermediate texton representations are generated from various descriptors of local image appearance. For example, Leung and Malik [60] create textons by quantizing filterbank responses, Varma and Zisserman [99] show that describing small local patches is sufficient, and Lazebnik *et al.* [57] apply affine covariant detectors to account for viewpoint changes and non-rigid deformations.

Certainly, the BoW methodology seems very appealing for situations where categories are sought. Contrary to specific instance search, the categorization process happens at a very generic level, where large diversity may occur among specific instances of a class. Representing instances by using a discrete number of visual words assures this generality. Robustness to viewpoint changes, occlusions, clutter and other deformations (*e.g.*, spatial translations) is given by modelling the input signal as an orderless distribution of the codewords. Consequently, the BoW representation does not capture any information of the spatial layout of the visual words. However, for modelling scenes, spatial information can be very descriptive and should not be discarded entirely. For example, a visual word describing a sky region is expected to be on top of the image. Enhanced BoW representations have been developed with generative models [7, 9, 32, 54], discriminative visual vocabularies [49, 70, 112] and geometric verification [3, 56]. The most popular technique for enhancing BoW methods, however, is spatial pyramid matching (SPM). To add spatial information Lazebnik *et al.* [58] partition the image into increasingly fine sub-regions. By concatenating the BoW-histograms for each sub-region grid cell they include coarse information about the spatial arrangement of the visual words. This method has been highly successful in image classification and has triggered the proposal of many succes-

sive methods relying on this concept. A general SPM framework can be described by three successive steps [104, 113] (see Figure 4.1 for an illustration):

- *Feature extraction*: In this stage low-level descriptors are applied at interest point locations or regular locations in a dense grid. Frequently used descriptors are GIST [73], SIFT [63], HOG [26], colour moments [115], spacetime extensions of image features, *e.g.*, HOG3D [50], 3DSIFT [85], local trinary patterns [114], spatiotemporal oriented energy (SOE) descriptors [106], as well as optical flow features. Some interest point detectors include Harris/Hessian corners with respective spacetime extensions [52, 107].

- *Coding:* This step transforms local descriptors into codes with desirable properties such as compactness, sparseness or statistical independence [10]. A trained codebook is applied to each feature point to quantize the descriptors either by hard vector quantization (a code with only one non-zero component, *i.e.* one visual word) [58] or by soft vector quantization that assigns several codes, either by focusing on sparsity [111] or locality [104]. Codebooks are typically created by simple K-means clustering.

- *Pooling:* The codewords are collected from local sub-regions and summarized in a histogram. Several neighbouring cells at different levels (*i.e.* grid sizes) are used to collect the local codes based on averaging [58] or max-pooling [104, 111] within a spatial cell. Subsequently, the pooled feature encodings, *e.g.*, histograms counting the visual word occurrences for average pooling, are concatenated into a final feature vector that describes the visual input.

Improved coding [10, 19, 104, 111] and pooling strategies [11, 14, 34, 46] have been extensively researched over the past years, with particular success at increasing classification performance. The proposed work here focuses on feature extraction and pooling and uses well-established methods for coding [58, 78, 104].

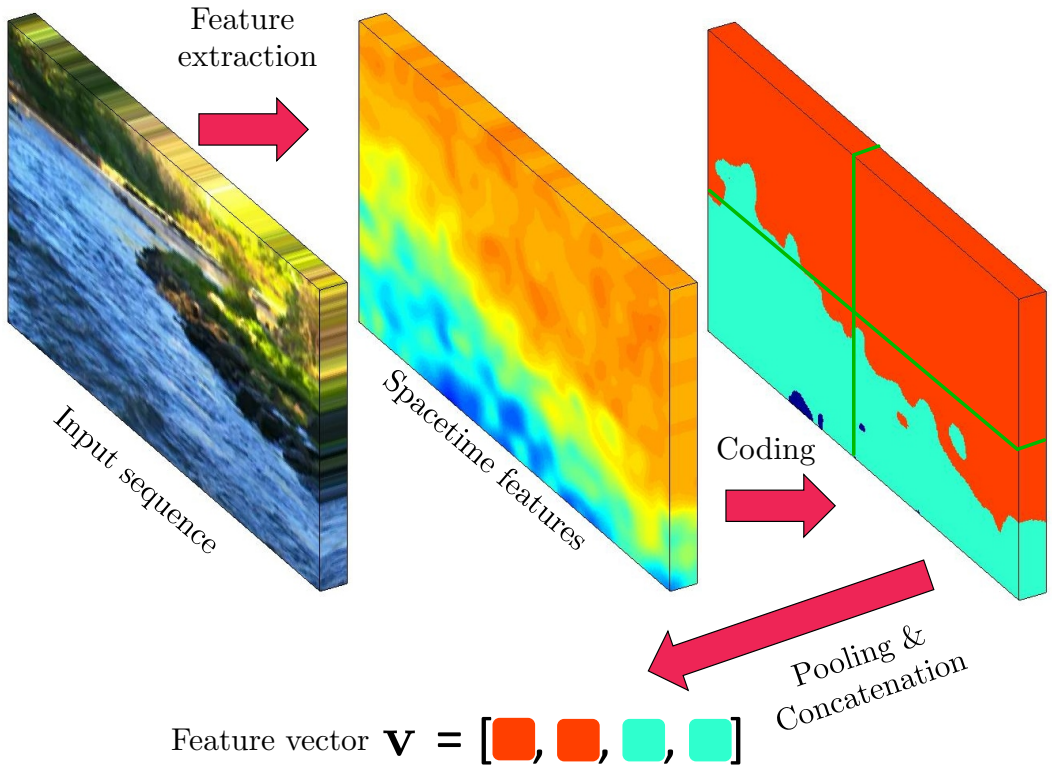


Figure 2.2: Feature vector extraction in a local coding and spatial pooling scheme.

2.4 Learning scenes from features

Several machine learning methods are applicable for classification tasks in computer vision. The three most prominent supervised learning methods are (i) Support Vector Machines (SVM) [22], that construct a maximum-margin hyperplane in a high-dimensional space to separate the data linearly; (ii) Boosting [36], which combines many simple classifiers, based on weights for emphasising poorly classified examples; and (iii) Decision Forests [23], which consist of an ensemble of decision trees, where each tree recursively splits the input data for classification.

Although the classifier choice does not have the same impact on recognition performance as a carefully designed feature representation, it has an exceptional influence on the classification speed. Generally, non-linear classifiers, such as SVMs with non-linear kernel functions, provide the best results, but at cost of a long processing time. On the other hand, linear classifiers, such as SVMs with linear kernels, Boosting, or decision forests provide a good trade-off between classification speed and performance. Therefore, if speed is important, linear classifiers are preferred for most applications; further, a recent trend has been to linearly approximate non-linear kernels for a better trade-off [101].

In this thesis, two specific instantiations of machine learning concepts are used. In the approach presented in Chapter 3 weak learners are used in the spatial, temporal and chromatic domain explicitly. By combining many simple classifiers, a Boosting variant (*e.g.*, AdaBoost [36]) or decision tree classifiers (*e.g.*, Random Forests [12]) are well suited for this task. The fundamental learning algorithm in this chapter is chosen to be a Random Forest, since this classifier is very fast to train and evaluate, whereas AdaBoost is relatively slow in the training stage.

The second approach presented in this thesis is a codebook-based BoW method. It has been shown that BoW models combined with histogram intersection SVM kernels achieve high classification accuracy [58] when used with vector quantized codes. Similar as in the popular spatial pyramid approach [58], the work presented in Chapter 4 uses the pyramid match kernel of Grauman and Darrell [39] to compare feature sets, encoded by vector quantization. For sparse coded features, however, linear SVM kernels achieve even better performance [111] and therefore are applied.

3

Spacetime Forests with Complementary Features for Dynamic Scene Recognition

This chapter first introduces a novel descriptor to capture the appearance and motion of natural scenes (Section 3.2). These complementary features allow the representation and classification of the scenes to be suited to the classes. Use of feature complementarity is useful because some classes may be better represented and distinguished by a specific type of feature; *e.g.*, a street traffic sequence may be best represented by motion information, while a forest fire sequence is better represented by appearance information. The second part of the chapter (Section 3.3) introduces a specific random forest instantiation that classifies the sequences in an incremental, bottom up manner. This approach allows fast incremental predictions with increasing confidence over time. Finally, Section 3.5 empirically evaluates the quality of the proposed method.

3.1 Contributions

In the light of previous research, the present work makes three main contributions. First, a novel descriptor is presented that encodes several frames of a video into complementary information. Separate spatial and temporal orientation measurements are aggregated in spacetime pyramids. Distinct from previous application of spatiotemporal orientation to dynamic scenes [27], separation of spatial and temporal orientation allows those components to be differently weighted in classification. Note that, in contrast to the features proposed in this chapter, the BoW approach presented in Chapter 4 builds on spacetime features that uniformly capture spatial and temporal orientation structure.

The second contribution is a specific instantiation of a random forest classifier, applied to dynamic scene recognition. This spacetime forest allows for automatic determination of the most discriminative features to separate the classes based on appearance and dynamics with computational efficiency. The approach allows the classifier to learn different weights for different class discriminations; *e.g.* a beach sequence may be better represented by its motion information, while a forest fire sequence might be better distinguished by its spatial appearance.

The third contribution is the processing of video in incremental temporal slices in a bottom up manner with scale matched preferentially to scene dynamics (in comparison to camera motion). This strategy allows for temporal alignment to be treated as latent in the classifier, efficient processing and robustness to large temporal variation across time (*e.g.* from camera motion), even while capturing intrinsic scene characteristics. Previous dynamic scene research has suffered in the presence of camera motion [27] and has provided little consideration of on-line processing concerns.

The approach has been evaluated on two publicly available datasets [27, 90], and has been presented at the British Machine Vision Conference 2013 [33]. Results show that it achieves a new state-of-the-art in dynamic scene recognition. Figure 3.1 gives an overview of our multi-class recognition framework.

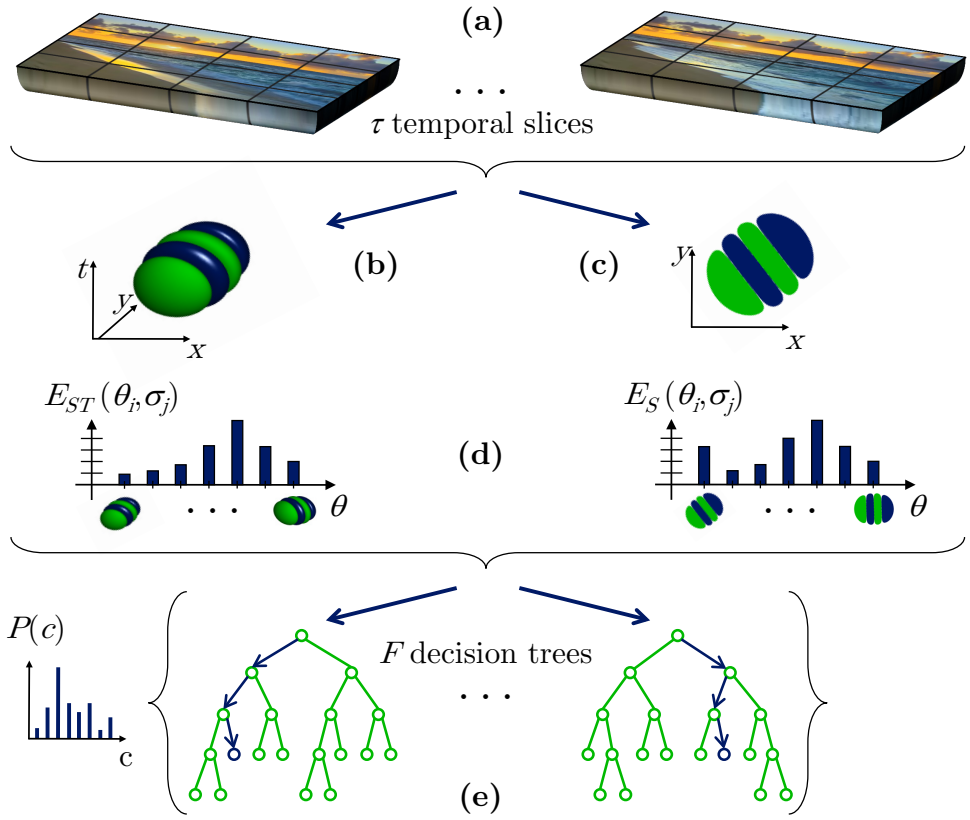


Figure 3.1: Overview of the proposed dynamic scene classification framework. (a) The input sequence is divided into cuboids using a spatiotemporal pyramid representation. τ temporal slices are created to process the frames in a sliding window approach. (b,c) The cuboids are filtered by banks of multscale, σ, oriented filters along equally separated directions, θ, in image spacetime (b) and space (c) to capture both dynamic and static appearance information. (d) Filter responses cast weighted votes in spacetime orientation cells. (e) The class of each temporal slice is determined via likelihood voting, using a multi-class random forest classifier. Subsequently, all slice-based classifications are combined across the entire input.

3.2 Complementary spacetime orientation descriptor

This section puts forth a novel descriptor for dynamic scene representation that is based on the complementary combination of several different primitive measurements. Spatially oriented measurements are used to capture static image appearance and are combined with spatiotemporally oriented measurements to capture image dynamics. Filtering operates at multiple scales to capture the multiscale characteristics of natural scenes. Furthermore, colour channels are included to capture complementary chromatic information. Interestingly, evidence from biological systems suggests that they exploit similar complementary feature combination in their visual processing [31, 37, 41, 76, 103].

3.2.1 Spatial information

Oriented spacetime energy measurements are used as building blocks of the descriptor. Spatial appearance information is extracted via application of multiscale filter banks that are further tuned for spatial orientation. In the spatial domain, 2D Gaussian third derivative filters (as shown in Figure 3.2),

$$G_{2D}^{(3)}(\theta_i, \sigma_j) = \kappa_{2D} \frac{\partial^3}{\partial \theta_i^3} \exp\left(-\frac{x^2+y^2}{2\sigma_j^2}\right), \tag{3.1}$$

with θ_i denoting orientation, σ_j scale, and κ_{2D} a normalization constant, are applied to yield a set of multiscale, multiorientation measurements according to

$$E_S(\mathbf{x}; \theta_i, \sigma_j) = \sum_{\Omega} |G_{2D}^{(3)}(\theta_i, \sigma_j) * \mathcal{I}(\mathbf{x})|^2, \tag{3.2}$$

where $\mathcal{I}$ is an image, $\mathbf{x} = (x, y)^\top$ spatial coordinates, $*$ convolution, Ω a local aggregation region and subscript S appears on E_S to denote *spatial* orientation.

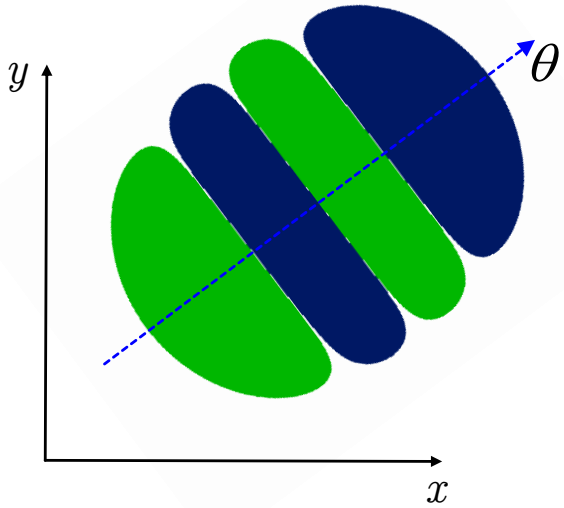


Figure 3.2: 2D Gaussian third derivative filters capture spatial orientation structure. Green colour indicates positive and blue colour negative segments. Best viewed in colour.

3.2.2 Temporal information

To analyze the oriented spatiotemporal structure of the input data, 3D Gaussian third derivative filters

$$G_{3D}^{(3)}(\theta_i, \sigma_j) = \kappa_{3D} \frac{\partial^3}{\partial \theta_i^3} \exp\left(-\frac{x^2 + y^2 + t^2}{2\sigma_j^2}\right) \tag{3.3}$$

are used. Figure 3.3 visualizes the spatiotemporal filter for a specific orientation.

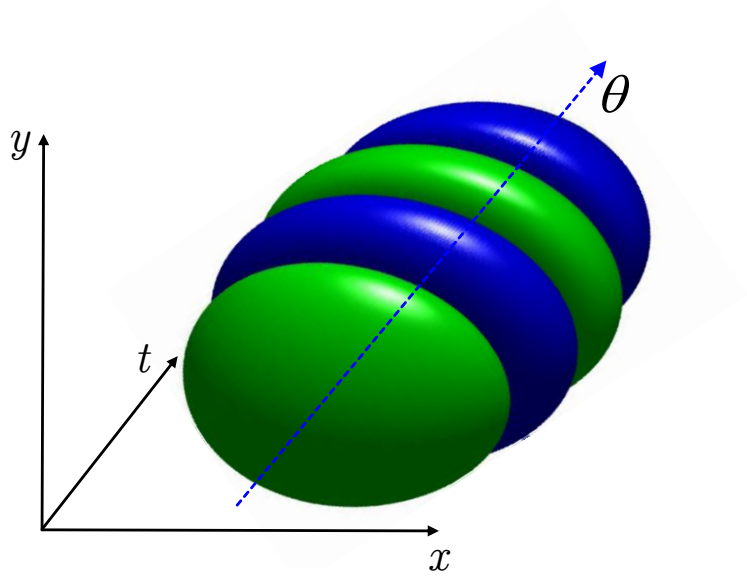


Figure 3.3: 3D Gaussian third derivative filters capture oriented spacetime structure. Green colour indicates positive and blue colour negative segments. Best viewed in colour.

Specifically, dynamic information is extracted via application of 3D Gaussian third derivative filters, $G_{3D}^{(3)}(\theta_i, \sigma_j)$ with κ_{3D} a normalization constant, and θ_i and σ_j now denoting the 3D filter orientations and scales, respectively, applied to the spacetime volume, $\mathcal{V}$, indexed by $\mathbf{x} = (x, y, t)^\top$, as generated by stacking all grayscale video frames of a sequence along the temporal axis, t, to yield

$$E_{ST}(\mathbf{x}; \theta_i, \sigma_j) = \sum_{\Omega} |G_{3D}^{(3)}(\theta_i, \sigma_j) * \mathcal{V}(\mathbf{x})|^2, \tag{3.4}$$

with subscript ST on E_{ST} to denote *spatiotemporal* orientation. At every spacetime loca-

tion $\mathbf{x}$, the local oriented energy $E_{ST}(\mathbf{x};\theta_i,\sigma_j)$ measures the power of local (σ_j) oriented structure along each considered orientation θ_i.

To uniformly sample the 3D spacetime domain, the filter orientations are chosen along the vertices of a dodecahedron. The 10 antipodal directions of the 20 dodecahedron vertices are discarded since these would induce redundant energy responses in (3.4). Therefore, the 10 employed directions are denoted as follows:

$$\theta_i \in \left\{ \begin{pmatrix} 0 \\ \phi \\ \phi^{-1} \end{pmatrix}, \begin{pmatrix} -\phi^{-1} \\ 0 \\ \phi \end{pmatrix}, \begin{pmatrix} \phi \\ \phi^{-1} \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ -\phi \\ \phi^{-1} \end{pmatrix}, \begin{pmatrix} \phi^{-1} \\ 0 \\ \phi \end{pmatrix}, \right. \\ \left. \begin{pmatrix} -\phi \\ \phi^{-1} \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \\ -1 \end{pmatrix}, \begin{pmatrix} -1 \\ 1 \\ 1 \end{pmatrix}, \begin{pmatrix} -1 \\ 1 \\ -1 \end{pmatrix}, \begin{pmatrix} -1 \\ -1 \\ -1 \end{pmatrix} \right\} \tag{3.5}$$

with $\phi = \frac{\sqrt{5}+1}{2}$ being the golden ratio. The ℓ_2 norm is applied to normalize each vector in (3.5). An illustration of these directions is given in Figure 3.4.

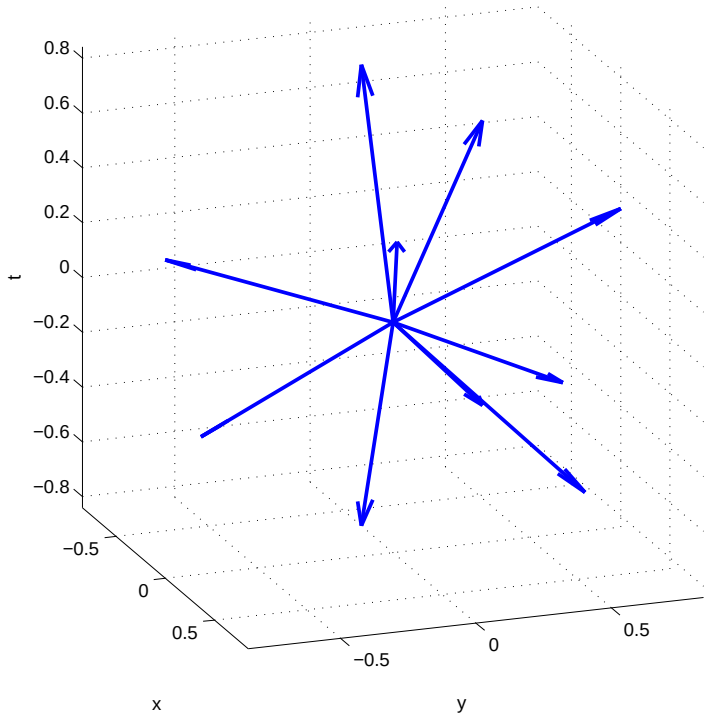


Figure 3.4: Vertices of the dodecahedron are used as the filtering directions θ_i to uniformly sample the spacetime domain. Antipodal directions are removed due to redundancy during energy filtering.

Following previous work in spacetime texture analysis [29], the spatiotemporal responses, (3.4), are further combined to yield measures of dynamic information independent of spatial appearance, as follows. In the frequency domain, motion occurs as a plane through the origin [105]. To yield measures sensitive only to dynamic orientation, irrespective of spatial orientation, the spacetime energies in equation (3.4) are combined by summing the energies across all orientations consistent with a single frequency domain plane. To span orientation space in a plane, $N+1$ basis directions for an N^{th} derivative 3D Gaussian are needed [35]. Let the plane be defined by its unit normal, $\hat{\mathbf{n}}$, then a basis set of $N+1$ equally spaced directions within the plane is calculated by

$$\hat{\theta}_i = \cos\left(\frac{\pi i}{N+1}\right)\hat{\theta}_a(\hat{\mathbf{n}}) + \sin\left(\frac{\pi i}{N+1}\right)\hat{\theta}_b(\hat{\mathbf{n}}), \tag{3.6}$$

with $0 \leq i \leq N$ and

$$\hat{\theta}_a(\hat{\mathbf{n}}) = \frac{\hat{\mathbf{n}} \times \hat{\mathbf{e}}_x}{\|\hat{\mathbf{n}} \times \hat{\mathbf{e}}_x\|_2}, \tag{3.7}$$

$$\hat{\theta}_b(\hat{\mathbf{n}}) = \frac{\hat{\mathbf{n}} \times \hat{\theta}_a(\hat{\mathbf{n}})}{\|\hat{\mathbf{n}} \times \hat{\theta}_a(\hat{\mathbf{n}})\|_2}, \tag{3.8}$$

where $N = 3$ is the order of the employed Gaussian derivative filter and $\hat{\mathbf{e}}_x$ denotes the unit vector along the x-axis in the Fourier domain.

By steering the responses consistent with a frequency domain plane $\hat{\mathbf{n}}$, it is possible to determine the energy along it

$$E_{MST}(\mathbf{x}; \hat{\mathbf{n}}, \sigma_j) = \sum_{i=0}^{N} E_{ST}(\mathbf{x}; \theta_i, \sigma_j), \tag{3.9}$$

with θ_i one of $N+1$ equally spaced orientations (3.6) consistent with the frequency domain plane and $N = 3$ is the order of the employed Gaussian derivative filters; for details see [29]. Since the summation of the energies (3.9) is located around the temporal frequency axis, the resulting measurements express a smooth approximation of the energy along the particular spacetime orientation $\hat{\mathbf{n}}$, independent of the spatial orientation. Therefore, the dynamic energies E_{MST} capture image dynamics consistent with a plane in the Fourier domain and are invariant to spatial appearance. Here, the subscript MST on E_{MST} serves to denote that the spatiotemporal measurements have been "*marginalized*"

with respect to purely spatial orientation.

3.2.3 Efficiency via separable and steerable Filters

Because convolution is a linear operation, the image only needs to be filtered with four and ten basis filters, for the spatial and temporal energies, respectively, to create responses for arbitrary orientations θ as linear combinations of these basis set volumes [35]. Another aspect for efficiency is separability. The $G^{(3)}$ filters are easily separable by expressing them as the outer product of one-dimensional vectors. Consequently, the features are very inexpensive to compute. Due to the separability, each of the basis set volumes are computed by 1D convolutions in x, y (for $G^{(3)}_{2D}$) or in x, y and t (for $G^{(3)}_{3D}$). Moreover, this can be implemented very efficiently by using a fast Fourier transform [75] algorithm.

3.2.4 Pooling of multiscale energies

Previous spacetime filtering approaches [27] to dynamic scene recognition tend to exhibit decreased performance when dealing with scenes captured with camera motion, in comparison to scenes captured with stationary cameras. A likely explanation for this result is that the approaches have difficulty in disentangling image dynamics that are due to camera motion vs. those that are intrinsic to the scenes. Here, it is interesting to note that camera motion often unfolds at coarser temporal scales (*e.g.*, extended pans and zooms) in comparison to intrinsic scene dynamics (*e.g.*, spacetime textures of water, vegetation, *etc.*); however, previous approaches have made their measurements using relatively coarse temporal scales and thereby failed to exploit this difference. In the present approach this difference in temporal scale is captured by making use of only fine scales, σ, during spatiotemporal filtering, (3.4), so that they are preferentially matched to scene, as opposed to camera, dynamics.

The orientation measurements, (3.2) and (3.9), can be taken as providing measures of the signal energy along the specified directions, θ_i. This interpretation is justified by Parseval's theorem [75], which states that the sum of the squared values over the spacetime domain is proportional to the sum of the squared magnitude of the Fourier components over the frequency domain; in the present case, the squared values of the orientation selective filtering operations are aggregated over the support regions, Ω.

3.2.5 Local contrast normalization

The linear filter responses in equation (3.4) are sensitive to image contrast. Owing to the bandpass nature of the Gaussian derivative filters, the oriented energy features are invariant to additive photometric variations (*e.g.*, as might arise from overall image brightness change in imaged scenes). To further provide for invariance to multiplicative photometric variations, each orientation selective measurement in (3.2) and (3.9) is normalized with respect to the sum of all filter responses at that point according to

$$\hat{E}_S(\mathbf{x};\theta_i,\sigma_j) = \frac{E_S(\mathbf{x};\theta_i,\sigma_j)}{\sum_{i=1}^{N} E_S(\mathbf{x};\theta_i,\sigma_j) + \epsilon} \tag{3.10}$$

for the purely spatially oriented measurements, (3.2), and similarly for the dynamic measurements, (3.9), to yield a correspondingly normalized set of measurements, $\hat{E}_{MST}$. Note that ϵ is a small constant added to the sum of the energies over all orientations. This bias operates as a noise floor and avoids numerical instabilities at low overall energies. Further, the contribution of ϵ in this ℓ_1-normalization process (3.10) is explicitly added to the set of filtering results. Calculated by

$$\hat{\epsilon}_S = \frac{\epsilon}{\sum_{i=1}^{N} E_S(\mathbf{x};\theta_i,\sigma_j) + \epsilon}, \tag{3.11}$$

to capture lack of spatial orientation structure in a region. Moreover, an analogously defined $\hat{\epsilon}_{MST}$ is added to capture lack of spatiotemporal structure. For example, notice that for regions that are devoid of oriented structure, the sum in the numerator will be dominated by ϵ so that the ratio will tend to 1 and thereby be indicative of lack of (orientation) structure.

A benefit of feature construction via convolution, (3.2) and (3.4), is the natural generation of smooth overlap between adjacent cuboids, which avoids border effects without the need for additional normalization and interpolation, as, *e.g.*, required for gradient orientation features [26].

An example for multiscale, multiorientation spatial energies, extracted from a temporal slice of an avalanche sequence, is shown in Figure 3.5. Temporal energies for the same temporal slice across several directions ($\hat{\mathbf{n}}$) are shown in Figure 3.6. Note that this scene is captured with fast camera jitter and therefore the horizontal and vertical flicker channels exhibit the strongest responses. Another example for temporal energies, extracted from a waterfall scene captured by a static camera setting is shown in Figure 3.7. One observes large filter responses for the static orientation in regions of zero image velocity and large energies for downward motion capturing the waterfall dynamics. Further, the $\hat{\epsilon}_{MST}$ channels indicates regions with lack of spatiotemporal orientation structure.

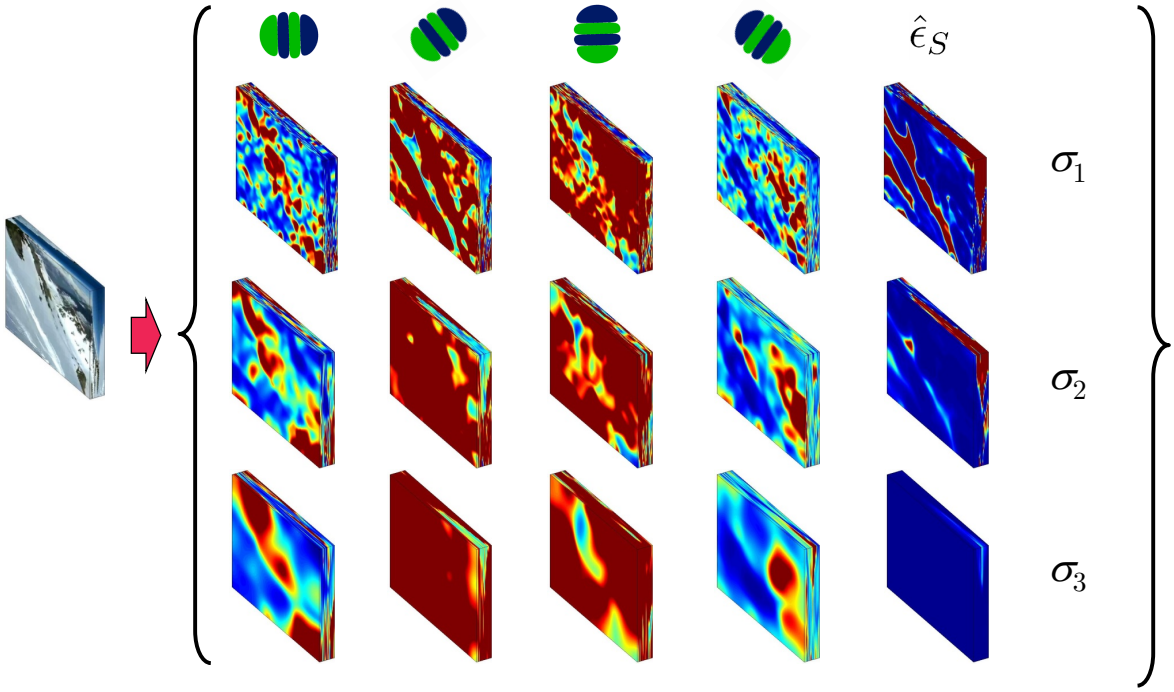


Figure 3.5: Temporal slice of an avalanche sequence filtered with oriented spatial filters at four orientations (θ) and three scales (σ), varying each scale by one octave. Furthermore the $\hat{\epsilon}_S$ channel indicates homogeneous regions in the image.

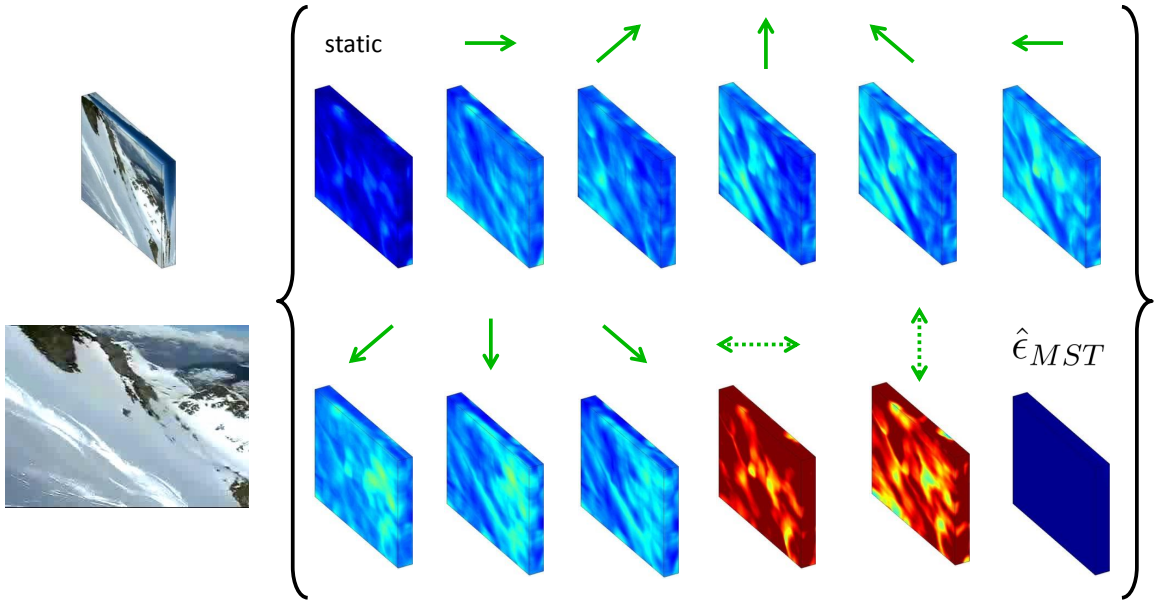


Figure 3.6: Marginalized spacetime energies for a temporal slice of an avalanche sequence filtered with spatiotemporal filters. The sequence is captured with fast camera jitter. Energies across ten motion directions are shown by steering the frequency domain plane $\hat{\mathbf{n}}$; moreover, the $\hat{\epsilon}_{MST}$ channel indicates homogeneous regions in the image sequence. The horizontal and vertical flicker channel show the most dominant responses due to the fast camera movement.

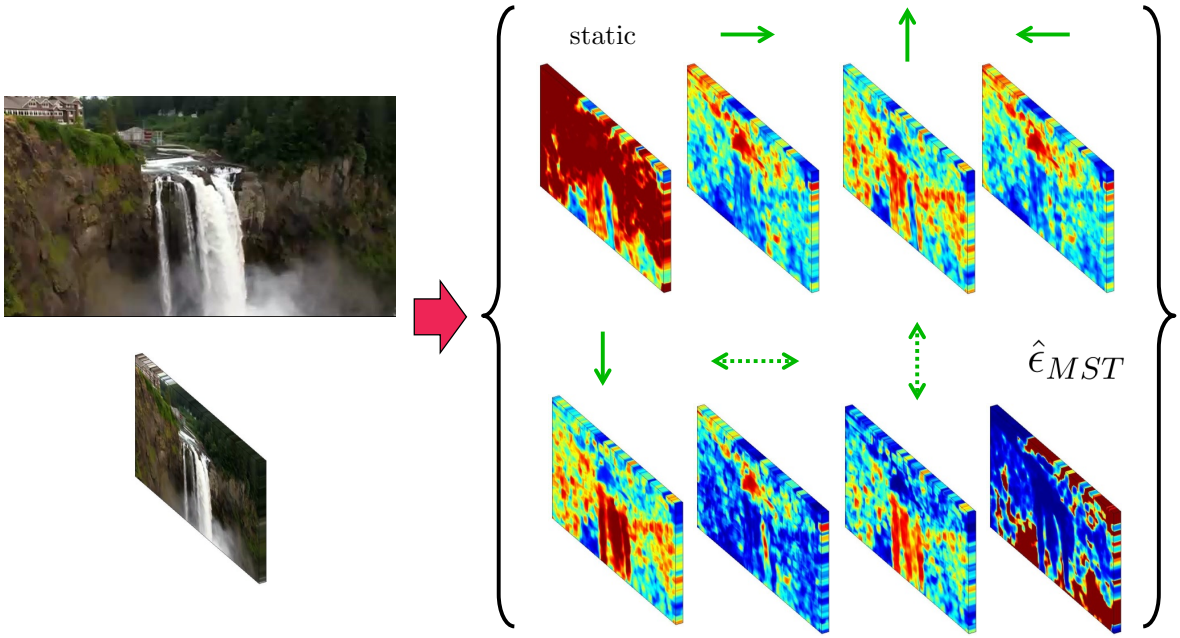


Figure 3.7: Dynamic energies for a temporal slice of a waterfall sequence obtained by convolution with spatiotemporal filters. Energies across six motion directions are shown by steering the frequency domain plane $\hat{\mathbf{n}}$; moreover, the $\hat{\epsilon}_{MST}$ channel indicates homogeneous regions in the image sequence. Strong energies in the downward motion channel are observed across the waterfall.

3.2.6 Chromatic information

Chromatic information can greatly influence (static) object and scene recognition [97] and also has proven useful in previous work on dynamic scene recognition [27, 90]. Correspondingly, chromatic information is incorporated in the present dynamic scene descriptor by adding three more measurements at each point in the image sequence taken as CIE-LUV colour space observations [110]. Other colour spaces also were considered (RGB, HSV, CIE-Lab [110]); however, LUV led to slightly better results in preliminary investigation.

3.2.7 Temporal slice-based aggregation

The complementary spacetime orientation measurements presented so far are defined pointwise across a video sequence. For the purpose of on-line classification of the entire video into a scene class, the local descriptors are summed across time, t, within τ discrete temporal slices of equal duration. This operation yields a set of temporally aggregated images, which are referred to as temporal slices. Fig. 3.8 illustrates the temporal parcelling of an avalanche sequence, taken from the Maryland dataset. Temporal slicing is motivated by the desire for incremental processing that can allow for efficient, on-line operation. Use of short-term parceling of the measurements also is well matched with the restriction to use of fine temporal scales during spatiotemporal filtering to favour scene over camera dynamics. During classification (Sec. 3.3) each temporal slice initially is classified individually, with the individual classifications subsequently combined to yield overall classification for the video.

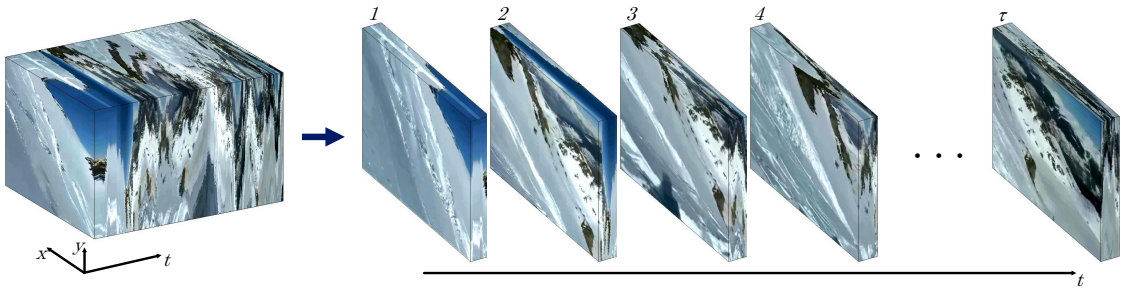


Figure 3.8: Temporal slice-based processing for on-line recognition.

Having established temporal slices for an input video, the complementary measure-

ments are processed in successive temporal slices across the video. Each slice is hierarchically aggregated into histograms to form a spatiotemporal pyramid, analogous to that used previously for static [58] and dynamic [27] scene analysis. At each level of the pyramid, each temporal slice is broken into $X \times Y \times T$ 3D cuboids (see Fig. 3.1(a)), with filter measurements collapsed into histograms within each cuboid, as illustrated in Fig. 3.1(d). The support of the cuboid at any given level of the pyramid corresponds to its outer scale [51]; indeed, it corresponds to the aggregation region Ω in the filtering equations, (3.2) and (3.4). Moreover, the adjacency structure of the cuboids capture the overall scene layout. For each cuboid, the histograms are l^1-normalized, to represent a distribution of chromatic, multiscale oriented spacetime energy and lack of oriented structure (via $\hat{\epsilon}$). Let M_θ, $M_{\hat{\mathbf{n}}}$, M_{σ_θ} and $M_{\sigma_{\hat{\mathbf{n}}}}$ be the number of spatial orientations, spatiotemporal orientations and their (inner) scales considered in the multiscale oriented filtering operations, resp. Then, the dimension of each histogram is the quantity $(M_\theta{+}1){\times}M_{\sigma_\theta}{+}(M_{\hat{\mathbf{n}}}{+}1){\times}M_{\sigma_{\hat{\mathbf{n}}}}{+}3$, with 1 added to the number of orientations due to $\hat{\epsilon}$, and 3 the number of colour channels. The histograms for all cuboids are concatenated into a final feature vector, $\mathbf{v}$, that comprises the Complementary Spacetime Orientation descriptor (CSO) to characterize a temporal slice of the video. Figure 3.9 shows a schematic of the feature vector extraction for one specific spacetime energy channel of a temporal slice at an outer spacetime scale of $X \times Y \times T = 2 \times 2 \times 1$.

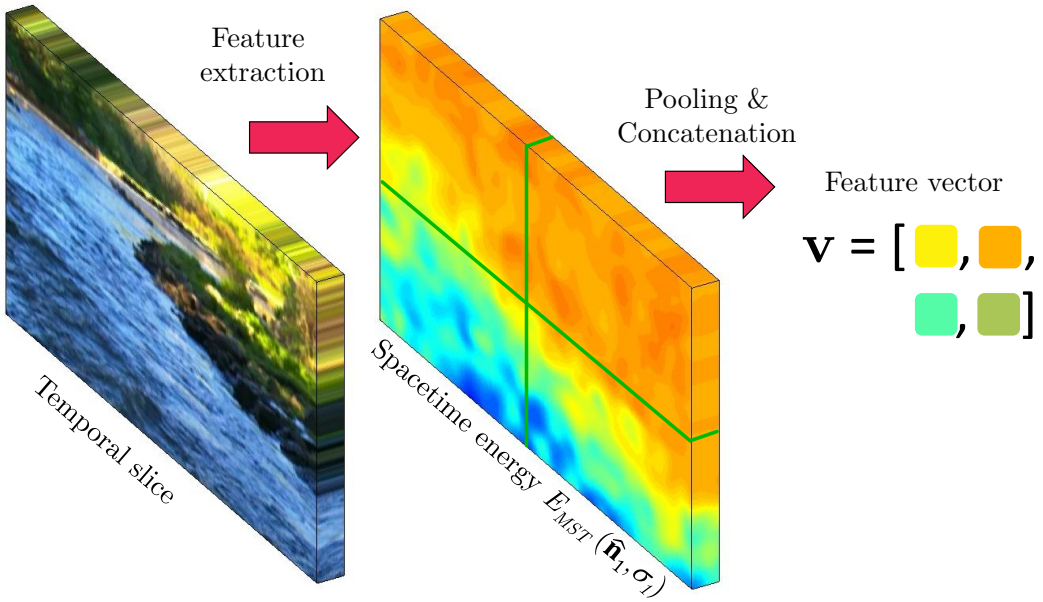


Figure 3.9: Spacetime volume description by one spacetime energy channel (static/dynamic energy) and pooling of the features in a spatiotemporal pyramid structure. Only a single outer scale ($2 \times 2 \times 1$) is illustrated.

3.3 Spacetime forests

In the present work, a Random Forest (RF) classifier is employed for its ability to combine several cues for multi-class prediction, as well as their increased speed in the training and testing processes over traditional classifiers. Here, the classes correspond to different dynamic scenes (*e.g.*, beach vs. city, *etc.*) and the feature vectors correspond to the CSO descriptors defined in the previous section. In this section, a particular instantiation of RFs, termed spacetime Random Forests (STRF) are defined.

3.3.1 Multi class random forests for recognition

RF classifiers have been introduced for character recognition in [2] and were extended by [12]. Further applications include fast keypoint tracking and feature matching [59], object recognition [71, 108], image classification [8] and segmentation [89]. Detailed descriptions of Random Forests may be found in the literature [2, 12, 23, 24].

RFs are an ensemble of F decision trees $\{\mathcal{T}_k\}_{k=1}^{F}$ learned by random feature selection. Each decision tree is used independently to classify the input feature vector, $\mathbf{v}$, based on the leaf node at which the corresponding feature vector arrives. Hence, the leaf nodes of all trees hold the posterior distribution $P(c|\mathbf{v})$ over the classes $c \in \{1, \ldots, C\}$.

3.3.2 Learning dynamic scenes

Note that previous work [27, 90] collapsed temporal information during training and testing and therefore discarded possible important temporal cues. Further, collapsing across the entire temporal extent of a video may have limited the ability of previous approaches to disentangle scene dynamics from camera motion.

During training, the temporal alignment of the video slices are treated as latent; correspondingly, each temporal slice of each training video generates its own feature vector according to procedures of Sec. 3.2. This approach allows leveraging of the high temporal diversity in the spatiotemporal patterns.

Each tree is constructed by drawing a random bootstrap sample from the training set. Bootstrapping in the training stage allows maximum training efficiency [24] and avoids over-fitting [2] to generalize well in practice. Further randomness is introduced in the

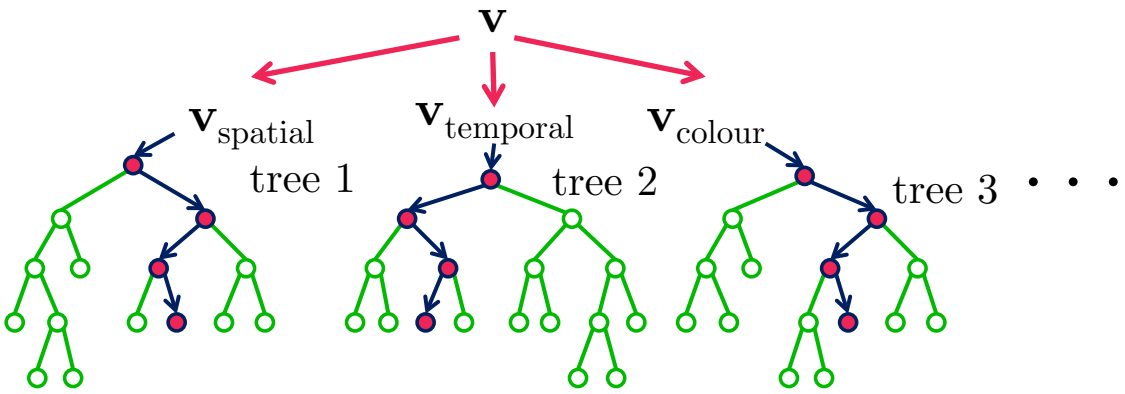


Figure 3.10: Spacetime forest construction. As some classes may be better represented by specific feature types, the node optimization process in each tree of the spacetime random forest is restricted to a single feature type.

node optimization step by selecting a random subset m of the feature vector's dimension to be considered in a random threshold test for determination of the best split for each node. Here, the split is selected as that which maximizes the information I in the final class distribution, after splitting into a left (L) and right (R) node:

$$I = H(Q) - \sum_{i \in \{L,R\}} \frac{|Q^i|}{|Q|} H(Q^i), \tag{3.12}$$

where $H(Q) = -\sum_{c \in C} p(c) log(p(c))$ is the Shannon entropy, $p(c)$ the proportion of classes in Q belonging to class c and $|\cdot|$ denotes the size of the set at a given node. Tests are selected for all nodes recursively until the growing process is stopped when no further information gain is achieved. The final class distributions at the leaves are calculated as the ratio between the number of feature vectors of each class and the total number of features which reach the leaf after training.

As some classes may be better represented by specific feature types, the node optimization process in each tree is restricted to a single feature type. To best separate the classes with the CSO descriptor, the input for the RF is first structured into the three complementary feature channels: spatial orientation, (marginalized) spatiotemporal orientation and colour. Then, the channels are used to train $\frac{F}{3}$ trees each, to best distinguish the classes, based on a particular channel only. An illustration is shown in Figure 3.10.

Lastly, these complementary trees are merged to obtain the spacetime forest $\{\mathcal{T}_k\}_{k=1}^{F}$.

3.3.3 Recognizing dynamic scenes

For classification, the feature vectors, $\mathbf{v}^\tau$, of scenes to be recognized are again decomposed into the three distinct channels and sent simultaneously through the respective complementary trees until the corresponding leaves are reached. Here, τ is the temporal slice of the input volume where the feature is extracted. Each tree gives a classification by voting for the class labels according to the class distribution $p_k(c|\mathbf{v}^\tau)$ of the leaf which is reached by $\mathbf{v}^\tau$ in tree k.

Given the resulting set of tree predictions, they are combined in two stages to yield a classification at each temporal instance. First, the prediction results of the forest for the current temporal slice, τ, are calculated as a simple averaging of the leaf distributions p_k in the F trees in the forest

$$P^\tau(c|\mathbf{v}^\tau) = \frac{1}{F}\sum_{k=1}^{F} p_k(c|\mathbf{v}^\tau). \tag{3.13}$$

Using this prediction, a class label can be assigned to each temporal slice via

$$c^\tau = \arg\max_c P^\tau(c|\mathbf{v}^\tau). \tag{3.14}$$

Second, to yield a final classification across all temporal slices available up to a given time, the class likelihoods for each slice are treated as temporal predictions and once again combined via averaging

$$P(c|\mathbf{v}) = \frac{1}{\tau}\sum_{l=1}^{\tau} P^l(c|\mathbf{v}^l). \tag{3.15}$$

The current classification of the video is then given as

$$c = \arg\max_c P(c|\mathbf{v}). \tag{3.16}$$

3.4 Implementation details

3.4.1 CSO video descriptor

In the current implementation, $M_\theta = 4$ and 10 in the oriented filtering operations, (3.2) and (3.4), resp., as those numbers span orientation space for the order and dimensionality of filters employed [35]. Here, it is of note that orientation selective filters other than Gaussian derivatives might have been used (*e.g.*, oriented Gabor filters[43]); however, the chosen filters enjoy the advantage of particularly efficient implementation owing to separability and steerability [35]. In any case, the results of the spatiotemporal filtering, (3.4), are further combined to capture frequency domain planes, (3.9), parameterized by $\hat{\mathbf{n}}$ corresponding to motion along the leftward, rightward, upward and downward directions as well as static (zero velocity) and flicker (infinite velocity). For each orientation, spatial filtering is performed at $M_{\sigma_\theta} = 4$ different scales, starting at $\sigma = 2$, varying coarser by octave; spatiotemporal filtering is performed at $M_{\sigma_{\hat{\mathbf{n}}}} = 1$ relatively fine scale ($\sigma = 2$) in preference for capturing short term temporal variations. To avoid border effects at the start and the end of the volumes, the filtering is performed with a temporal offset of half of the largest filter size used. During normalization, (3.10), $\epsilon = 500$. The spacetime pyramid is constructed at 4 levels with number of cuboids $(X \times Y \times T) \in \{(8 \times 8 \times 1), (4 \times 4 \times 1), (2 \times 2 \times 1), (1 \times 1 \times 1)\}$. Pyramids are constructed for each temporal slice of an input video, with the length of temporal slices set to 16 frames. To represent the colour distribution in each cuboid, a 3 bin histogram of the CIE-LUV colour channels is employed.

3.4.2 STRF classifier

Even if the training data exhibits the same number of videos for each class, they may have different durations. Therefore, the classifier would be severely biased towards the classes containing long videos. To compensate for these differences, priors are used in the training stage subsampling process. These are given by the inverse of the number of temporal slices τ of all videos from a specific class c in the training set. For all experiments a multi-class STRF with 500 trees for each of the three feature channels (*i.e.*, spatial orientation, (marginalized) spatiotemporal orientation and colour) of the video descriptor is trained. At each split node, a random sample of m features is drawn for consideration

in splitting [12]. In particular, $m = \lfloor \log_2 D \rfloor$, where D is the feature vector dimensionality. The best split of the random tests, determined by Eq. (3.12), is used to split the node. For training the node splits, each tree uses a random bootstrap sample consisting of two thirds of the training data. The error rate for observations left out of the training data is monitored as out-of-bag error rate and may be used as an unbiased estimate of the classification performance of each tree trained on a bootstrap sample [12].

3.4.3 Invariance to scale variations

To provide a degree of robustness to scale variations of image capture (*e.g.*, due to variable zoom), the multiscale filtering (parametrized by σ in (3.2) and (3.4)) may be performed multiple times for each training video, but with the particular set of scales considered shifted by a different amount each time. This approach allows for a range of imaged scene scales, σ_j, to be captured at any given image capture resolution. During training, a feature vector is constructed for each set of shifted scales and used separately as training data for the forest. While this approach is somewhat redundant in processing of scale, it will allow for a scene class to be recognized at any of the given scale shifts while only sending a single set of scales into the forest. Notice that since the forests have been built with various scale shifts during training, a feature vector derived from an imaged scene at any of those shifts (*i.e.*, variable resolutions) will be captured properly.

Since the underlying datasets of the experimental validation all provide a similar scale (*i.e.* all videos are captured at a similar zoom), the scale invariant implementation is not used in this work. Essentially, by running the experiments with the scale invariant implementation, identical performance was achieved in preliminary investigations on these datasets.

3.5 Experimental evaluation

This section evaluates the proposed approach for dynamic scene recognition on the Maryland "In-The-Wild" [90] and YUPENN Dynamic Scenes [27] datasets, consisting of 13 and 14 classes with 10 and 30 videos per class, respectively. As shown in Section 2.2, the datasets contain videos showing a wide range of natural dynamic scenes (avalanches, traffic, forest fires, waterfalls *etc.*); see Tables 3.1 and 3.2 where complete listings can be read off the left most columns of the tables. A notable difference between the two datasets is that the Maryland dataset includes camera motion, while the YUPENN dataset does not.

3.5.1 Evaluation methodology

To be consistent with previous evaluations in [27, 90], we use the same evaluation protocol as the authors of the datasets; *i.e.* a leave-one-video-out recognition experiment. For a systematic evaluation of the contributions of (i) the video descriptor (CSO), (ii) classifier (STRF), and (iii) the use of temporal slicing and priors for classification, these components are evaluated separately in the remainder of this section.

For the sake of comparison, several alternative video descriptors and classifiers are considered that have shown strong performance in previous evaluations [27, 90]. Descriptors considered are GIST [73] + HOF [67], GIST + chaotic dynamic features (Chaos) [90] and Spatiotemporal Oriented Energies (SOE) [27]. All of these approaches include colour histograms [40], as this addition generally increases classification performance [27, 90]. Furthermore, the performance of the recently introduced approach by Theriault *et al.* [95], that based on Slow Feature Analysis (SFA), is shown. The classifiers considered are Nearest-Neighbor (NN), Support Vector Machine (SVM), Random Forest (RF) and the proposed Spacetime Random Forest (STRF). NN and SVM are included, as they have been employed in previous evaluations [27, 90]; RF is a random forest trained uniformly across all components of the training vectors, which is included for the sake of comparison to the proposed STRF, which trains separate trees for the spatial and temporal orientation as well as colour components. The alternative approaches build their feature vectors by collapsing across *all* temporal frames; for the sake of comparison, the proposed approach is shown processing temporal information in several ways: Use a single temporal slice for classification (RF and STRF, $\tau = 1$); average the CSO feature vectors

calculated for individual slices across the entire video (RF, τ =*all*); combine individual slice classifications across the entire video (STRF, $\tau = all$) according to the proposed averaging (3.15), *i.e.* the complete proposed approach. Results are shown in Tables 3.1 and 3.2 for the Maryland and YUPENN datasets, respectively.

3.5.2 Results

Maryland "In-The-Wild"										
Descriptor	HOF+ GIST	Chaos+ GIST		SOE		SFA	CSO (proposed)			
Classifier	NN	NN	SVM	NN	RF	SVM	RF	RF	STRF	STRF
Temporal τ	*all*	*all*	*all*	*all*	*all*	*all*	1	*all*	1	*all*
Avalanche	20	40	60	10	40	60	40	50	60	60
Bo. Water	50	40	60	50	50	70	80	80	80	80
Ch. Traffic	30	70	70	80	60	80	90	100	80	90
Forest Fire	50	40	60	40	10	10	30	50	80	80
Fountain	20	70	60	10	50	50	40	50	90	80
Iceberg Co.	20	50	50	10	40	60	50	40	60	60
Landslide	20	50	30	50	20	60	20	40	20	30
Sm. Traffic	30	50	50	70	30	50	60	50	60	50
Tornado	40	90	80	60	70	70	70	60	90	80
Volcanic Er.	20	50	70	30	10	80	50	80	50	70
Waterfall	20	10	40	20	60	50	50	50	50	50
Waves	80	90	80	80	50	60	70	70	60	80
Whirlpool	30	40	50	40	70	80	80	50	80	70
Overall	**33**	**52**	**58**	**42**	**43**	**60**	**57**	**59**	**66**	**68**

Table 3.1: Average classification rates for different video descriptor and classifier combinations on the Maryland dataset. The combination of diverse, informative feature channels (CSO) with a suitable classifier (STRF) gives overall best results.

In comparison to the most closely related descriptor (SOE[1]) running under the same classifier (RF), it is seen that the proposed CSO features improve overall performance in the presence of camera motion (Maryland dataset) from 43% to 57% recognition accuracy using only a single temporal slice ($\tau = 1$), with an additional boost to 59% when feature vectors are combined across all slices (τ =*all*). In contrast, when camera motion is not present (YUPENN), the performance of the two feature sets under the same classifier is essentially indistinguishable (81% vs. 82%). These results support the ability

[1]Recall that SOE derives from integrated spatiotemporal filtering, (3.4), without temporal slicing, and without the proposed separation into complementary spatial, (3.2), and temporal, (3.9), components.

YUPENN Dynamic Scenes data set									
Descriptor	HOF+ GIST	Chaos+ GIST	SOE		SFA	CSO (proposed)			
Classifier	NN	NN	NN	RF	SVM	RF	RF	STRF	STRF
Temporal τ	*all*	*all*	*all*	*all*	*all*	1	*all*	1	*all*
Beach	87	30	90	93	93	97	100	100	100
Elevator	87	47	90	100	97	97	97	97	100
Forest Fire	63	17	87	67	70	80	83	76	83
Fountain	43	3	50	43	57	47	53	40	47
Highway	47	23	73	70	93	67	70	67	73
Lightning S.	63	37	90	77	87	90	90	93	93
Ocean	97	43	97	100	100	90	90	90	90
Railway	83	7	90	80	93	87	90	90	93
Rushing R.	77	10	90	93	87	93	93	97	97
Sky-Clouds	87	47	93	83	93	87	90	100	100
Snowing	47	10	50	87	70	47	43	57	57
Street	77	17	87	90	97	93	90	97	97
Waterfall	47	10	47	63	73	67	70	80	76
Windmill F.	53	17	73	83	87	93	87	93	93
Overall	**68**	**23**	**79**	**81**	**85**	**81**	**82**	**84**	**86**

Table 3.2: Average classification rates for different video descriptor and classifier combinations on the YUPENN dataset. The complementary feature channels of the CSO descriptor, combined with the STRF classifier, achieves best results.

of the proposed approach to capture intrinsic scene dynamics with robustness to camera motion. Further allowing the classifier to consider the complementary feature components separately (STRF) shows even better performance whether with $\tau = 1$ slice or with combination across $\tau =$*all* slices.

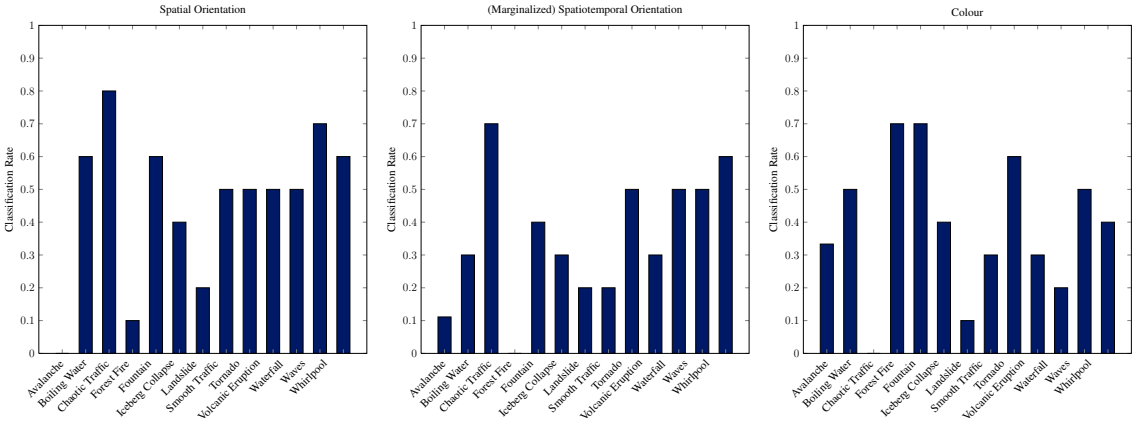
More generally, the proposed approach's 68% accuracy on the Maryland dataset improves on Chaos+GIST under SVM by 10%. Further, its accuracy of 86% on the YUPENN dataset sets a new state-of-the-art for that case as well (even with SOE given the advantage of RF-based classification, which it did not enjoy in its original application to dynamic scenes [27], but which is included here for fair comparison). In comparison to the recently published approach based on slow feature analysis (SFA) [2] by Theriault *et al.* [95], the proposed method outperforms SFA by 8% on Maryland and 1% on YUPENN. Indeed, while all other compared approaches show high variation in performance between the two datasets, the proposed approach provides the best overall performance

[2]Consider the erratum with the correct classification rates of SFA on `http://webia.lip6.fr/~theriaultc/sfa.html`

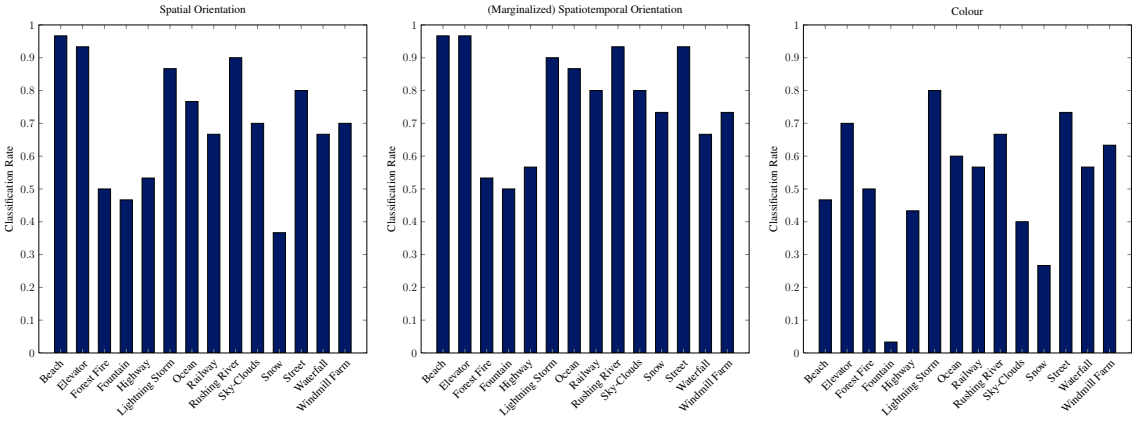
in both cases. Moreover, best overall performance is attained even when only a single temporal slice of 16 frames is processed (CSO, STRF, $\tau = 1$).

The complementary nature of the CSO descriptor components is illustrated explicitly in Fig. 3.11(a) and 3.11(b). The figure shows estimates of cross-validation performance as indicated by the out-of-bag error rate [12] when training the random forest with spatial and temporal orientation as well as colour information separately. It is seen that different classes are better distinguished by different types of information; correspondingly, their combination provided by the proposed approach (Table 3.1 and Table 3.2) yields improved classification performance.

Examination of class confusions made by the proposed approach is presented in Table 3.3. The confusion matrix exhibits dominant diagonal entries. One observes that most of the confusions are between intuitively similar scenes, *e.g.*, Smooth Traffic classified as Chaotic Traffic, or Waterfalls classified as Fountains. Note, however, that some confusions on the Maryland dataset are rather diffuse, *e.g.*, the Landslide class. This may be explained by the large variations within these classes.



(a) Maryland "In-The-Wild"



(b) YUPENN Dynamic Scenes data set

Figure 3.11: Classification performance measured by the out-of-bag error rate when training the random forest separately with spatial and spatiotemporal orientation as well as colour components of the CSO descriptor. A single temporal slice of each clip is used for training.

Maryland "In-The-Wild"

Actual \ Classified	*Avalanche*	*Bo. Water*	*Ch. Traffic*	*Forest Fire*	*Fountain*	*Iceberg Co.*	*Landslide*	*Sm. Traffic*	*Tornado*	*Volcanic Er.*	*Waterfall*	*Waves*	*Whirlpool*
Avalanche	**5**					1	1			1		1	1
Bo. Water		**8**					1			1			
Ch. Traffic			**8**					1					1
Forest Fire				**7**					1		2		
Fountain	1				**8**	1							
Iceberg Co.	1		1		1	**6**						1	
Landslide	1		2	2			**3**					1	
Sm. Traffic			3		1			**5**			1		
Tornado				1					**8**	1			
Volcanic Er.	1					1			1	**7**			
Waterfall	1			1	2		1	1			**4**		
Waves	1								1			**7**	1
Whirlpool			2			1							**7**

YUPENN Dynamic Scenes data set

Actual \ Classified	*Beach*	*Elevator*	*Forest Fire*	*Fountain*	*Highway*	*Lightning S.*	*Ocean*	*Railway*	*Rushing R.*	*Sky-Clouds*	*Snowing*	*Street*	*Waterfall*	*Windmill F.*
Beach	**30**													
Elevator		**30**												
Forest Fire			**25**		1						3	1		
Fountain				**14**				1	2	1	1	1	7	3
Highway					**22**	1		2	2			1	2	
Lightning S.			1			**28**				1				
Ocean	3						**27**							
Railway								**27**	2	1				
Rushing R.	1								**29**					
Sky-Clouds										**30**				
Snowing		2		1	4			2		1	**17**	1	2	
Street					1							**29**		
Waterfall				2	1				3	1			**23**	
Windmill F.													1	**29**

Table 3.3: Confusion matrices for both datasets, using CSO descriptor and STRF classification.

3.5.3 Exploration of the spacetime pyramid parameter space

As each tree in the forest is constructed on only two-thirds of the training set, the data left out of the training set may be used to provide an unbiased estimate of cross-validation performance [12]. Here, the classification performance on the out-of-bag data for training a random forest with different spacetime grid sizes is evaluated. The spacetime pyramid is constructed at different levels with number of cuboids in the range of $(X \times Y \times T) \in \{(4 \times 4 \times 1), \ldots, (16 \times 16 \times 16)\}$. Figure 3.12 shows the effect of this different outer-scale grid sizes on the YUPENN dataset. Note that in this particular case the whole clip is used for training (no slicing), as temporal slices are generated implicitly by applying a temporal grid. One observes that finer grids, which lead to a higher feature dimensionality, increase overall classification accuracy up to a certain point where the representation over-fits the underlying volumes. In all cases, a good trade-off between dense spatial description and good generalization can be achieved with a grid size of $(X \times Y \times T) = (8 \times 8 \times 1)$. Note, however, that the YUPENN dataset does not include much temporal variation, since all the classes, besides Elevator, consist of constant spacetime textures over time. Therefore, Figure 3.12 also indicates, that collapsing the temporal information is as effective as separate temporal grids. This is a consequence of the continuous spatiotemporal patterns, captured by a static camera, in the YUPENN dataset.

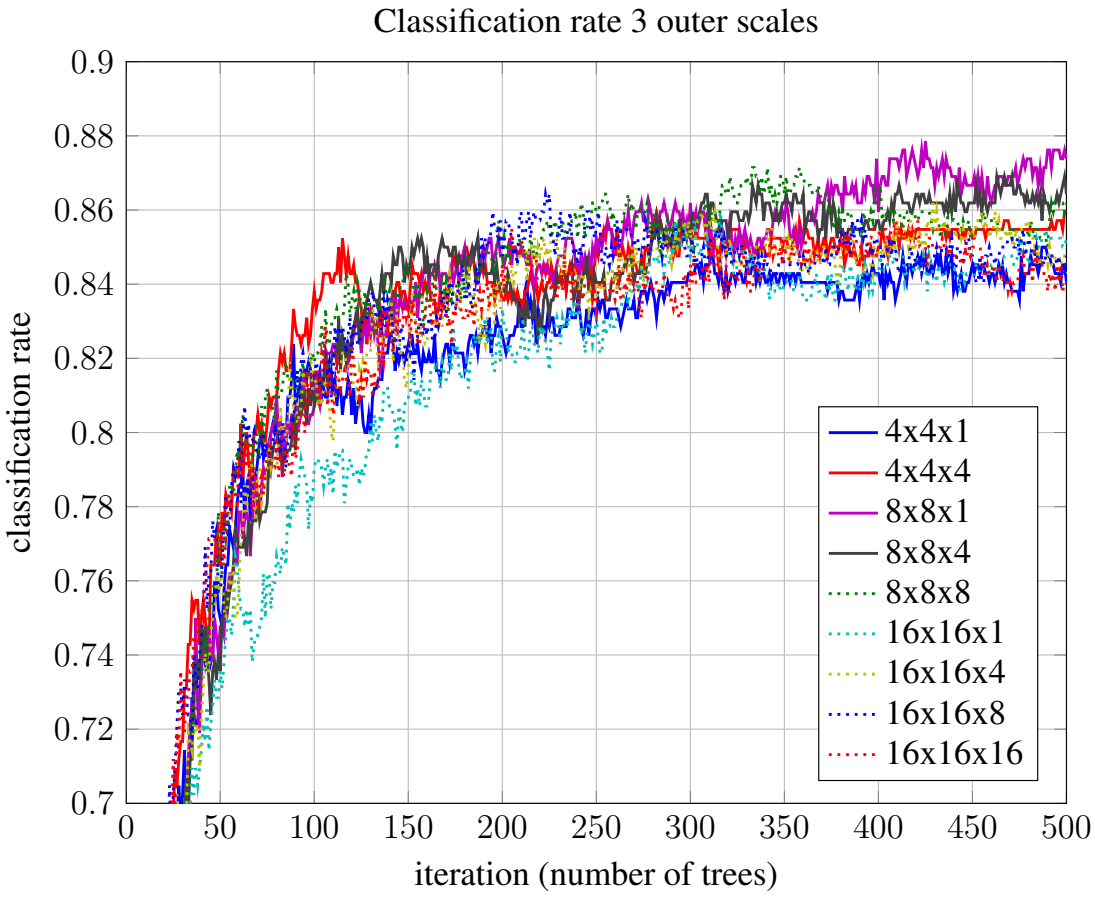


Figure 3.12: Performance of CSO on YUPENN, with 3 outer scales used, the grid size of the finest outer scale is listed. Sparse grids, such as $4 \times 4 \times 1$ and too dense grids, *e.g.*, $16 \times 16 \times 16$ lower the classification accuracy of the forest.

Next, the impact of discarding the coarse grids of the pyramid representation is investigated (*e.g.* only using a $8 \times 8 \times 1$ grid for representation and discarding the coarse grids $4 \times 4 \times 1$ and $2 \times 2 \times 1$). Figure 3.13 shows different spacetime grids, when only a single outer scale is used; *i.e.* only the finest grid. The average decrease in performance, compared to using the full spatial pyramid (Figure 3.12) is minor. This suggests that the most discriminative information is captured by the finest spatial pyramid levels.

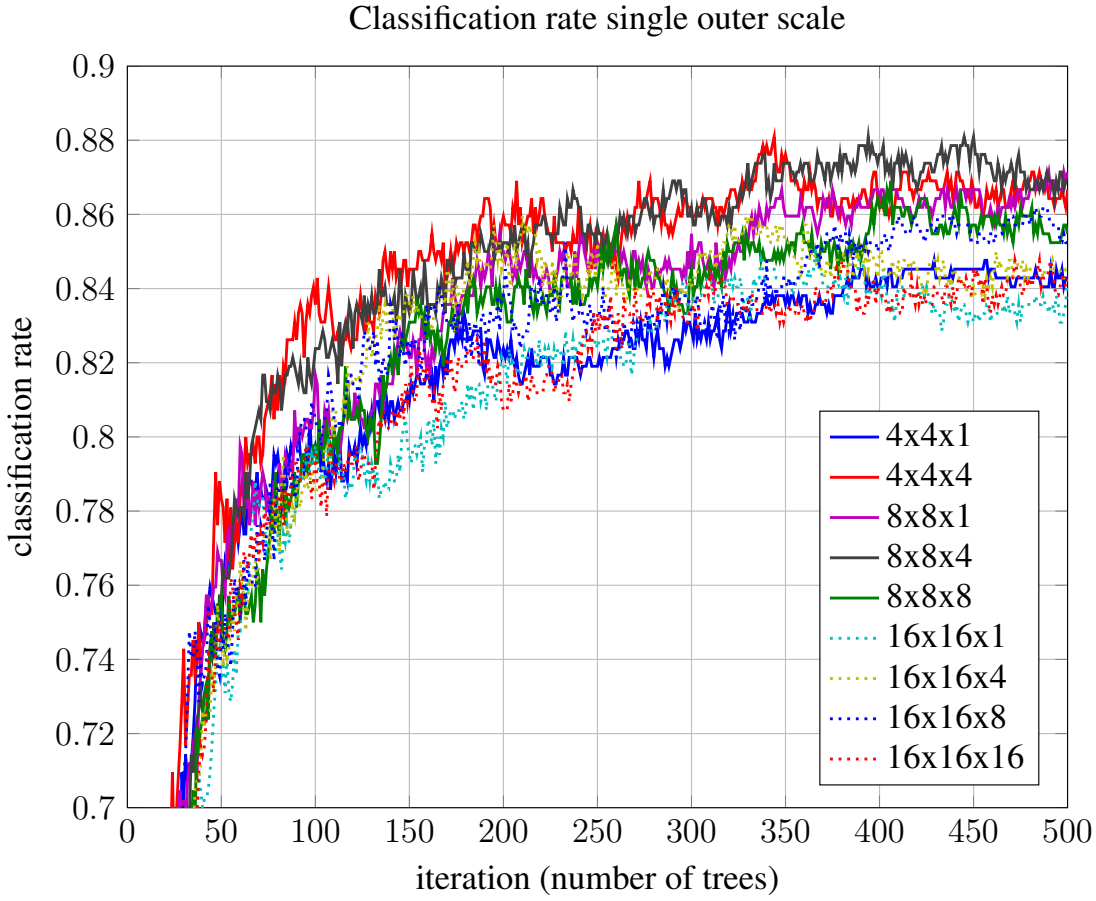


Figure 3.13: Performance of CSO on YUPENN, when pooled from the finest outer scale only, *i.e.*, the coarse grids of the spatial pyramid are not used.

3.5.4 Computational time

The current implementation is written in Matlab and may be further optimized. In terms of execution speed, the full proposed approach computes a feature vector for a 16 frame slice in 4 seconds (due to separable and steerable filtering) and takes an additional 5 milliseconds on average to report a class label. As shown in Section 3.5.2, even using a single temporal slice of the input sequence yields high-quality results. Therefore, the approach allows state-of-the-art scene classification, being within 2% accuracy of the best performance across both datasets (also attained by the proposed approach, but using a complete set of slices), in nearly real time. Moreover, filtering and random forests are readily parallelizable for GPU implementations.

3.6 Conclusion

This chapter has presented a novel approach to dynamic scene recognition based on three key ideas. First, different scenes are best characterized in terms of different combinations of spatial, temporal and chromatic information. Correspondingly, the CSO descriptor has been introduced that affords complementary consideration of spatial and spatiotemporal orientation as well as colour information. Second, a particular instantiation of random forests, STRF, has been introduced that allows the complementary components of the CSO descriptor to be exploited during classification. Third, temporal slicing with scale matched to intrinsic scene dynamics has been employed. Matching the scale of spatiotemporal filtering to scene dynamics allows for recognition that is robust to camera motion. Slicing also allows for efficient, incremental processing of video as well as treatment of temporal alignment as latent during classification. In empirical evaluation relative to a variety of previous algorithms for dynamic scene recognition, including the previously most effective ones, the proposed approach has yielded superior accuracy in dynamic scene recognition, both with and without camera motion.

A limitation of the proposed approach is that spacetime features are aggregated over relatively large spatial regions and therefore only capture a smooth approximation of the spatiotemporal energy in a region. The next chapter will overcome this problem by representing local spacetime features, encoded as visual words, that are used for a deeper representation of the image sequence. However, despite the better representation, local feature encoding enforces a higher feature dimensionality and is computationally expensive. Therefore, due to the inherent computational efficiency of the approach presented in this chapter, it is well suited for applications with relevance to real-world scenarios, including video indexing and browsing as well as surveillance and monitoring. Integration of CSO and STRF with such applications would serve as an interesting direction for future research.

4

Bags of Spacetime Energies for Dynamic Scene Recognition

This chapter extends the idea of temporal slicing introduced in Chapter 3. In contrast to the approach presented in Chapter 3, the extracted features here capture unified spatial and temporal orientation structure of a local region.

The proposed approach models dynamic scenes with Bags of Spacetime Energies (BoSE). The well-established BoW image classification architecture is applied. Local spacetime features are extracted from a training set to build a visual codebook that is used to project these feature distributions into a mid-level representation. Finally, the local feature codes are pooled over a spatiotemporal pyramid to form the global representation for classification.

In the last decade, the state of the art in image classification and object recognition is dominated by three general steps: (i) In the *feature extraction* step, low-level descriptors are extracted either from interest points or densely from regular locations. (ii) The *coding* step generates intermediate bag of visual word (BoW) representations that transform local features into more effective representations for the underlying task, and (iii) the *pooling* step accumulates encoded features over pre-defined regions. Spatial pyramid matching (SPM) [58] is typically employed to embed weak geometric information by using increasingly finer subregions for pooling, while still providing important properties of spatial invariance (see Figure 4.1 for an illustration).

A recent BoW approach [95] to dynamic scene recognition uses purely spatial descriptors and their variation over time. These features are transformed in several stages, resulting in a relatively complex system with numerous components. Therefore, it is not very clear which modules are essential for good performance. For example, the authors of [95] claim that the use of slowly varying features is essential for the performance of their approach; however, they only report results for the full SFA system, without discussing the benefit of mid-level feature coding for dynamic scene classification.

The present work builds on an evaluation [27] of different feature types for dynamic scene recognition, where it has been shown that features, jointly capturing spatial and temporal structure, are performing best for the task. Also the results of CSO+STRF in Chapter 3 have demonstrated that using spatial and temporal features is important for good recognition performance. However, in contrast to these approaches, the method presented in this chapter only focuses on local spatiotemporal features (specifically, CSO+STRF is based on local temporal, but non-local spatial features). After evaluating several popular feature coding strategies for the proposed local spacetime features, a carefully-designed BoW model, *i.e.* the Bags of Spacetime Energies (BoSE), for dynamic scene recognition is introduced. An overview of the framework is shown in Figure 4.1. One striking result of this work is that a well-designed representation, combined with common feature encoding concepts, outperforms all previous dynamic scene categorization approaches from the literature as well as CSO+STRF by a significant margin.

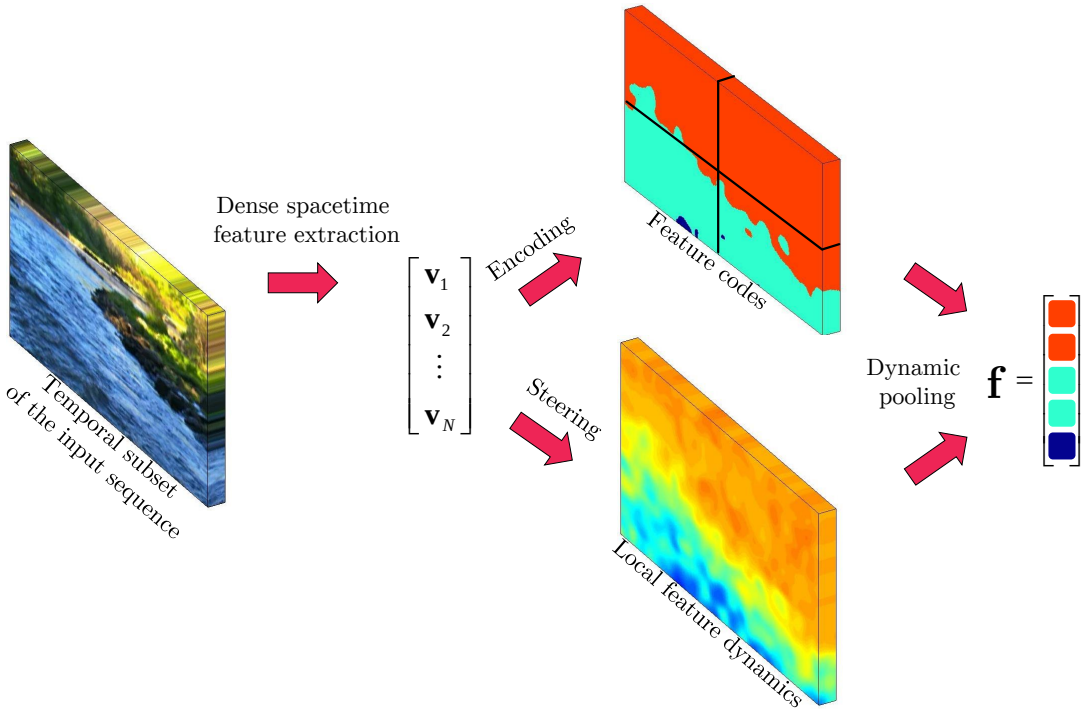


Figure 4.1: Proposed Representation for Dynamic Scene Recognition. First, spatiotemporal oriented primitive features are extracted from a temporal subset of the input video. Second, features are encoded into a mid-level representation learned for the task and also steered to extract dynamic pooling energies. Third, the encoded features are pooled via a novel dynamic spacetime pyramid that adapts to the temporal image structure, as guided by the pooling energies. The pooled encodings are concatenated into vectors that serve as the final representation for online recognition.

4.1 Contributions

This chapter makes several substantial contributions to dynamic scene classification. First, a novel feature representation based on a weighted aggregation of local fine-scale spacetime energies is proposed. Second, an evaluation of several novel encoding methods [19] for dynamic scene recognition is presented. Third, based on the outcome of this evaluation, the proposed local spacetime features are encoded via the locally constrained linear coding (LLC) approach [104] to form a general BoW model applicable to visual recognition tasks. While the FV representation [78] has recently shown state-of-the-art results for scene classification from a single image [48], object detection [20, 21], face verification [92], or action and event recognition [74], this does not appear to hold for dynamic scene understanding, where the present work indicates that LLC [104] performs especially well for the representation of highly dynamic scenes. Fourth, the impact of image stabilization for factoring out camera motion prior to feature extraction is explored. Fifth, to tailor the model specifically for dynamic scene recognition, a pooling scheme based on scene dynamics is proposed that directly builds on the encoded features. Therefore, the proposed pooling method inherits important local properties of the features and further originates efficiently as a by-product in the feature extraction process, *i.e.*, the visual words are pooled based on their temporal energy in the frequency domain which is computed efficiently from linear combinations of the extracted feature responses. An additional contribution of this chapter is the application of a temporal slice combination technique based on histogram intersection kernels.

The selected feature, encoding and pooling approaches have been assembled into a complete system for dynamic scene recognition, *i.e.*, the Bags of Spacetime Energies (BoSE). The presented methods are extensively evaluated by applying them to the two publicly available dynamic scene recognition datasets, Maryland [90] and YUPENN [27]. The proposed BoSE framework with the novel pooling scheme achieves overall best recognition accuracy on both scenes captured with and without camera motion. The experiments reveal the most crucial aspects for good performance and demonstrate that a carefully designed spacetime BoW model substantially outperforms the previous state of the art in the dynamic scene recognition literature.

4.2 Preliminaries and related work

The first part of this section reviews support vector machines and the pyramid match kernel for comparison between sets of features. The second part of the section introduces well-established codebook generation, feature encoding and pooling methods from the literature of object recognition and scene classification.

4.2.1 Support vector machines for classification

Support Vector Machines (SVMs) are discriminative classifiers that learn decision boundaries with a maximum margin for binary classification problems. The margin is defined as the distance between the separating hyperplane and the closest positive and negative training labels.

For training data consisting of feature vectors $\mathbf{v}_i \in \mathbb{R}^d$ and corresponding class labels $c_i \in \{-1, +1\}$, the algorithm searches a hyperplane $\mathbf{w} \cdot \mathbf{v} - b = 0$, which best separates the data based on the maximum margin to the nearest training examples (*i.e.* the support vectors). Therefore, the separating plane is only influenced by the closest training examples. Here, $\mathbf{w} \in \mathbb{R}^d$ is the normal vector to the hyperplane and b is the distance of the hyperplane to the origin.

If the data is not linearly separable, soft margin SVMs, proposed by Cortes and Vapnik [22], allow misclassified examples during training by adding a slack variable ξ_i to each training example $\mathbf{v}_i$. The slack variable is computed by a loss function $f(\mathbf{v}_i)$ and adds a misclassification penalty to the SVMs objective function:

$$\min_{\mathbf{w},\xi,b} \left\{ \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \xi_i \right\} \tag{4.1}$$

subject to

$$c_i(\mathbf{w} \cdot \mathbf{v}_i - b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \tag{4.2}$$

Minimizing $\|\mathbf{w}\|$ assures maximum distance between the hyperplane and its support vectors (*i.e.* $\frac{2}{\|\mathbf{w}\|}$), while minimizing over the slack variables ξ_i assures a small error on the training data. Therefore, the optimization process is a tradeoff between a large margin and a small error penalty. This tradeoff between regularization and constraint violation is

controlled by the parameter C.

4.2.2 Pyramid match kernel

As reviewed in the previous section, SVMs are linear classifiers, which classify based on a dot product between the feature point $\mathbf{v}_i$ and a learned hyperplane. The replacement of the dot product by a kernel function makes nonlinear feature separation possible. The hyperplane is then constructed in a transformed, possibly high dimensional, feature space. Comparison of two features $\mathbf{v}, \mathbf{z} \in \mathbb{R}^d$ is achieved by using a dot product $k(\mathbf{v}, \mathbf{z}) = \phi(\mathbf{v}) \cdot \phi(\mathbf{z})$, where $k(\mathbf{v}, \mathbf{z}) : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ is the kernel function.

In [39], Grauman and Darrell propose the pyramid match kernel for matching hierarchical histogram pyramids with the intersection as similarity measure. The intention of the pyramid match kernel is to match two sets of features at different resolutions in the feature space. This is achieved by generating histograms of varying bin-size and counting the number of features that land in the corresponding bin. For measuring the similarity between two histograms $H_l^i(\mathbf{v})$, $H_l^i(\mathbf{z})$ at level l, the histogram intersection is applied as

$$\mathcal{H}_l = \sum_i \min(H_l^i(\mathbf{v}), H_l^i(\mathbf{z})), \tag{4.3}$$

where i denotes histogram bins. Since matches found at level l are also included in matches at a coarser level $l+1$, Grauman and Darell [39] assign higher weights to similarity scores at finer levels. Therefore, the pyramid match kernel computes the weighted change of intersection at each of the L histogram levels

$$k(\mathbf{v}, \mathbf{z}) = \sum_{l=1}^{L} \frac{1}{2^{L-l+1}} (\mathcal{H}_l - \mathcal{H}_{l+1}). \tag{4.4}$$

4.2.3 Feature coding and pooling methods

Recently, research activity for improving feature encoding and spatial pooling for BoW based object classification approaches has increased drastically.

For visual scene recognition tasks, several different coding procedures exist to convert local features $\mathbf{v}_i \in \mathbb{R}^D$ into more effective local representations $\mathbf{f}_i \in \mathbb{R}^K$. To further convert the local codes into a global feature representation a spatial pooling operation ρ

is applied.

In this section, several popular strategies for encoding $\mathbf{v}_i$ with a trained codebook $\mathbf{B} \in \mathbb{R}^{D \times K}$, *i.e.* vector quantization, locality constrained linear coding, and (improved) Fisher vectors, as well as the two existing basic spatial pooling methods, *i.e.* average pooling and max pooling, are reviewed.

4.2.3.1 Codebook generation

Traditionally, a codebook with K visual words is learned in an unsupervised manner, *e.g.* by using K-means to cluster the descriptor space into K significant regions. Given N training features $\mathbf{v}_1, \ldots, \mathbf{v}_N \in \mathbb{R}^D$, the K-means algorithm searches assignments of the features to K sets $\mathbf{S}_i$ with cluster centres $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K \in \mathbb{R}^D$ such that the sum of squares error within the sets is minimized

$$\arg\min_{\boldsymbol{\mu}_i} \sum_{i=1}^{K} \sum_{\mathbf{v}_j \in S_i} \|\mathbf{v}_j - \boldsymbol{\mu}_i\|^2 . \tag{4.5}$$

This assures that each feature vector is assigned to the cluster set with the nearest mean. After convergence, the codebook $\mathbf{B} = \mathbf{b}_1, \ldots, \mathbf{b}_K \in \mathbb{R}^D$ is correspondingly given by the K mean vectors $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K \in \mathbb{R}^D$. The method proposed in this chapter uses the approximated nearest neighbour algorithm based on randomized test-bin-first KD-tree forests from [72], implemented in the VLFeat toolbox [100]. This allows convenient run-times by solving the cluster assignment (4.5) with an approximate nearest neighbour search.

4.2.3.2 Encoding via vector quantization

Vector quantization (VQ) is the most common encoding approach. This baseline coding method assigns one single codeword to each local descriptor $\mathbf{v}_1, \ldots, \mathbf{v}_N$, based on the minimum distance in the D-dimensional feature space

$$\arg\min_{\mathbf{f}_i} \sum_{i=1}^{N} \|\mathbf{v}_i - \mathbf{B}\mathbf{f}_i\|^2 , \tag{4.6}$$

with the constraints $\|\mathbf{f}_i\|_0 = 1, \|\mathbf{f}_i\|_1 = 1, \mathbf{f}_i \succeq 0$ such that only a single code with unit weight is assigned. By restricting the cardinality of $\mathbf{f}_i$ only the codeword with the smallest

Euclidean distance in the feature space is assigned to each local descriptor $\mathbf{v}_i$. Thus, $\mathbf{f}_i$ consists of only one non-zero element, indicating the nearest visual word in $\mathbf{B}$.

4.2.3.3 Encoding via locality constrained linear coding

Using only a single codeword does not incorporate distances in the feature space during coding, discards much descriptive information and is sensitive to noise [55, 66, 98]. Therefore, recent work has shown that using a representation of sparse codes achieves much higher classification performance [10, 11, 14, 19, 34, 46, 87, 104, 111]. Sparsity is achieved via regularization, since assigning too many codewords to a single descriptor would lead to overfitting. Lately, the most frequently used coding approach is Locality-constrained Linear Coding (LLC) introduced by Wang *et al.* [104]. LLC uses a sparse representation of local codes in the feature space. Each local feature $\mathbf{v}_i$ is encoded by $M \ll K$ codewords $\mathbf{b}_i$ which exhibit the lowest Euclidean distance to $\mathbf{v}_i$.

$$\underset{\mathbf{1}^\top \mathbf{f}_i = 1}{\arg\min} \sum_{i=1}^{N} \|\mathbf{v}_i - \mathbf{B}\mathbf{f}_i\|^2 + \lambda \|\mathbf{d}_i \odot \mathbf{B}\mathbf{f}_i\|^2 . \tag{4.7}$$

where $\odot$ is the component-wise product and $\mathbf{d}_i \in \mathbb{R}^K$ denotes the locality adaptor which measures the similarity of the codewords to the given feature vector $\mathbf{v}_i$:

$$\mathbf{d}_i = \exp\left(\frac{\text{dist}(\mathbf{v}_i, \mathbf{B})}{\sigma}\right), \tag{4.8}$$

where $\text{dist}(\mathbf{v}_i, \mathbf{B})$ is the Euclidean distance between $\mathbf{v}_i$ and the codewords $\mathbf{b}_1, \ldots, \mathbf{b}_M$, and σ controls the exponential weighting. It is notable that the coding speed for LLC is considerably higher than for vector quantization, because, not only the nearest neighbour, but the M nearest neighbours have to be sought for all the local features. To achieve reasonable encoding run-times this can be performed by an approximate nearest neighbour search [19].

4.2.3.4 Encoding via Fisher vectors

Fisher kernels combine generative methods, that concentrate on the modelling of a joint conditional distribution, and discriminative approaches, which focus on direct discrimination with a trained distribution [42, 78]. In contrast to a discriminative approach, gen-

erative methods are able to create new feature vectors by sampling from the modelled class-conditional distribution. Fisher vectors (FV) [19, 78, 80], that are a special case of the more general Fisher kernels, learn the distribution of local descriptors $\mathbf{v}_i \in \mathbb{R}^D$ by using a Gaussian Mixture Model (GMM) $p(\mathbf{v}_i|\boldsymbol{\theta})$, with parameters $\boldsymbol{\theta} = (w_k, \mu_k, \Sigma_k, k = 1, \ldots, K)$:

$$p(\mathbf{v}_i|\boldsymbol{\theta}) = \sum_{k=1}^{K} w_k p_k(\mathbf{v}_i|\mu_k, \Sigma_k), \tag{4.9}$$

with the components

$$p_k(\mathbf{v}_i|\mu_k, \Sigma_k) = (2\pi)^{-\frac{D}{2}} |\Sigma_k|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{v}_i - \mu_k)^\top \Sigma_k^{-1}(\mathbf{v}_i - \mu_k)\right), \tag{4.10}$$

where $|\cdot|$ denotes the determinant and, for the k^{th} Gaussian, w_k is the prior weight, fulfilling the constraint $\sum_{k=1}^{K} w_k = 1$, μ_k the mean vector, and Σ_k the covariance matrix. For each of the K codewords, a Gaussian encodes the relative frequency (w_k), the mean (μ_k) and the variation around the mean (Σ_k). For lower computational cost, the covariance matrices are restricted to be diagonal, since any distribution can be approximated with arbitrary precision by a weighted sum of Gaussians with diagonal covariances [78].

The Fisher vectors are used to represent the difference (first and second order) of a feature set and the average distribution of the training features, modelled by the GMM. Training of the $K(2D+1)$ GMM parameters is realized with the Expectation Maximization (EM) algorithm. For each of the $k = 1, \ldots, K$ mixtures, the posterior probability $p(k|\mathbf{v}_i, \boldsymbol{\theta})$ for a feature vector $\mathbf{v}_i$ is given as

$$p(k|\mathbf{v}_i, \boldsymbol{\theta}) = \frac{w_k p_k(\mathbf{v}_i|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} w_j p_j(\mathbf{v}_i|\mu_k, \Sigma_k)}. \tag{4.11}$$

Following [19], for each Gaussian mixture k, the mean and covariance vectors are given by

$$\boldsymbol{\Phi}_k^{(\mu)} = \frac{1}{N\sqrt{w_k}} \sum_{i=1}^{N} p(k|\mathbf{v}_i, \boldsymbol{\theta}) \Sigma_k^{-\frac{1}{2}} (\mathbf{v}_i - \mu_k) \tag{4.12}$$

and

$$\boldsymbol{\Phi}_k^{(\Sigma)} = \frac{1}{N\sqrt{2w_k}} \sum_{i=1}^{N} p(k|\mathbf{v}_i, \boldsymbol{\theta}) [(\mathbf{v}_i - \mu_k)\Sigma_k^{-1}(\mathbf{v}_i - \mu_k) - 1] \tag{4.13}$$

The final Fisher vector $\boldsymbol{\Phi}^{(\text{FV})}$ is created by stacking the first and second order differences

between the descriptors $\mathbf{v}_i$ and the K trained mixtures

$$\mathbf{\Phi}^{(\mathrm{FV})} = [\mathbf{\Phi}_1^{(\mu)}, \mathbf{\Phi}_1^{(\Sigma)}, \ldots, \mathbf{\Phi}_K^{(\mu)}, \mathbf{\Phi}_K^{(\Sigma)}]. \tag{4.14}$$

Note that this step implicitly performs a pooling of the local features into a vector representation. Consequently, the $2DK$-dimensional FV $\mathbf{\Phi}^{(\mathrm{FV})}$ encodes the differences between a set of test feature vectors and a learned GMM distribution from the training features.

Improved Fisher vectors

Regarding normalization of $\mathbf{\Phi}^{(\mathrm{FV})}$, Perronnin *et al.* [80] show that by using signed square rooting applied to each element of the FV, classification performance of Fisher vectors can be improved:

$$\mathbf{\Phi}_k^{(\mathrm{IFV})} = \mathrm{sign}(\mathbf{\Phi}_k^{(\mathrm{FV})})|\mathbf{\Phi}_k^{(\mathrm{FV})}|^{-\frac{1}{2}}. \tag{4.15}$$

Finally an ℓ_2 normalization is applied to yield the final improved Fisher vector (IFV). Further recent work as well has shown that using this normalization consistently improves the quality of the representation [45, 48, 79].

4.2.3.5 Average-pooling

A non-local representation is created by pooling the local feature codes $\mathbf{f}_i \in \mathbb{R}^K$ in spatial (sub-)regions by applying a feature pooling function ρ that delivers a joint distribution of codes in a given spatial region across all N locations.

Average pooling creates a histogram by summing all feature codes $\mathbf{f}_i$ in a region $\mathcal{R}$:

$$\rho_{\mathrm{avg}}(\mathcal{R}) = \frac{1}{|\mathcal{R}|} \sum_{i \in \mathcal{R}} \mathbf{f}_i, \tag{4.16}$$

where $|\mathcal{R}|$ denotes the number of visual words in region $\mathcal{R}$. This operator is equivalent to a ℓ_1-norm of the encoded feature statistics. Note that pooling in the region of interest is orderless and therefore provides invariance for local transformations, but also assumes a uniform spatial distribution of the local codes.

4.2.3.6 Max-pooling

Since average pooling is susceptible to noise [111], the max pooling operation is widely used for image classification. In the max-pooling procedure, only the strongest instance of each codeword occurring in all code vectors $\mathbf{f}_i$ over a region $\mathcal{R}$ is used for representation

$$\rho_{\max}(\mathcal{R}) = \max_{i \in \mathcal{R}} \mathbf{f}_i^{(k)}, \text{ for } k = 1, \ldots, K, \tag{4.17}$$

which conforms to the l_∞-norm of the encoded features. Max-pooling focuses on only the most salient codes in the local pooling region and therefore has been shown to be more discriminative than average pooling [11, 34, 104, 111].

Generally, for each visual word $\mathbf{b}_k$, the pooling operation can be described by a single equation

$$\rho(p) = \sum_{i=1}^{N} \left((\mathbf{f}_i^{(k)})^p \right)^{\frac{1}{p}}, \tag{4.18}$$

where $p = 1$ for average pooling and $p = \infty$ for the max-pooling operation.

It is notable that both average and max pooling discard the spatial distributions of the local codes. Advanced pooling techniques which consider geometric consistency during the statistical summarization process (4.18) have gained considerable attention in recent works. Feng *et al.* [34] have proposed a weighted ℓ_p-norm spatial pooling method to account for class-specific spatial feature distributions during pooling. By assuming that visual words of certain classes exhibit specific geometric properties they learn the spatial distribution for individual words to achieve higher discriminative pooling results. The training is performed to maximize the class separability by considering a local smoothness constraint on the spatial correlation of adjacent features. Cao *et al.* [14] enforce geometric consistency by using superpixel segments to generate more semantically meaningful spatial layouts (than traditional SPM) when pooling local codes into BoW histograms. Jia *et al.* [46] adaptively learn receptive fields to get better spatial regions for pooling. They adopt the idea of over-completeness by first using a large number of possible spatial regions and subsequently train a classifier with structured sparsity to only use a sparse subset of all the features. While these recent advances in pooling are more flexible, they do not adapt dynamically to the time varying information that is present in a given dynamic scene and their utility will thereby be limited in application to this problem domain.

4.2.4 Scene representation

The features developed in the upcoming Section 4.3 are used to construct an intermediate BoW representation. The encoding methods presented in this section are employed, combined with the respective pooling methods used in the original publications. As in the original publication, the pyramid match kernel is preferable for VQ-based encoding [58]. Therefore, for features encoded by vector quantization, BoW histograms are accumulated which is equivalent to average pooling. For LLC encoded features, the max-pooling operation [104] is needed for good performance when coupled with a linear SVM classifier. Fisher vectors are also favourably compared by linear SVM, *e.g.*, [45, 80].

The visual words are pooled using a spatiotemporal hierarchy. For pooling the visual words within the same temporal instance, spatial pyramid matching (SPM) [58] is used to employ coarse geometric information into the final feature vector. The weak spatial layout of the scene is captured by pooling in grids of size $(X \times Y \times T) \in \left\{(2^l \times 2^l \times 1)\right\}_{l=0}^{2}$, with three outer scales [51] corresponding to $l = 0, 1, 2$. Note that for $l = 0$ the representation is equivalent to an orderless BoW.

Following the original publications, the pooled features in each region are normalized properly. The vector quantized BoW histograms are normalized in each region using the ℓ_1-norm. In contrast, the ℓ_2-norm is applied to FV, IFV, as well as to LLC encoded features after they are pooled within the grid cells and concatenated into a global feature vector. This is important for classification with linear SVMs for which ℓ_2-normalization is optimal [101]. Note that the difference between FV and IFV is that prior to the ℓ_2-normalization IFVs are individually normalized via signed square rooting in each spatiotemporal subregion (4.15).

When comparing feature codes from different temporal instances of a sequence, camera, as well as scene dynamics may cause the captured image content to appear at different locations over time. Two approaches are investigated to temporally pool the visual words: Section 4.5 combines temporal slices, based on the amount of intersection between the visual words, and Section 4.6 proposes a novel method to pool the codewords based on their dynamic energies.

4.3 Local spacetime descriptor

In this section the representation for dynamic scene description is presented. In comparison to Chapter 3, the descriptor is based on spatiotemporal oriented measurements that jointly capture spacetime image appearance and dynamics. Local chromatic information is encoded by including colour channels. All features are extracted by filtering the input sequence at multiple scales to capture the multiscale characteristics of natural scenes.

4.3.1 Spacetime orientation features

To extract the representation of spacetime orientation the input volume is filtered with oriented filters. Similar to the CSO descriptor in Section 3.2, the proposed approach uses 3D Gaussian third-derivative filters as illustrated in Figure 3.3.

Filtering is performed at different 3D orientations θ_i and scales σ_j. To uniformly sample the 3D spacetime domain, the same filter orientations (3.5) as in Section 3.2 are used. The responses are point-wise squared and integrated to yield oriented spacetime energy measurements

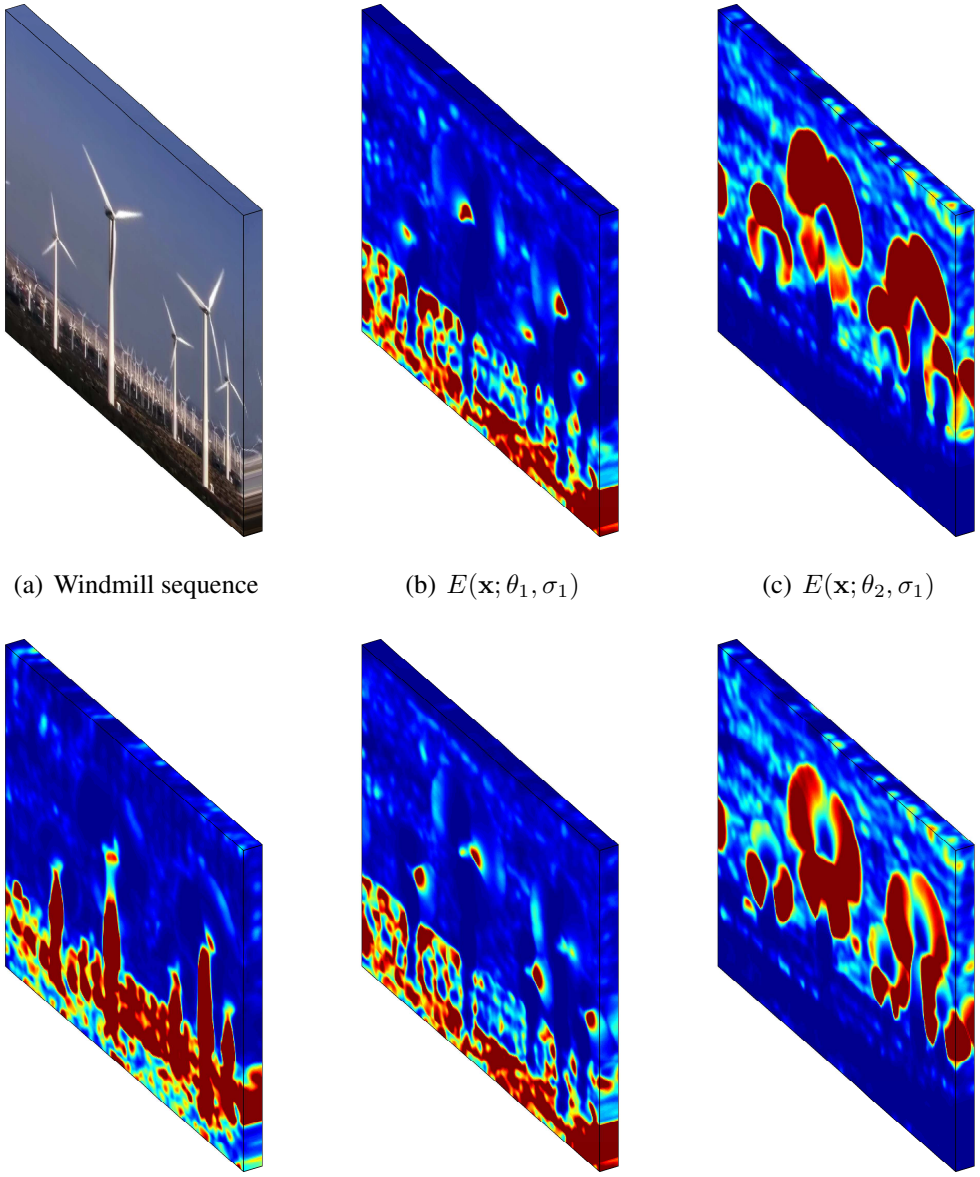
$$E(\mathbf{x}; \theta_i, \sigma_j) = G_{3D}(\sigma_j) * |G_{3D}^{(3)}(\theta_i, \sigma_j) * \mathcal{V}(\mathbf{x})|^2, \tag{4.19}$$

where $G_{3D}(\sigma_j)$ is a three-dimensional Gaussian with integration scale σ_j, $\mathbf{x} = (x, y, t)^\top$ are spacetime coordinates, $\mathcal{V}$ is the grayscale spacetime volume and $*$ denotes convolution.

Convolution with G_{3D} serves to blur the filter responses, thereby ameliorating phase sensitivity and suppressing noise. Furthermore, a smooth overlap between adjacent locations results, which avoids border effects without the need for additional normalization and interpolation, as, *e.g.*, required for gradient orientation features [63]. In contrast to previous work using similar oriented filter responses for dynamic scene recognition, which immediately aggregated filter responses over some support region (*e.g.*, [27]), here local responses are desired to drive subsequent encoding and correspondingly local smoothing is appropriate.

In Figures 4.2 and 4.3, the spatiotemporal energies for the employed filter orientations are depicted for a sample sequence showing a windmill farm. The energies collect dynamic information, see *e.g.* the dominant energies in Figure 4.3(b) capturing the move-

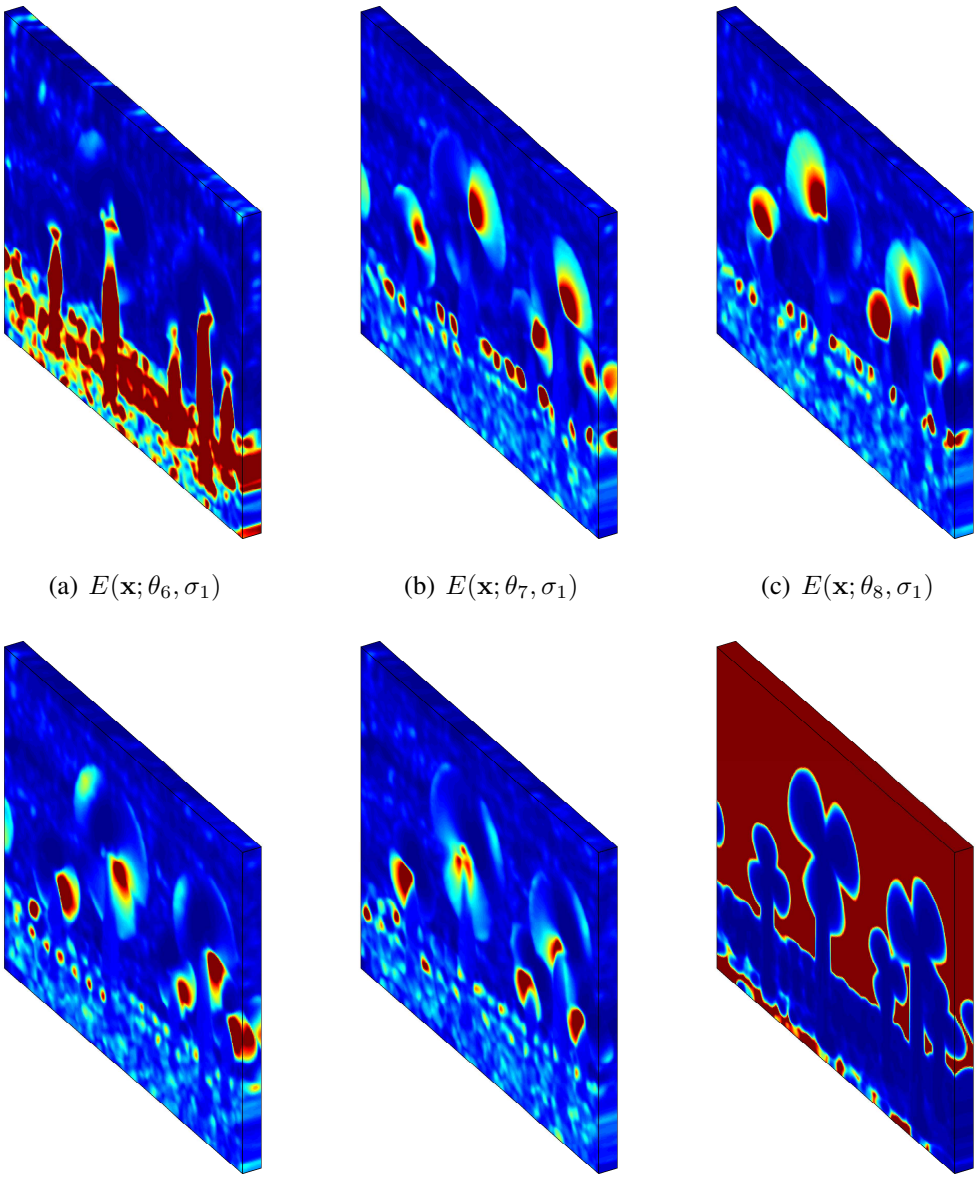
ment of the rotating rotor blades, as well as spatial information, see *e.g.* the energies in Figure 4.2(d), reaching high values for spatial orientation structure on the ground of the scene.



(a) Windmill sequence (b) $E(\mathbf{x}; \theta_1, \sigma_1)$ (c) $E(\mathbf{x}; \theta_2, \sigma_1)$

(d) $E(\mathbf{x}; \theta_3, \sigma_1)$ (e) $E(\mathbf{x}; \theta_4, \sigma_1)$ (f) $E(\mathbf{x}; \theta_5, \sigma_1)$

Figure 4.2: Distribution of spatiotemporal oriented energies of a windmill sequence form the YUPENN dataset. (a) shows a 16 frame slice from the sequence. (b)-(f) shows the distribution of the first five oriented energies, calculated using Gaussian derivative filtering and weighted accumulation over the filter support region (4.19). Warmer colours indicate larger filter responses.



(a) $E(\mathbf{x};\theta_6,\sigma_1)$ (b) $E(\mathbf{x};\theta_7,\sigma_1)$ (c) $E(\mathbf{x};\theta_8,\sigma_1)$

(d) $E(\mathbf{x};\theta_9,\sigma_1)$ (e) $E(\mathbf{x};\theta_{10},\sigma_1)$ (f) $E_\epsilon(\mathbf{x};\sigma_1)$

Figure 4.3: Five oriented spacetime energies of the temporal slice of length 16 frames in 4.2(a) are shown in Figure (f)-(e) and (f) illustrates the no structure channel. Warmer colours indicate larger filter responses.

4.3.2 Local contrast normalization

Similar as in (3.10), the spacetime orientation measurements are normalized with respect to the sum of all aggregated filter responses at a point to provide multiplicative photometric invariance,

$$\hat{E}(\mathbf{x};\theta_i,\sigma_j) = \frac{E(\mathbf{x};\theta_i,\sigma_j)}{\sum_{k=1}^{|\theta|} E(\mathbf{x};\theta_k,\sigma_j) + \epsilon}, \tag{4.20}$$

where $|\theta|$ denotes to the number of orientations, *i.e.* 10, and the noise bias ϵ avoids numerical instabilities at low overall energies. Again, to explicitly capture lack of oriented structure (*i.e.* homogeneous regions) another feature channel,

$$\hat{E}_\epsilon(\mathbf{x};\sigma_j) = \frac{\epsilon}{\sum_{k=1}^{|\theta|} E(\mathbf{x};\theta_k,\sigma_j) + \epsilon}, \tag{4.21}$$

is added to the contrast-normalized filter responses of (4.20). Figure 4.3(f) shows $\hat{E}_\epsilon(\mathbf{x};\sigma_j)$ for a windmill sequence, where large responses are seen in the unstructured sky region.

4.3.3 Chromatic features

Previous evaluations in [27, 90], and in Chapter 3, showed that integrating colour cues is useful for dynamic scene categorization. Chromatic information is incorporated in the present spacetime descriptor by aggregation of three locally weighted spacetime colour measurements as

$$C_k(\mathbf{x};\sigma_j) = G_{3D}(\sigma_j) * \mathcal{V}_k(\mathbf{x}), \tag{4.22}$$

where k is one of the three colour channels, $G_{3D}(\sigma_j)$ is a three-dimensional Gaussian filter with integration scale σ_j, $\mathbf{x} = (x, y, t)^\top$ are spacetime coordinates, $\mathcal{V}$ is the image sequence and $*$ denotes convolution. The CIE-LUV colour space [110], *i.e.* $k \in \{L, U, V\}$, is employed. It is of special note that the colour measurements are taken at the same scales as the spatiotemporal orientation measurements (4.19). Overall, at each sampled point, in the spatiotemporal image volume, $\mathcal{V}$, yields a locally defined, primitive feature vector, $\mathbf{v}_i$, that is formed by concatenating the normalized, multiscale orientation measurements, (4.20), with the measures of unstructuredness, (4.21), and colour, (4.22).

Preliminary experiments, in which the spacetime filtering (4.19) was applied separately to each of the three colour channels, led to lower recognition performance than explicitly capturing the local chromatic distribution as in (4.22). This result is explained

by the fact that separate spatiotemporal filtering on colour channels captures redundant information that is already encoded in the grayscale orientation measurements (4.19).

4.3.4 Coarse-scale dynamic features for pooling

The presented feature pooling method in this work builds on the coarse scale motion characteristics of the local features to be pooled. In order to provide robustness to camera movement, the spacetime filtering and integration in equation (4.19) is performed for relatively fine scales σ_j only. Coarse scale motion characteristics are extracted similar to the marginalized spatiotemporal orientation measurements E_{MST}, introduced in Chapter 3. For convenience, the respective equations for extracting marginalized spacetime energies are replicated in (4.23)-(4.24). Large scale spatiotemporal information is first extracted by aggregating the 3D Gaussian third derivative responses over a 3D region $\mathcal{R}$.

$$E^{\mathcal{R}}(\mathbf{x}; \theta_i, \sigma_j) = \sum_{\mathbf{x} \in \mathcal{R}} |G_{3D}^{(3)}(\theta_i, \sigma_j) * \mathcal{V}(\mathbf{x})|^2, \tag{4.23}$$

where $\mathcal{R}$ is a rectangular spacetime region defined by $\{\mathcal{R}_x, \mathcal{R}_y, \mathcal{R}_t\}$ and centred at $\mathbf{x}$. The reason for using a rectangular aggregation region is to be consistent with the rectangular pooling grids of a spatiotemporal pyramid.

Note that the phase dependency of the Gaussian third derivative filters is neutralized due to the summation over a spatiotemporal support region. Thus, the measures of signal energy in (4.19) and (4.23) are a function of spatiotemporal orientation and contrast only. For the proposed pooling method, only the dynamics of the features are relevant. To remove spatial information from the initial spatiotemporal orientation measurements, (4.23), they are summed across all orientations consistent with a single frequency domain plane [105]. Let the plane be defined by its unit normal, $\hat{\mathbf{n}}$, then measurements of orientation consistent with this plane are given as

$$E^{\mathcal{D}}(\mathbf{x}; \hat{\mathbf{n}}, \sigma_j) = \sum_{i=0}^{N} E^{\mathcal{R}}(\mathbf{x}; \hat{\theta}_i, \sigma_j), \tag{4.24}$$

with $\hat{\theta}_i$ denoting the equally spaced orientations consistent with $\hat{\mathbf{n}}$.

For the purpose of capturing image motion in various directions, a set of 11 spacetime energies $E^{\mathcal{D}}(\mathbf{x}; \hat{\mathbf{n}}, \sigma_j)$ are computed by steering the frequency domain plane $\hat{\mathbf{n}} =$

$[\hat{n}_x, \hat{n}_y, \hat{n}_t]^\top$:

- $\hat{\mathbf{n}}_s = [0, 0, 1]^\top \Longleftrightarrow$ static/no motion/orientation orthogonal to the image plane
- $\hat{\mathbf{n}}_r = [1, 0, 1]^\top \Longleftrightarrow$ rightward motion,
- $\hat{\mathbf{n}}_l = [-1, 0, 1]^\top \Longleftrightarrow$ leftward motion,
- $\hat{\mathbf{n}}_u = [0, 1, 1]^\top \Longleftrightarrow$ upward motion
- $\hat{\mathbf{n}}_d = [0, -1, 1]^\top \Longleftrightarrow$ downward motion,
- $\hat{\mathbf{n}}_{ru} = [\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 1]^\top \Longleftrightarrow$ diagonal rightward and upward
- $\hat{\mathbf{n}}_{lu} = [-\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 1]^\top \Longleftrightarrow$ diagonal leftward and upward
- $\hat{\mathbf{n}}_{ld} = [-\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}, 1]^\top \Longleftrightarrow$ diagonal leftward and downward
- $\hat{\mathbf{n}}_{rd} = [\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}, 1]^\top \Longleftrightarrow$ diagonal rightward and downward

4.3.5 Filtering details

The filters scales employed are $\sigma = \{1, 2\}$ with local filter support regions of $(x, y, t)^\top \in \left\{(13, 13, 13)^\top, (25, 25, 25)^\top\right\}$ pixels. For normalization, a bias of $\epsilon = 500$ is used for computing the ℓ_1-norm in (4.20). Notably, owing to the separability and steerabilty of the underlying filtering operations, the features can be extracted with modest computational expense.

4.4 Feature extraction on stabilized temporal slices

Features computed in the spatiotemporal domain (*e.g.*, optical flow, temporal gradients, or spacetime filtering employed in the previous section) are susceptible to camera movement. There exist several ways to remove disturbing camera movement from video. These methods typically first estimate the global image motion, then apply motion compensation, followed by an image inpainting to complete the missing image parts [68]. In this work only the first two steps are applied, since, here, the goal of stabilization is to facilitate spacetime feature extraction.

It has been shown recently that stabilization prior to feature extraction can improve recognition performance. For example, for the action recognition task, Jain *et al.* [44] achieve a significant performance improvement by compensating the dominant (camera) motion with an affine optical flow estimate. For pedestrian detection, Park *et al.* [77] show that very simple motion features, based on temporal differences calculated on weakly stabilized video frames, are able to achieve a five-fold reduction in false-positives. They use coarse scale Lucas-Kanade optical flow to align neighbouring frames, *i.e.*. they choose a large support window for the local flow estimation.

The estimation of the global image motion between the frames can be achieved either directly from *optical flow* measurements [5, 18, 68, 116] or by *feature-based* methods [13, 15, 16, 117]. The latter are generally faster than optical flow based alignment but more susceptible to errors.

To find the dominant motion for sequences with multiple moving objects, the application of robust methods, *e.g.*, RANSAC to estimate a planar homography, can be very tricky. For dynamic content, these methods are prone to errors in cases where large regions of the image consist of moving foreground objects, and therefore are not applicable for the domain of dynamic scene recognition where, *e.g.*, videos showing waterfalls exhibit large regions with dominant foreground motion.

For this reason, *i.e.* that finding the dominant background motion and separating it from the motion of the independently moving foreground objects is an unsolved problem, the stabilization approach in this work searches a global transformation between the frames. This allows the approach to be fairly robust to highly dynamic scenes with a large degree of foreground motion. Different camera motion models may be used for stabilization, *e.g.*, 2D translational models, translational + rotational + scaling models, affine

models, or 8-parameter homography models. In this particular work, global optical flow measurements are used to estimate the motion parameters using a translational and an affine motion model. Both models are evaluated explicitly in Section 4.7.

Similar to [69], global motion estimation [5] is used to estimate the inter-frame image transformation in order to stabilize the temporal slices of the video. First, the global motion between adjacent frames is estimated to chain the transformation of all frame pairs, starting from the centre frame of the slices, to get centre-frame relative motion estimations for all frames in a slice. The inter-frame motion is estimated by the hierarchical model proposed by Bergen *et al.* [5] and only applied for stabilization if a reliability measure is fulfilled. A description of the inter-frame motion estimation is given in Appendix A.

Does stabilization improve overall classification performance?

To evaluate the effect of different stabilization methods, several experiments are performed on both dynamic scene datasets: (1) YUPENN, that is captured by a static camera and (2) Maryland, which contains severe camera movement in several sequences. The minimum eigenvalue threshold (A.9), used to assure reliability of the optical flow estimation, is set to $T = 10^4$. Vector quantized feature codes are used for classification. To examine the influence of stabilization on overall recognition performance, each experiment is repeated 10 times due to the randomization in the codebook generation. Table 4.1 summarizes the potential benefits on classification performance for applying slice-based stabilization prior to feature extraction. The average classification rates for 10 experiments and the respective standard deviations of the individual experiments are reported. Both motion models discussed in this section are considered. As expected, stabilization negatively affects the classification performance on the YUPENN dataset, since the stabilization procedure may only introduce new motion effects by applying it to an already stabilized sequence, whereas on the Maryland dataset a benefit of the slice-based stabilization can be observed. Considering the different motion models in the stabilization method, little difference between a translational and an affine model can be found. However, by visually inspecting the stabilized sequences, the translational model produced more reasonable results, as it did not produce any scaling artefacts in the videos.

Detailed experimental results on stabilization can be found in the in Section 4.7.2.

YUPENN Dynamic Scenes Dataset			
Stabilization method	no stabilization	translational	affine
Classification rate (%)	94.68 ± 00.50	93.71 ± 00.21	92.92 ± 01.04
Maryland "In-The-Wild" Dataset			
Stabilization method	no stabilization	translational	affine
Classification rate (%)	65.51 ± 01.64	67.98 ± 00.94	66.92 ± 01.37

Table 4.1: Overall classification accuracy for different stabilization methods. Recognition rates drop slightly on the already stabilized YUPENN dataset. Contrarily, the feature extraction on stabilized slices facilitates recognition on the Maryland dataset.

4.5 Temporal slice combination based on histogram intersection

Previous approaches to dynamic scene recognition tend to collapse all image measurements over time [27, 90, 95]. Although pooling the features over time leads to a richer feature representation (*i.e.* all the available information is pooled from a given video), severe changes in visual appearance, *e.g.* due to camera movement or scene dynamics, may cause distortion errors in the resulting feature vector. The proposed temporal slice based aggregation avoids such distortions by pooling the features from very small durations. However, the amount of information captured by a single feature vector is thereby significantly lower.

The pyramid match kernel proposed by Grauman and Darrell [39], in combination with the traditional spatial pyramid [58], offers a convenient way to combine adjacent temporal slices based on the similarity of their histograms in the feature space.

Consider the 20 temporal slices of an example video shown in Figure 4.4. The histogram intersection between the local feature codes of each slice is illustrated below. The intersection kernel (4.4) varies between 1 for identical feature histograms and 0 for histograms without any intersecting bin. The construction of a representative feature vector for the whole clip by feature pooling across all temporal slices would integrate severe distortions in the pooled feature vector due to the dissimilarity of the individual slices.

This section investigates the possible benefit of combining similar adjacent temporal slices, *i.e.* the question "does temporal combination allow pooling to yield richer feature representations for classification" is investigated.

To that end, the pyramid match kernel between feature vectors, pooled from adjacent temporal slices, is examined. Adjacent BoW histograms H are merged, only if the histogram intersection k between all the features in the temporal neighbourhood $\mathcal{N}$ is above a slice merge threshold γ:

$$H(\mathbf{f}_j) = H(\mathbf{f}_j) + H(\mathbf{f}_k) \text{ iff } \forall j,k \in \mathcal{N} : k(\mathbf{f}_j, \mathbf{f}_k) > \gamma. \tag{4.25}$$

where $+$ denotes bin-wise summation. The resulting combined BoW histograms are subsequently ℓ_1 normalized prior to classification. The comparison in equation (4.25) can be implemented efficiently, by searching all diagonal sub-matrices having only elements

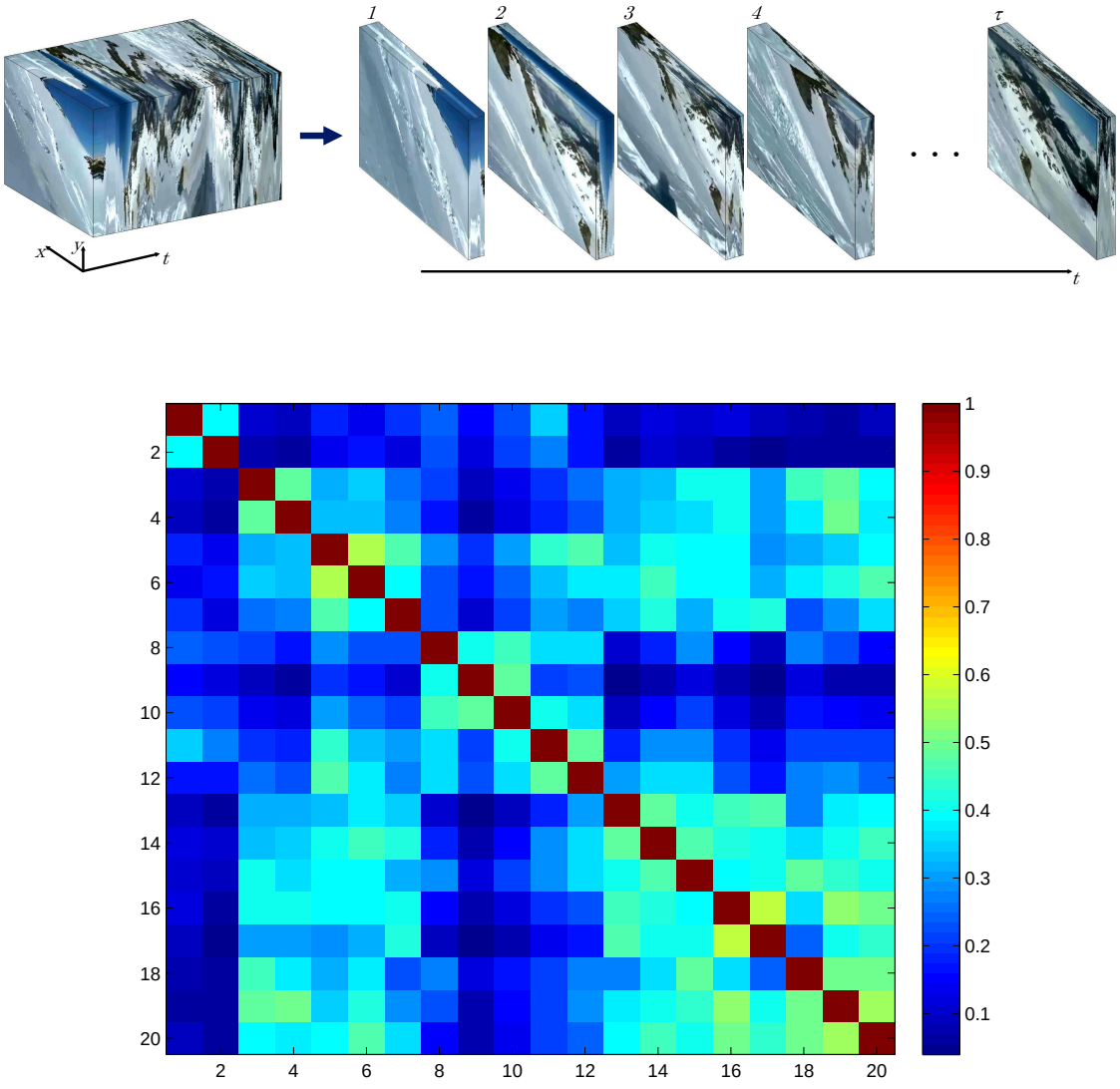


Figure 4.4: Histogram intersection kernel for the temporal slices of an avalanche sequence shown above. Abscissa and ordinate labels indicate slice numbers.

larger than γ in the precomputed classification kernel. For the corresponding experiments, presented in Section 4.7.3, γ is varied between 0 (merge all slices) and 1 (merge only identical feature histograms). An example is visualized in Figure 4.4, where the BoW histograms of all slices from number 13 to 17 exhibit relatively large intersection values and therefore could be merged as in (4.25).

4.6 Feature pooling based on static and dynamic energies

As previously shown in Section 4.4, cancelling the global image motion produces better spacetime image features when recognizing scenes captured with camera motion. However, for statically captured scenes, a performance loss is implied as the scene dynamics influence the stabilization algorithm inevitably.

This section addresses the aforementioned problem from another angle, *i.e.* the feature pooling step. Since, in SPM, features are pooled from spatial subregions, highly dynamic features (*i.e.* visual words with coarse scale image motion) are likely to be pooled from different spatial cells over time. Therefore, features that significantly change their spatial location across time should be pooled adaptively in a correspondingly dynamic fashion. For example, global image motion induced by a camera pan could cause the image features to move with time and pooling that is tied to finely specified image location will fail to capture this state of affairs. Similarly, when regions change their spatial relations with time, pooling should adapt. In such situations, a lack of appropriately dynamic pooling will degrade recognition performance, as features pooled at one location will have moved to a different location at a subsequent time and thereby be at risk of improper matching. Significantly, this challenge persists if the pooling positions are hierarchically arranged [58] or even more adaptively defined [14, 34, 46], but without explicit attention to temporal changes in pooling regions. In contrast, features that retain their image positions over time (*i.e.*, static patterns) can be pooled within finer, predefined grids, *e.g.*, as with standard spatial pyramid matching (SPM) [58]. Indeed, even highly dynamic features that retain their overall spatial position across time (*i.e.*, temporally stochastic patterns, such as fluttering leaves on a bush and other dynamic textures) can be pooled with fine grids. Thus, it is not simply the presence of image dynamics that should relax finely gridded pooling, but rather the presence of larger scale coherent motion (*e.g.*, as encountered with global camera motion).

This section presents a novel spatial pooling method for collecting the feature codes $\mathbf{f}_i$ in a region. The proposed method integrates the local scene dynamics into the pooling process. Dynamic features with coarse scale motion are intended to be pooled without geometric context, whereas static features are pooled by using increasingly finer SPM grids. The local dynamic information (4.24) is extracted closely related to the encoded features (4.19) and therefore provides a very intuitive and efficient strategy to enhance

the two existing pooling strategies in the literature: (i) Average-pooling (4.16), where the encoded spacetime features are counted additively in each region using histograms; and (ii) max-pooling (4.17), that, for each visual word, allocates the maximum response of the spacetime feature encodings in a given region. As described above, by following the original publications, a common hierarchical partitioning into spatial regions of size 1×1, 2×2, and 4×4 is employed to involve geometry in the pooling process.

4.6.1 Local Decomposition into dynamic spacetime energies

When locally pooling the encoded features from dynamic scenes, highly dynamic features that significantly change their location are likely to be pooled from different spatial regions in the spatial pyramid matching (SPM) scheme. Especially for highly dynamic scenes, or scenes captured with relatively large camera movement, this can substantially degrade recognition performance when using SPM. Here, to facilitate pooling, dynamic coefficients are introduced, based on the feature's local motion characteristics in the Fourier domain.

For this purpose, the multiscale spacetime energy decomposition (4.24) is used, which delivers spacetime energy factored into static energy ($E^{\mathcal{D}}(\mathbf{x}; \hat{\mathbf{n}}_s, \sigma_j)$), and energy across several directions ($E^{\mathcal{D}}(\mathbf{x}; \hat{\mathbf{n}}_*, \sigma_j)$)[1]. The spacetime region size $\mathcal{R}$ describes the window for integrating the filter measurements and therefore regulates the scale of the motion energy. Note that this shares some analogy with the scale in optical flow based motion estimation, *e.g.*, the window size in Lucas-Kanade flow [64].

The goal is to estimate coarse-scale motion that is used as a prior when incorporating geometric information in the pooling process. By setting the integration region $\mathcal{R}$ according to the smallest spatial region in the SPM partitioning, the estimated energy decomposition becomes robust to fine motions, but sensitive to coarse-scale motions. Therefore, the computed energy coefficients indicate objects moving at a scale-order of the finest grid in the spatial pyramid.

Let $\mathcal{V}(\mathbf{x})$ denote the spacetime slice in the filtering process (4.23) with width $\mathcal{V}_w$, height $\mathcal{V}_h$ and duration $\mathcal{V}_t$. Then, for a hierarchical 3-level spatial pyramid with the finest grid size of 4×4, the integration region for the energy responses is set to $\{\mathcal{R}_x, \mathcal{R}_y, \mathcal{R}_t\} = \left\{\frac{\mathcal{V}_w}{4}, \frac{\mathcal{V}_h}{4}, \mathcal{V}_t\right\}$.

[1] with $_*$ corresponding to the motion directions employed in 4.3.4, *i.e.*, $r, l, u, d, ru, lu, rd, ld$

The aggregated spacetime energies are not able to distinguish between coherent motion (*e.g.*, as exemplified by large scale motion resulting from camera movement) and incoherent motion (*e.g.*, as exemplified by stochastic dynamic textures) [1, 106]. Therefore, opponent-motion channels are added by computing the absolute arithmetic difference between energies of opponent directions

$$E^{\mathcal{D}}_{|r-l|}(\mathbf{x};\sigma_j) = |E^{\mathcal{D}}(\mathbf{x};\hat{\mathbf{n}}_r,\sigma_j) - E^{\mathcal{D}}(\mathbf{x};\hat{\mathbf{n}}_l,\sigma_j)| \tag{4.26}$$

$$E^{\mathcal{D}}_{|u-d|}(\mathbf{x};\sigma_j) = |E^{\mathcal{D}}(\mathbf{x};\hat{\mathbf{n}}_u,\sigma_j) - E^{\mathcal{D}}(\mathbf{x};\hat{\mathbf{n}}_d,\sigma_j)| \tag{4.27}$$

$$E^{\mathcal{D}}_{|ru-ld|}(\mathbf{x};\sigma_j) = |E^{\mathcal{D}}(\mathbf{x};\hat{\mathbf{n}}_{ru},\sigma_j) - E^{\mathcal{D}}(\mathbf{x};\hat{\mathbf{n}}_{ld},\sigma_j)| \tag{4.28}$$

$$E^{\mathcal{D}}_{|lu-rd|}(\mathbf{x};\sigma_j) = |E^{\mathcal{D}}(\mathbf{x};\hat{\mathbf{n}}_{lu},\sigma_j) - E^{\mathcal{D}}(\mathbf{x};\hat{\mathbf{n}}_{rd},\sigma_j)| \tag{4.29}$$

to yield a set of dynamic energies representing coherent image motion in 4 equally spaced directions (horizontal ($r-l$), vertical ($u-d$) and two diagonals ($ru-ld$ and $lu-rd$)). In contrast to the individual motion direction consistent energy samples from (4.24), the opponent motion channels explicitly capture coherent motion across various directions. For example, a spatial region with a stochastically moving spacetime pattern, *e.g.* the leaves of a tree in the wind can exhibit large motions in several specific directions $\hat{\mathbf{n}}$; however, after taking the absolute arithmetic difference from opponent directions, the coherent motions (4.26)-(4.29) of such stochastic spacetime texture patterns are approximately zero. On the other hand, regions that are dominated by a single direction of motion (*i.e.* coherent motion regions) will yield a large response in the most closely matched channel.

The coherent motion energies are ℓ_1 normalized together with the static energy channel that indicates lack of coarse motion,

$$\hat{E}^{\mathcal{D}}_{\Lambda_k}(\mathbf{x};\sigma_j) = \frac{E^{\mathcal{D}}_{\Lambda_k}(\mathbf{x};\sigma_j)}{\sum_{i\in\Lambda} E^{\mathcal{D}}_{\Lambda_i}(\mathbf{x};\sigma_j) + \epsilon},\ \forall k \in \Lambda, \tag{4.30}$$

to yield a pointwise distribution of static, coherent, as well as unstructured energy via the normalized ϵ indicating homogeneous regions,

$$\hat{E}^{\mathcal{D}}_{\epsilon}(\mathbf{x};\sigma_j) = \frac{\epsilon}{\sum_{i\in\Lambda} E^{\mathcal{D}}_{\Lambda_i}(\mathbf{x};\sigma_j) + \epsilon}, \tag{4.31}$$

$$\tag{4.32}$$

with $\Lambda = \{s, |r-l|, |u-d|, |ru-ld|, |lu-rd|\}$.

Since, regions without motion or with only fine scale motion (indicated by $\hat{E}^{\mathcal{D}}_{s}$), as well as homogeneous regions (indicated by $\hat{E}^{\mathcal{D}}_{\epsilon}$), should be pooled with geometric information, static energy is arithmetically combined with unstructured energy as

$$\hat{E}^{\mathcal{D}}_{s+\epsilon}(\mathbf{x};\sigma_j) = \hat{E}^{\mathcal{D}}_{s}(\mathbf{x};\sigma_j) + \hat{E}^{\mathcal{D}}_{\epsilon}(\mathbf{x};\sigma_j), \tag{4.33}$$

to yield the final set of (coherent) motion directions

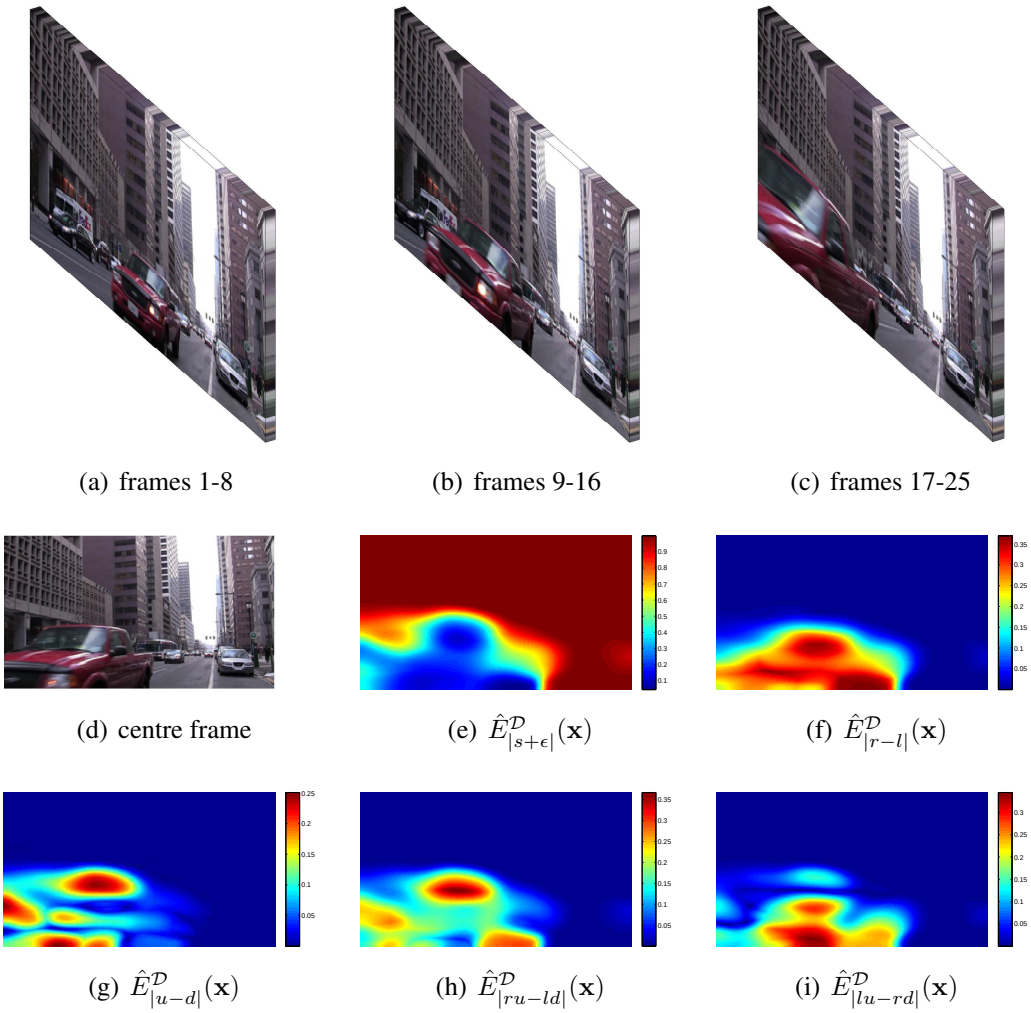
$$\Lambda = \{s+\epsilon, |r-l|, |u-d|, |ru-ld|, |lu-rd|\}\,. \tag{4.34}$$

The visual features are extracted with multiple scales σ_j. Since the same (multiscale) basis filters are steered for the encoded local features as well as for the dynamic pooling energies, the final set of multiscale pooling energies is computed by combination across scale

$$\tilde{E}^{\mathcal{D}}_{\Lambda_k}(\mathbf{x}) = \frac{1}{|\sigma|}\sum_{j=1}^{|\sigma|}\hat{E}^{\mathcal{D}}_{\Lambda_k}(\mathbf{x};\sigma_j),\ \forall k \in \Lambda, \tag{4.35}$$

where $|\sigma|$ denotes the number of scales.

In Figure 4.5, the dynamic pooling energies for a temporal subset of a street sequence are shown. Filtering is performed by Gaussian third derivative filters of scale $\sigma = \{1, 2\}$ with local filter support of $(x, y, t)^\top \in \left\{(13, 13, 13)^\top, (25, 25, 25)^\top\right\}$. For the purpose of proper illustration, the temporal support of the largest $G^{(3)}_{3D}$ filter is depicted in Figures 4.5(a)-4.5(c). Figure 4.5(d) depicts the central frame of the filtered sequence and 4.5(e)-4.5(i) show the decomposition of the filtered sequence into a distribution of static and directional dynamic energies. Observe that the static + unstructured channel consists of large responses for stationary image structures, *e.g.*, the buildings in the scene, as well as for homogeneous regions such as the sky in the centre of the scene. Whereas the foreground red car's dynamic energy can be decomposed into several coherent motion channels with a large part originating from the horizontal motion channel, *i.e.*, $\hat{E}^{\mathcal{D}}_{|r-l|}(\mathbf{x})$, shown in Figure 4.5(f). Note that fine-scale motions, such as the moving cars in the background, are not captured by the coherent motion channels (Fig. 4.5(f)-4.5(i)) and therefore exhibit strong responses in the static channel 4.5(e), which is appropriate as they form (part of) the background dynamic texture. Due to ℓ_1-normalization (4.30) the energies across all channels sum to one.



(a) frames 1-8 (b) frames 9-16 (c) frames 17-25

(d) centre frame (e) $\hat{E}^{\mathcal{D}}_{|s+\epsilon|}(\mathbf{x})$ (f) $\hat{E}^{\mathcal{D}}_{|r-l|}(\mathbf{x})$

(g) $\hat{E}^{\mathcal{D}}_{|u-d|}(\mathbf{x})$ (h) $\hat{E}^{\mathcal{D}}_{|ru-ld|}(\mathbf{x})$ (i) $\hat{E}^{\mathcal{D}}_{|lu-rd|}(\mathbf{x})$

Figure 4.5: Distribution of spatiotemporal oriented pooling energies of a street sequence form the YUPENN dataset. (a), (b), and (c) show the first 8, center 8 and last 9 frames of the filter support region. (e)-(i) show the decomposition of the sequence into a distribution of spacetime energies indicating rigidity/homogeneity in (e), and coarse coherent motion for several directions in (f)-(i). Warmer colours indicate larger filter responses.

4.6.2 Dynamic spacetime pyramid

Every temporal slice of the image sequence is represented by a single feature vector computed from a set of locally pooled feature descriptors. The pooling process extracts important statistics based on the feature codes in the pooled region.

A spatiotemporal pyramid is proposed here that specifically captures spatial and temporal information as described in the following four steps. First, to keep weak geometry information of the pooled encodings, a traditional 3 level spatial pyramid of size $\left\{(2^l \times 2^l \times 1^l)\right\}_{l=0}^{2}$ is constructed for each temporal slice of the input video, resulting in $M = 21$ regions $\{\mathbf{R}_m\}_{m=1}^{M}$. Second, for pooling at the coarsest level $l = 0$, *i.e.* in the region without geometric grid, $\mathbf{R}_1$, classical average- or max-pooling is applied. Third, in regions with geometric grids *i.e.* $l > 0$ and $\{\mathbf{R}_m | m > 1\}$, the static pooling energies $\hat{E}^{\mathcal{D}}_{s+\epsilon}$ are used as geometric coefficients, emphasizing the local contribution of each visual word. Fourth, to explicitly pool features favourably from regions with coherent motion, four more channels $\Lambda = \{|r-l|, |u-d|, |ru-ld|, |lu-rd|\}$ are added. Due to the coarse-scale motion of these features, the lowest pyramid scale $l = 0$ is used for those. Therefore, the final spatiotemporal pyramid encodes a temporal slice in $M+4 = 25$ channels, with each channel capturing specific spatial and temporal properties of the pooled codewords.

4.6.2.1 Dynamic average-pooling

For average pooling, the statistics $\tilde{\mathbf{f}}_i^{(k)}$ of the k^{th} codeword in a given region of interest $\mathbf{R}_m$ are then pooled as

$$\tilde{\mathbf{f}}_m^{(k)} = \frac{1}{|\mathbf{R}_1|} \sum_{\mathbf{x} \in \mathbf{R}_1} \mathbf{f}^{(k)}(\mathbf{x}), \qquad \text{for } m = 1 \tag{4.36}$$

$$\tilde{\mathbf{f}}_m^{(k)} = \frac{1}{|\mathbf{R}_m|} \sum_{\mathbf{x} \in \mathbf{R}_m} \hat{E}^{\mathcal{D}}_{|s+\epsilon|}(\mathbf{x}) \mathbf{f}^{(k)}(\mathbf{x}), \qquad \text{for } 2 \leq m \leq M \tag{4.37}$$

$$\tilde{\mathbf{f}}_m^{(k)} = \frac{1}{|\mathbf{R}_1|} \sum_{\mathbf{x} \in \mathbf{R}_1} \hat{E}^{\mathcal{D}}_{\Lambda_{m-M}}(\mathbf{x}) \mathbf{f}^{(k)}(\mathbf{x}), \qquad \text{for } M+1 \leq m \leq M+4, \tag{4.38}$$

where $|\mathbf{R}_m|$ denotes the number of words in region $\mathbf{R}_m$.

The stationary coefficients $\hat{E}^{\mathcal{D}}_{|s+\epsilon|}$ assign higher weights to all features pooled from regions with a spatial grid (4.37). Note that these static weights $\hat{E}^{\mathcal{D}}_{|s+\epsilon|}$ are ℓ_1-normalized

together with the dynamic energies that indicate coarse scale coherent motion. Therefore, dynamic features with coarse scale motion characteristics are given low weights when pooling in spatial grids of the SPM. To explicitly model the visual words with coarse scale dynamics, equation (4.38) pools features with specific directions. For example, visual words on horizontally moving objects are pooled with high corresponding weights $\hat{E}^{\mathcal{D}}_{|r-l|}$ to explicitly capture horizontally moving image structures in the dynamic spacetime pyramid.

4.6.2.2 Dynamic max-pooling

For a specific visual word, max-pooling finds the most salient response in a region $\mathbf{R}_m$. The proposed dynamic max-pooling operation finds the location $\mathbf{x}^{(k)}_m$ of a response, which is salient and exhibits some desired dynamics, by using the local distribution of the image dynamics as a geometric prior. Again, the dynamic energy distribution $\hat{E}^{\mathcal{D}}$ is used as a weighting for the pooling locations and four more channels are again added to explicitly capture the coarse-scale motion without geometric context

$$\mathbf{x}^{(k)}_m = \begin{cases} \arg\max_{\mathbf{x}\in\mathbf{R}_1} \mathbf{f}^{(k)}(\mathbf{x}) & \text{for } m = 1 \\ \arg\max_{\mathbf{x}\in\mathbf{R}_m} \hat{E}^{\mathcal{D}}_{|s+\epsilon|}(\mathbf{x})\mathbf{f}^{(k)}(\mathbf{x}) & \text{for } 2 \leq m \leq M \\ \arg\max_{\mathbf{x}\in\mathbf{R}_1} \hat{E}^{\mathcal{D}}_{\Lambda_i}(\mathbf{x})\mathbf{f}^{(k)}(\mathbf{x}) & \text{for } M+1 \leq m \leq M+4, \end{cases} \tag{4.39}$$

with $\Lambda = \{|r-l|, |u-d|, |ru-ld|, |lu-rd|\}$. The statistic signature for the k^{th} visual word is then given as

$$\tilde{\mathbf{f}}^{(k)}_m = \mathbf{f}^{(k)}(\mathbf{x}^{(k)}_m). \tag{4.40}$$

A concatenation generates the final description of $\mathbf{R}_m$ by the code vector $\tilde{\mathbf{f}}_m = [\tilde{\mathbf{f}}^{(1)}_m, \tilde{\mathbf{f}}^{(2)}_m, \cdots, \tilde{\mathbf{f}}^{(K)}_m]^\top \in \mathbb{R}^K$. This pooling procedure is termed as *dyn-max* in the remainder of this thesis.

Another closely related dynamic max-pooling alternative is evaluated in this thesis. This method differs from the (*dyn-max*) method above in the way that the static energies are used only for the finest grid of the spatial pyramid, *i.e.*, $4 \times 4 \times 1$, and only the energies indicating coherent horizontal and vertical motion (no diagonals) are added as

an orderless BoW:

$$\mathbf{x}_m^* = \begin{cases} \arg\max_{\mathbf{x}\in\mathbf{R}_m} \mathbf{f}^{(k)}(\mathbf{x}) & \text{for } 1 \leq m \leq 5 \\ \arg\max_{\mathbf{x}\in\mathbf{R}_m} \hat{E}^{\mathcal{D}}_{|s+\epsilon|}(\mathbf{x})\mathbf{f}^{(k)}(\mathbf{x}) & \text{for } 6 \leq m \leq M \\ \arg\max_{\mathbf{x}\in\mathbf{R}_1} \hat{E}^{\mathcal{D}}_{\Lambda_{m-M}}(\mathbf{x})\mathbf{f}^{(k)}(\mathbf{x}) & \text{for } M+1 \leq m \leq M+2 \end{cases}, \tag{4.41}$$

with $\Lambda = \{|r-l|, |u-d|\}$. This dynamic max-pooling alternative is subsequently termed as *dyn-max-alt*. For this approach, dynamic features can be pooled without (*i.e.* at pyramid level $l = 0$), or with coarse geometric information (*i.e.* at pyramid level $l = 1$). However, at the finest pyramid level (*i.e.*, $l = 2$), static features shall be pooled favourably. The main idea for this alternative is that dynamic features are more probable to generate an error at the finest pyramid scale. Moreover, as dynamic features are likely to be pooled in all coarse pyramidal regions (*i.e.*, $\{\mathbf{R}_m\}_{m=1}^5$), only two directional dynamic channels are added explicitly in equation (4.41).

Finally, a global feature vector $\tilde{\mathbf{f}}$, representing a temporal slice, is concatenated by stacking the descriptors $\tilde{\mathbf{f}}_m$ of all dynamic spacetime pyramid channels and an SVM classifier is consequently used to predict the class label of the temporal slice.

4.6.3 Summary of the implemented recognition procedure

1. **Sliding window local feature extraction.** The video is processed in a temporal sliding window by dense extraction of normalized oriented spacetime energies (4.19) and colour distributions (4.22), with the $|\theta| = 10$ filter orientations (3.5), one unstructured channel ($\hat{E}_\epsilon$), and three LUV colour channels. All measurements are taken at $|\sigma| = 2$ relatively fine scales (see the filtering details in Section 3.2.3) to describe only the local spacetime orientation structure of the video. The resulting multiscale spacetime orientation features of dimension $D = (|\theta|+1+3)\times|\sigma| = 28$ are extracted densely over a spatiotemporal grid by varying $\mathbf{x}$ in spatial steps of 8 pixels and temporal steps of 16 frames. Note that the largest employed scale uses 25-tap filters and therefore the descriptors are extracted with spatial and temporal overlap.

2. **Codebook generation.** For VQ and LLC, the codebook entries are learned by quantizing the extracted descriptors from the training sequences with K-means.

To maintain low computational complexity, a random subset of features from the training set, consisting of a maximum of 5000 descriptors from each training sequence, are used to learn a visual vocabulary of size $K = 200$ codewords. An approximated nearest neighbour (ANN) search [100] based on KD-tree forests [72] is used for clustering. In the case of Fisher vector coding, a GMM with $K_{\text{GMM}} = 50$ mixtures is fitted to the subsampled training descriptors. Moreover, the impact of varying the vocabulary size is evaluated in the remainder of this thesis.

3. **Feature coding.** The local spacetime descriptors are encoded via vector quantization (VQ), locality constrained linear coding (LLC), Fisher vectors (FV), or improved Fisher vectors (IFV). The parameters in LLC are set to the default values from the original publication [104]; *i.e.* the considered neighbouring visual words are set to $M = 5$ and the projection parameter is set to $\lambda = 10^{-4}$. Further, an ANN algorithm [100] is applied for searching the M nearest neighbours.

4. **Feature pooling.** A $l = 3$ level SPM is used to maintain weak spatial information of the features extracted in each temporal instance. The resulting 21 pooling regions from spatial grids of size $2^l \times 2^l$ create a $21 \times K = 21 \times 200 = 4200$ dimensional feature vector for VQ and LLC encoding and a $21 \times 2 \times K_{\text{GMM}} \times D = 21 \times 2 \times 50 \times 28 = 58800$ dimensional feature vector for Fisher encoding. The pooling is performed by taking the average (VQ), maximum (LLC) of the encoded features, or by taking the first and second order differences between the local descriptors and the trained GMM distribution for FV and IFV. Dynamic pooling adds four additional code vectors for the coherent motion channels to yield $25 \times K = 5000$ dimensional feature vectors.

   Further pooling experiments are performed with the proposed dynamic energies (4.24) in a dynamic average and dynamic max pooling fashion, for VQ and LLC encoded features, respectively. The dynamic energies in (4.24) are computed efficiently by steering the basis filter responses from step 1.

5. **Learning and Classification.** Each set of encoded features pooled from the same temporal instance generates a feature vector. For training, all feature vectors extracted from the training set are used to train a one-vs-rest SVM classifier. The histogram intersection kernel (4.4) is used for vector quantized features, while a linear SVM is applied for Fisher and LLC coded features. ℓ_2 normalization is ap-

plied to the feature vectors used in a linear SVM. The LIBSVM implementation [17] is used to find the hyperplane that separates the data points with a maximum margin in the feature space. The SVM's regularization loss trade-off parameter C is set after cross validation on the training data.

During classification, each feature vector of a test video is classified by the one-vs-rest SVM to yield a temporal prediction. All temporal predictions are subsequently combined to yield an overall classification of the video by the majority of the temporal class predictions.

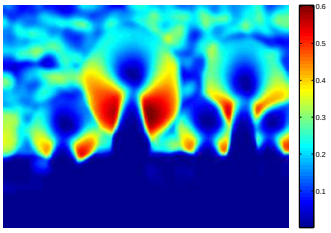
4.7 Experimental evaluation

This section evaluates the proposed codebook-based approach for dynamic scene recognition on the Maryland "In-The-Wild" [90] and YUPENN Dynamic Scenes [27] datasets. A leave-one-video-out recognition experiment is again used for consistency with previous evaluations in [27, 90]. The structure of the experiments is six-layered. At first (Section 4.7.1), the best encoding method, in the context of dynamic scene understanding, is sought for the proposed spacetime orientation features in Section 4.3. This evaluation includes novel feature coding methods [19] that either base on local codeword statistics (vector quantization and locality constrained linear coding), or the difference between the codewords and features to encode (Fisher vector coding). Second, Section 4.7.2 discusses the benefit of factoring out the global image motion by camera stabilization of temporal video slices. This is followed by experiments on the combination of similar temporal feature statistics with histogram intersection in Section 4.7.3. An evaluation of the novel dynamic pooling framework is given in Section 4.7.4 and the effect of different vocabulary sizes is examined in Section 4.7.5. Finally, in Section 4.7.6 the full proposed approach is compared with the state-of-the-art in dynamic scene classification.

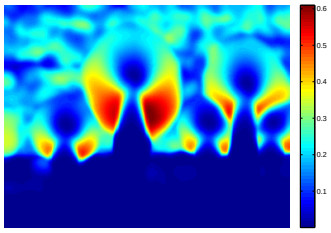
Intra-slice based feature aggregation

In this chapter, results for three different approaches to aggregate the features from a temporal slice, and therefore to collapse temporal information, are reported. These are: (i) mean-aggregation, that computes the average of all features along the temporal axis of a slice, (ii) max-aggregation, that selects the maximum feature response along the temporal axis of a slice, and (iii) mid-aggregation, that aggregates the features from the temporal centre of the slice. Note that this intra-slice based aggregation differs from the final pooling step of the coded features, since it operates on the extracted features directly, prior to the conventional coding and pooling steps reviewed in Section 4.2.3. As examples, the intra-slice based feature aggregation of three oriented energies, *i.e.* the energies in Figure 4.2(f), 4.3(b) and 4.3(e), is shown Figure 4.6. The overall difference between the aggregation methods is very small. The reason lies within the Gaussian smoothing of the filter measurements in equation (4.19).

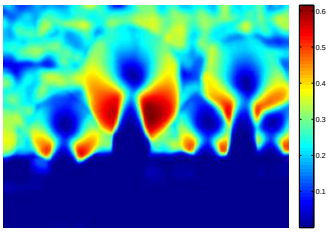
(a) Windmill slice



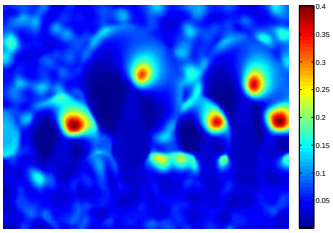
(b) $E(\mathbf{x};\theta_5,\sigma_1)|_{t=8}$



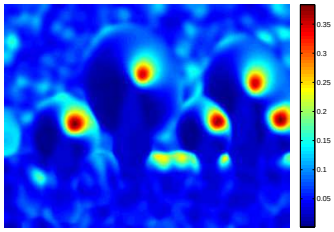
(c) $\underset{0\leq t\leq 15}{\text{mean}}\, E(\mathbf{x};\theta_5,\sigma_1)$



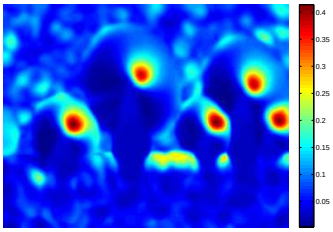
(d) $\underset{0\leq t\leq 15}{\max}\, E(\mathbf{x};\theta_5,\sigma_1)$



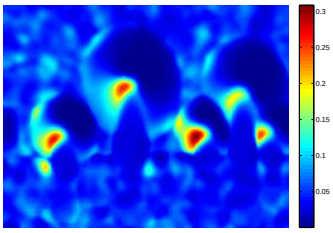
(e) $E(\mathbf{x};\theta_{10},\sigma_1)|_{t=8}$



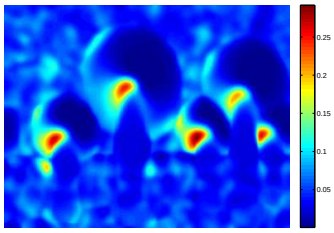
(f) $\underset{0\leq t\leq 15}{\text{mean}}\, E(\mathbf{x};\theta_7,\sigma_1)$



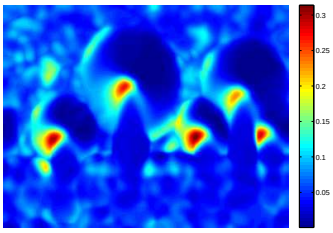
(g) $\underset{0\leq t\leq 15}{\max}\, E(\mathbf{x};\theta_7,\sigma_1)$



(h) $E(\mathbf{x};\theta_{10},\sigma_1)|_{t=8}$



(i) $\underset{0\leq t\leq 15}{\text{mean}}\, E(\mathbf{x};\theta_{10},\sigma_1)$



(j) $\underset{0\leq t\leq 15}{\max}\, E(\mathbf{x};\theta_{10},\sigma_1)$

Figure 4.6: Slice aggregation of spatiotemporal oriented energy distributions from a 16 frames temporal slice (*i.e.* $0 \leq t \leq 15$) of a Windmill sequence (a). The respective energies shown in Figure 4.2(f),4.3(b) and 4.3(e) are pooled by taking the central frame of the slice ((b), (e), and (h)), the average of the slice ((c),(f), and (i)), or the maximum of the slice ((d), (g), and (j)).

4.7.1 Comparison of feature coding methods

The effect of different encoding methods on the recognition performance is compared in this section. The investigated coding approaches are vector quantization (VQ), locality constrained linear coding (LLC), Fisher vectors (FV), and improved Fisher vectors (IFV), as reviewed in Section 4.2.3.

Stabilization method	Maryland dataset				YUPENN dataset			
	VQ	LLC	FV	IFV	VQ	LLC	FV	IFV
Unstabilized	65.38	**69.23**	63.85	66.92	94.52	95.48	91.43	**96.19**
Translational	70.00	**73.08**	66.92	67.69	93.33	94.05	93.57	**95.71**
Affine	69.23	**74.62**	67.69	68.46	92.38	93.10	89.29	**95.48**

Table 4.2: Average dynamic scene recognition accuracy with different feature coding methods. The classification performance averaged over all classes is shown for encoded spacetime features, extracted from unstabilized as well as stabilized sequences.

In Table 4.2, The overall recognition performance for the four different encoding approaches is listed when using different camera stabilization methods prior to feature extraction. On both datasets very competitive performance is achieved by the LLC encoding. Again, it can be seen that, for all methods, camera stabilization improves the classification performance on the Maryland dataset and decreases the performance on the YUPENN dataset. However, only the IFV encoding is able to maintain very close to the same degree of performance when applying camera stabilization to the YUPENN dataset. On the Maryland dataset, the higher order Fisher vector encodings are outperformed by LLC. This is interesting, given the fact that for other image classification tasks Fisher vectors are generally performing superior [19], or at least equally [48], to sparse coding methods such as LLC.

4.7.2 Temporal slice-based stabilization using different camera motion models

The following results comparing the different stabilization approaches are reported as best performance achieved for the listed parameters. Due to the randomization in the K-means clustering and quantization in the codebook generation process, results may vary for subsequent experiments with fixed parameters. For average results from several experiments please consider the comparison in Table 4.1.

Image stabilization for vector quantized (VQ) features

For the reported results in Tables 4.3-4.6, vector quantization is used as coding scheme and the histogram intersection kernel is applied for classification in an SVM. The Tables show the classification performance for different motion models during stabilization, slice alignments in classification and slice aggregation in feature extraction. First, consider the *motion model* that describes either a translational (2 parameters) or an affine (six parameters) model in the optical flow estimation procedure (A.3). Comparing the translational to the affine motion model, it can be seen that the more complex affine model improves performance when camera motion is present. Compare Table 4.3 to 4.4, where *e.g.* for the Landslide class, that is captured with a large degree of camera movement, performance grows significantly with the more complex affine model. However, a more complex motion model leads to a higher deficit in recognition accuracy on statically captured sequences. Compare Table 4.5 to 4.6, where *e.g.*, the Windmill Farm class is recognized with lower accuracies under an affine motion model. This is due to the rotational movement of the rotor blades, which causes the stabilization algorithm to rotate the whole sequence and therefore decrease the quality of the extracted features. Second, the *slice alignment* can be either collapsed, which uses one feature vector for each video, or latent, that creates a feature vector for each temporal slice. Here, it can be observed that latent alignment yields overall better results on the Maryland dataset, whereas on the YUPENN dataset collapsing all visual words over time yields slightly higher recognition rates. This can be attributed to the larger amount of information captured by each feature vector for collapsed slice alignment. The low result for collapsing temporal information on Maryland is due to the large degree of temporal variation present in this dataset (*e.g.*, the Landslide class, with high temporal variation is classified poorly with collapsed slice

information). Despite the slight benefit of collapsing temporal slices on YUPENN, latent temporal alignment is preferred, due to the advantage of online classification and better performance on the Maryland dataset. Third, the *slice aggregation* can be either mean, max, or mid, that takes the average, maximum or centre measure of the features in each temporal slice for encoding. In Tables 4.3-4.6 it is observable that changing from mean to max aggregation has little effect on the performance achieved, with mean aggregation leading to overall more stable results. The mid-aggregation method produced similar results as the mean-aggregation approach and therefore is omitted here for clarity.

Translational motion model				
Slice alignment Slice aggregation	collapsed mean	collapsed max	latent mean	latent max
Avalanche	20	30	60	60
Boiling Water	90	90	70	70
Chaotic Traffic	90	90	90	80
Forest Fire	90	90	90	90
Fountain	70	60	60	30
Iceberg Collapse	80	80	70	70
Landslide	30	20	40	40
Smooth Traffic	80	80	70	70
Tornado	70	80	90	90
Volcanic Eruption	60	70	70	80
Waterfall	80	70	70	70
Waves	100	100	90	100
Whirlpool	10	40	40	50
Overall	66.92	69.23	70	69.23

Table 4.3: Average recognition rates for different slice alignment and aggregation strategies on the Maryland dataset. A translational stabilization is applied prior to feature extraction.

Affine motion model				
Slice alignment Slice aggregation	collapsed mean	collapsed max	latent mean	latent max
Avalanche	20	20	40	30
Boiling Water	90	90	70	70
Chaotic Traffic	90	90	100	80
Forest Fire	80	90	90	90
Fountain	50	50	40	40
Iceberg Collapse	70	70	60	60
Landslide	60	40	60	40
Smooth Traffic	60	70	70	70
Tornado	80	80	100	100
Volcanic Eruption	50	50	50	50
Waterfall	70	90	80	70
Waves	100	90	100	90
Whirlpool	10	50	40	40
Overall	63.85	67.69	69.23	63.85

Table 4.4: Stabilizing the Maryland dataset: Average classification rates for different slice alignment and aggregation strategies. An affine stabilization is applied prior to feature extraction.

Translational motion model				
Slice alignment Slice aggregation	collapsed mean	collapsed max	latent mean	latent max
Beach	93	97	97	100
Elevator	97	97	97	97
Forest Fire	93	93	87	87
Fountain	83	83	80	83
Highway	97	100	100	97
Lightning	93	97	93	97
Ocean	100	100	100	100
Railway	97	97	100	100
Rushing River	90	90	87	90
Sky-Clouds	93	90	93	90
Snowing	100	100	97	93
Street	100	100	100	100
Waterfall	83	80	93	80
Windmill Farm	100	97	83	83
Overall	94.29	94.29	93.33	92.62

Table 4.5: Stabilizing the YUPENN dataset: Average classification rates for different temporal slice alignment and aggregation strategies. A translational stabilization is applied prior to feature extraction.

Affine motion model				
Slice alignment Slice aggregation	collapsed mean	collapsed max	latent mean	latent max
Beach	93	93	93	93
Elevator	97	97	97	97
Forest Fire	93	93	87	90
Fountain	80	77	83	87
Highway	100	97	100	93
Lightning	93	93	93	93
Ocean	100	100	100	97
Railway	93	93	97	93
Rushing River	97	93	93	93
Sky-Clouds	90	83	97	97
Snowing	100	97	93	93
Street	93	93	97	97
Waterfall	80	83	90	87
Windmill Farm	93	87	83	80
Overall	93.10	91.43	93.10	92.14

Table 4.6: Stabilizing the YUPENN dataset: Average classification rates for different temporal slice alignment and aggregation strategies. An affine stabilization is applied prior to feature extraction.

Image stabilization for locality constrained linear coding (LLC)

Since varying the temporal feature aggregation made little difference in the previous experiments, it is now fixed to mid-aggregation which takes the centre measure of the features in each temporal slice for the subsequent encoding. In Tables 4.7 and 4.8, the recognition rates for LLC encoded features, extracted from stabilized sequences are shown. A direct comparison between unstabilized and stabilized performance is given for the individual classes. Overall, a similar trend as for VQ codes can be observed. The more complex the motion model gets, the larger is the performance gain on Maryland (see Table 4.7) and the lower gets performance on YUPENN (see Table 4.8).

	Stabilization method		
	Unstabilized	Translational	Affine
Avalanche	60	60	70
Boiling Water	70	70	70
Chaotic Traffic	80	90	90
Forest Fire	90	90	90
Fountain	70	70	80
Iceberg Collapse	50	60	60
Landslide	60	50	50
Smooth Traffic	70	60	50
Tornado	90	90	90
Volcanic Eruption	70	80	90
Waterfall	60	80	90
Waves	70	100	100
Whirlpool	60	50	40
Overall	69.23	73.08	74.62

Table 4.7: Stabilizing the Maryland dataset: Recognition accuracy for LLC encoded spacetime features extracted from stabilized frames.

	Stabilization method		
	Unstabilized	Translational	Affine
Beach	100.00	96.67	90.00
Elevator	96.67	96.67	96.67
Forest Fire	93.33	90.00	90.00
Fountain	83.33	83.33	86.67
Highway	100.00	96.67	90.00
Lightning Storm	96.67	96.67	93.33
Ocean	100.00	100.00	100.00
Railway	100.00	100.00	96.67
Rushing River	93.33	86.67	96.67
Sky-Clouds	96.67	100.00	93.33
Snow	93.33	96.67	96.67
Street	100.00	100.00	100.00
Waterfall	83.33	76.67	80.00
Windmill Farm	100.00	96.67	93.33
Overall	95.48	94.05	93.10

Table 4.8: Stabilizing the YUPENN dataset: Recognition accuracy for LLC encoded spacetime features extracted from stabilized frames.

Image stabilization for improved Fisher vectors (IFV)

The impact of stabilization on IFV encoded features is shown in Tables 4.9 and 4.10, for the Maryland and YUPENN datasets, respectively. The same trend in performance as for VQ and LLC coding can be observed; however, here it is remarkable that stabilization only causes a slight drop in performance on YUPENN (from 96.19% to 95.48%). It is also noteable that, among the other feature encodings, IFV is the only encoding which is able to perform flawlessly on the Snow class (under a latent temporal alignment). Thus, it seems that IFV is able to better encode fine visual information (*e.g.*, snowflakes).

	Stabilization method		
	Unstabilized	Translational	Affine
Avalanche	60	60	70
Boiling Water	60	60	60
Chaotic Traffic	70	90	70
Forest Fire	70	70	80
Fountain	70	60	80
Iceberg Collapse	60	70	70
Landslide	50	50	50
Smooth Traffic	70	60	70
Tornado	90	80	70
Volcanic Eruption	60	80	70
Waterfall	70	70	70
Waves	90	100	90
Whirlpool	50	30	40
Overall	66.92	67.69	68.46

Table 4.9: Stabilizing the Maryland dataset: Recognition accuracy for IFV encoded spacetime features extracted from stabilized frames.

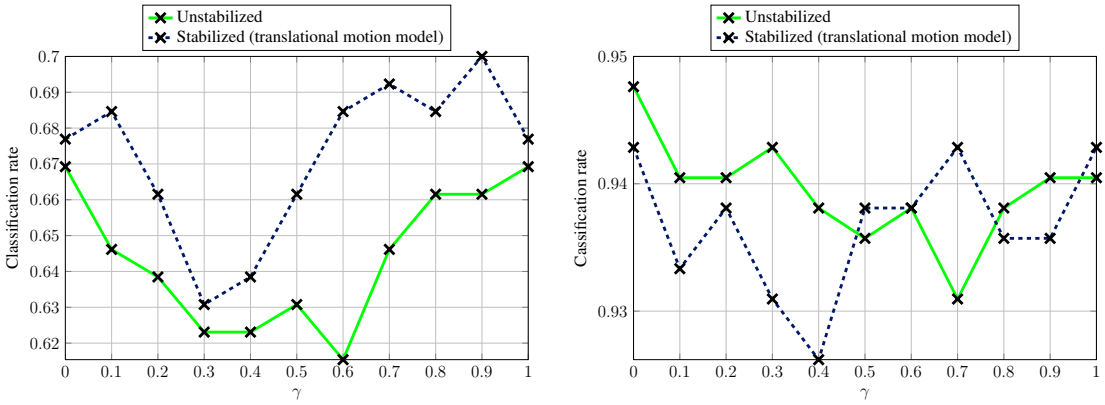
Note that detailed results are not reported for the basic FV encoding, as it performs significantly worse than IFV (see Table 4.2).

	Stabilization method		
	Unstabilized	Translational	Affine
Beach	96.67	100.00	96.67
Elevator	96.67	96.67	96.67
Forest Fire	93.33	90.00	93.33
Fountain	90.00	93.33	86.67
Highway	100.00	96.67	96.67
Lightning Storm	96.67	96.67	96.67
Ocean	100.00	100.00	100.00
Railway	100.00	100.00	96.67
Rushing River	96.67	90.00	96.67
Sky-Clouds	96.67	96.67	96.67
Snow	100.00	100.00	100.00
Street	100.00	100.00	100.00
Waterfall	83.33	80.00	86.67
Windmill Farm	96.67	100.00	93.33
Overall	96.19	95.71	95.48

Table 4.10: Stabilizing the YUPENN dataset: Recognition accuracy for IFV encoded spacetime features extracted from stabilized frames.

4.7.3 Temporal slice combination based on histogram intersection

In Figure 4.7, the classification performance on the two considered datasets (*i.e.* Maryland in 4.7(a) and YUPENN in 4.7(b)) is plotted for stabilized as well as unstabilized feature extraction. Similar visual word histograms of adjacent temporal slices are merged according to equation (4.25). The merging threshold γ is increased from 0 (merge all slices) to 1 (merge only identical feature histograms). Both plots indicate that combining adjacent slices does not provide any significant performance gain on either one of the datasets. In fact, there is no observable pattern as both plots show large fluctuations. Consequently, no merging at all, *i.e.* $\gamma = 1$, provides the best choice, as it allows for fast online classification with lowest latencies.



(a) Maryland dataset

(b) YUPENN dataset

Figure 4.7: Classification rate for merging visual words in temporal slices with unstabilized as well as stabilized features. On Maryland, stabilization increases performance notably on this dataset. On YUPENN, since the dataset is captured by a stabilized camera, the stabilization prior to feature extraction leads to a minor performance decrease.

The conclusion of this section is that combining similar temporal slices does not generally facilitate classification performance. One reason is that there is not much discriminative information that can be inferred from similar (*e.g.* adjacent) temporal slices. Alternatively, if there are large differences in inter-slice based appearance, the results in this section demonstrate that the classifier is able to learn the rich spatiotemporal patterns of the dynamic scene sequences, even if the same sequence exhibits large variations

in spacetime appearance. This conclusion can also be drawn from the results using a random forest classifier in Chapter 3. Furthermore, the histogram intersection approach limits the encoding to vector quantization and average pooling, since advanced coding strategies such as sparse coding tend to achieve decreased recognition performance when combined with intersection kernels [111].

4.7.4 Feature pooling based on static and dynamic energies

In this section, the proposed energy based pooling methods are evaluated. In Section 4.7.4.1, the presented dynamic average pooling approach from Section 4.6.2.1 is applied to VQ feature codes and Section 4.7.4.2 analyses the proposed dynamic max pooling methods from 4.6.2.2 with LLC encoded features. Note that any combination of encoding and pooling is possible; however, these choices have been made because max-pooling is necessary for good performance of LLC [104] and average-pooling works best for VQ codes [10]. Further, note that Fisher vectors implicitly perform a pooling step by calculating the first (4.12) and second order (4.13) differences between the trained parametric model and a set of descriptors in a given region. Therefore, FVs and IFVs are not evaluated with the proposed dynamic pooling operations.

4.7.4.1 Dynamic average-pooling for vector quantized codes

The extracted spacetime features are coded via vector quantization and pooled via dynamic averaging (see Section 4.6.2.1) of all codewords in a spatial region of a temporal slice. Results for different temporal feature aggregation, as well as different stabilization strategies are given in Tables 4.12. It is of special interest to compare the overall classification performance of the proposed dynamic pooling with the average pooling used in the accuracies reported in Table 4.1, which is replicated here in Table 4.11, for convenience.

YUPENN dataset			
Stabilization method	no stabilization	translational	affine
Classification rate (%)	94.68 ± 00.50	93.71 ± 00.21	92.92 ± 01.04
Maryland dataset			
Stabilization method	no stabilization	translational	affine
Classification rate (%)	65.51 ± 01.64	67.98 ± 00.94	66.92 ± 01.37

Table 4.11: Performance for average pooling of the VQ codewords from mean-aggregated features in unstabilized/stabilized temporal slices.

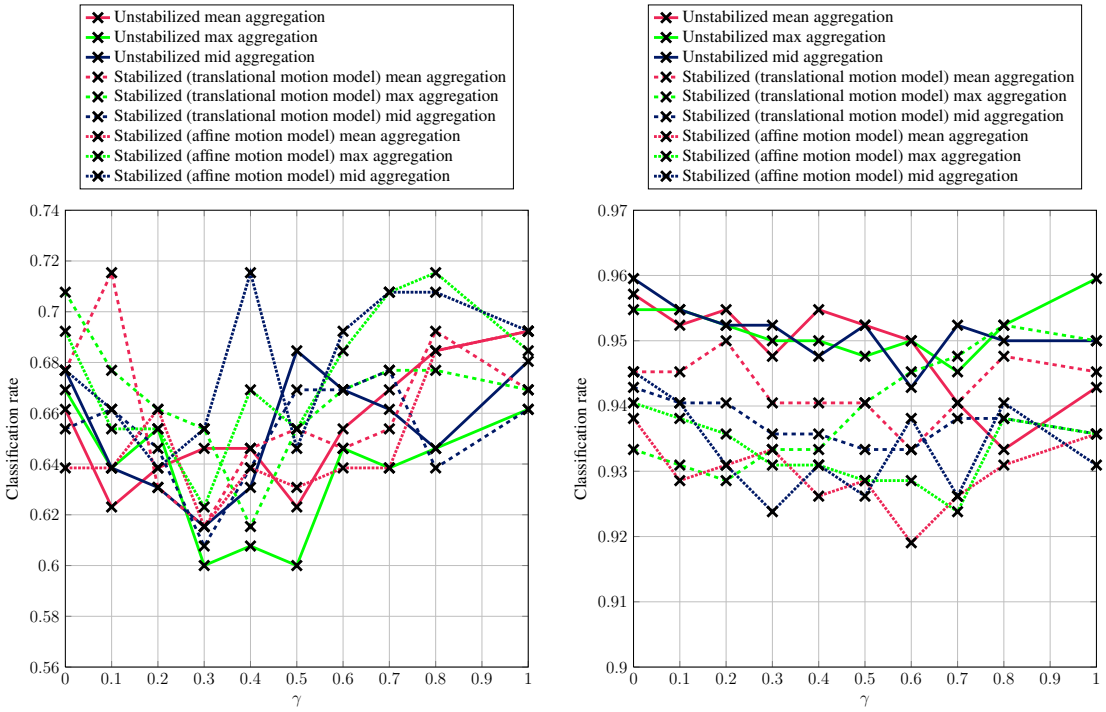
Regarding camera motion, one observes, that when using dynamic pooling the performance on the YUPENN dataset does not decrease as significantly as for average pooling

when applying stabilization prior to the feature extraction process (comparing 4.12 to 4.11). On the Maryland dataset, in contrast to the results for classic average pooling in Table 4.11, the performance does not generally increase when applying stabilization, but already yields best performance when used on features extracted without camera stabilization. Generally, the proposed dynamic pooling increases performance on both datasets (with and without camera motion) and therefore is superior to stabilization prior to feature extraction.

	Maryland			YUPENN		
Stabilization method	Temporal aggregation					
	mean	max	mid	mean	max	mid
Unstabilized	69.23	66.15	68.46	94.28	95.95	95.00
Translational	66.92	66.92	66.15	94.52	95.00	93.57
Affine	69.24	68.46	69.24	93.57	93.57	93.09

Table 4.12: Classification rate for dynamic average-pooling of VQ codes in unstabilized as well as stabilized temporal slices.

Figure 4.8 shows the results for dynamic average-pooling, combined with the temporal slice combination based on histogram intersection, evaluated in the previous section (4.7.3). It can again be observed that the combined pooling from adjacent slices does not generally increase recognition performance on both datasets, even with the dynamic pooling approach.



(a) Maryland dataset (b) YUPENN dataset

Figure 4.8: Dynamic average-pooling for VQ codes. Classification rate for merging visual words in temporal slices with unstabilized as well as stabilized features. Two stabilization models and three different temporal feature aggregation methods are compared.

4.7.4.2 Dynamic max-pooling for LLC-based codes

Results for LLC encoding and the three variants of max-pooling, *i.e.*, classic max-pooling (4.17), dynamic max-pooling (4.39), and the dynamic max pooling alternative (4.40), are shown in Table 4.13. Each experiment is repeated 5 times, due to the randomization in the codebook generation; *i.e.*, the subsampling of training data and the random intitalization in K-means clustering. One observes that the proposed dynamic max-pooling (*dyn-max*) outperforms the conventional max-pooling by around 6%. The alternative max-pooling (*dyn-max-alt*), which uses static energy coefficients only for max-pooling in the finest spatial grid as well as only added horizontal and vertical BoWs performs slightly worse.

Pooling	*max*	*dyn-max*	*dyn-max-alt*
Avalanche	64.00 ± 5.48	66.00 ± 5.48	72.00 ± 8.37
Boiling Water	70.00 ± 0.00	70.00 ± 0.00	70.00 ± 0.00
Chaotic Traffic	76.00 ± 5.48	82.00 ± 4.47	78.00 ± 4.47
Forest Fire	86.00 ± 5.48	90.00 ± 0.00	86.00 ± 5.48
Fountain	70.00 ± 0.00	70.00 ± 0.00	68.00 ± 4.47
Iceberg Collapse	50.00 ± 0.00	58.00 ± 4.47	60.00 ± 0.00
Landslide	60.00 ± 0.00	60.00 ± 0.00	60.00 ± 0.00
Smooth Traffic	70.00 ± 0.00	70.00 ± 0.00	70.00 ± 0.00
Tornado	90.00 ± 0.00	88.00 ± 4.47	90.00 ± 0.00
Volcanic Eruption	66.00 ± 5.48	68.00 ± 8.37	68.00 ± 8.37
Waterfall	64.00 ± 5.48	96.00 ± 5.48	82.00 ± 4.47
Waves	78.00 ± 10.95	88.00 ± 4.47	88.00 ± 4.47
Whirlpool	60.00 ± 0.00	78.00 ± 4.47	74.00 ± 5.48
Overall	$\mathbf{69.54 \pm 0.42}$	$\mathbf{75.69 \pm 1.17}$	$\mathbf{74.31 \pm 0.88}$

Table 4.13: Maryland dataset: Classification accuracy for the individual classes when using different pooling methods. Latent slice alignment, LLC encoding and no stabilization prior to feature extraction is used.

The significant performance gain due to dynamic max-pooling on the Maryland dataset may be attributed to the severe camera movement contained in this dataset. Since camera movement generally manifests itself at coarse temporal scales and the proposed dynamic pooling method favourably pools from locations without coarse motion, it is able to focus on informative features describing scene properties, rather than camera dynamics.

The experiments for different max-pooling strategies applied for classification on the YUPENN dataset are shown in Table 4.14. The proposed approach achieves exceptional high recognition rates of over 95%. A minor performance improvement of around 0.5% is obtained by the proposed dynamic max-pooling method. It is remarkable, that even for scenes captured with a stationary camera, the proposed dynamic pooling increases performance. One reason is that coherently moving objects are specifically matched by the dynamic pooling channels in equation (4.40). For example, vertically moving visual words from a waterfall sequence will be explicitly matched, since these are favourably pooled within the $\hat{E}^{\mathcal{R}}_{|u-d|}$ channel of the dynamic spacetime pyramid.

Pooling	*max*	*dyn-max*	*dyn-max-alt*
Beach	100.00 ± 0.00	100.00 ± 0.00	97.78 ± 1.92
Elevator	96.67 ± 0.00	96.67 ± 0.00	96.67 ± 0.00
Forest Fire	94.00 ± 1.49	93.33 ± 0.00	96.67 ± 0.00
Fountain	83.33 ± 0.00	86.00 ± 1.49	84.44 ± 1.92
Highway	100.00 ± 0.00	100.00 ± 0.00	98.89 ± 1.92
Lightning	96.67 ± 0.00	96.67 ± 0.00	96.67 ± 0.00
Ocean	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00
Railway	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00
Rushing River	93.33 ± 0.00	96.67 ± 0.00	94.44 ± 1.92
Sky-Clouds	96.67 ± 0.00	96.67 ± 0.00	96.67 ± 0.00
Snowing	93.33 ± 0.00	96.67 ± 0.00	93.33 ± 0.00
Street	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00
Waterfall	84.00 ± 1.49	82.67 ± 1.49	81.11 ± 1.92
Windmill Farm	100.00 ± 0.00	100.00 ± 0.00	98.89 ± 1.92
Overall	$\mathbf{95.57 \pm 0.21}$	$\mathbf{96.10 \pm 0.21}$	$\mathbf{95.40 \pm 0.27}$

Table 4.14: YUPENN dataset: Classification accuracy for the individual classes when using different pooling methods. Latent slice alignment, LLC encoding and no stabilization prior to feature extraction is used.

Collapsed temporal pooling of LLC codes

Next, the slice alignment for LLC-based encoded features is investigated. Recall that latent temporal alignment produces a feature vector for each temporal slice which is individually classified. Collapsed temporal alignment, however, aggregates the feature codes from all temporal slices in a sequence into the same feature vector which is used for classification. The previous evaluations in this section report results for latent temporal alignment.

Classification rates for collapsed alignment are reported in Tables 4.15 and 4.16 for Maryland and YUPENN, respectively. Note that max-pooling is still performed for each temporal slice separately, only the features are aggregated into a single vector before SVM classification. Both Tables indicate that, regardless of the pooling operation, the approach performs much worse than with latent slice alignment (see Tables 4.13 and 4.14).

Pooling	*max*	*dyn-max*	*dyn-max-alt*
Avalanche	14.00 ± 5.48	20.00 ± 7.07	22.00 ± 8.37
Boiling Water	80.00 ± 0.00	72.00 ± 8.37	68.00 ± 8.37
Chaotic Traffic	70.00 ± 0.00	62.00 ± 14.83	82.00 ± 10.95
Forest Fire	84.00 ± 5.48	96.00 ± 5.48	90.00 ± 7.07
Fountain	32.00 ± 10.95	24.00 ± 11.40	28.00 ± 8.37
Iceberg Collapse	30.00 ± 0.00	24.00 ± 5.48	30.00 ± 12.25
Landslide	22.00 ± 10.95	12.00 ± 13.04	14.00 ± 8.94
Smooth Traffic	34.00 ± 5.48	40.00 ± 14.14	42.00 ± 13.04
Tornado	80.00 ± 0.00	60.00 ± 14.14	74.00 ± 5.48
Volcanic Eruption	18.00 ± 10.95	26.00 ± 5.48	20.00 ± 0.00
Waterfall	56.00 ± 5.48	30.00 ± 18.71	42.00 ± 8.37
Waves	74.00 ± 5.48	72.00 ± 8.37	76.00 ± 5.48
Whirlpool	32.00 ± 10.95	26.00 ± 18.17	22.00 ± 4.47
Overall	**48.15 ± 0.42**	**43.38 ± 2.64**	**46.92 ± 1.80**

Table 4.15: Maryland dataset: Classification accuracy for the individual classes when using different pooling methods. Collapsed slice alignment, LLC encoding and no stabilization prior to feature extraction is used. The performance drops severely due to the collapsing of the max-pooled features over time.

An especially large degree of performance loss can be observed on the Maryland dataset (compare Table 4.13 with Table 4.15), which might be due to the large temporal

Pooling	*max*	*dyn-max*	*dyn-max-alt*
Beach	92.67 ± 1.49	91.33 ± 1.83	90.00 ± 0.00
Elevator	96.67 ± 0.00	96.67 ± 0.00	96.67 ± 0.00
Forest Fire	90.00 ± 0.00	87.33 ± 1.49	88.89 ± 3.85
Fountain	72.00 ± 2.98	70.00 ± 0.00	71.11 ± 1.92
Highway	96.67 ± 0.00	94.00 ± 1.49	94.44 ± 1.92
Lightning	88.00 ± 2.98	91.33 ± 4.47	87.78 ± 3.85
Ocean	89.33 ± 1.49	90.00 ± 0.00	91.11 ± 1.92
Railway	100.00 ± 0.00	95.33 ± 2.98	95.56 ± 1.92
Rushing River	96.67 ± 0.00	96.67 ± 0.00	96.67 ± 0.00
Sky-Clouds	86.67 ± 0.00	86.67 ± 0.00	87.78 ± 1.92
Snowing	96.67 ± 0.00	96.00 ± 1.49	96.67 ± 0.00
Street	100.00 ± 0.00	98.67 ± 2.98	100.00 ± 0.00
Waterfall	77.33 ± 1.49	80.00 ± 0.00	81.11 ± 1.92
Windmill Farm	96.67 ± 0.00	96.67 ± 0.00	98.89 ± 1.92
Overall	**91.38 ± 0.11**	**90.76 ± 57**	**91.19 ± 0.41**

Table 4.16: YUPENN dataset: Classification accuracy for the individual classes when using different pooling strategies for LLC codes. Collapsed slice alignment is applied.

variations in the dataset. Interestingly, for vector quantized codes the slice alignment does not have a large influence on recognition performance, as reported previously in Section 4.7.2. The reason is assumed to be that the spatial average pooling used in the VQ approach is analogue to the temporal averaging in the collapsed temporal alignment. On the other hand, when collapsing max-pooled LLC codes (by averaging) this leads to the large performance deficit reported in Tables 4.15 and 4.16. A more promising strategy for LLC-based features might be a max-selection over all temporal slices. However, this is not evaluated further since the latent temporal alignment allows for fast online classification.

4.7.4.3 Dynamic energy pooling or camera stabilization?

This section compares the efficiency of the proposed dynamic energy pooling with the feature extraction on stabilized temporal slices. The previous evaluations in this thesis have shown, that LLC encoded features perform best for recognizing dynamic scenes. Therefore, the performance of LLC-based feature codes that are pooled either via conventional max-pooling, or the proposed dynamic max-pooling, under application of different camera stabilization methods, is evaluated here.

Stabilization method	Maryland dataset		YUPENN dataset	
	max pooling	dyn-max pooling	max pooling	dyn-max pooling
Unstabilized	69.23	**77.69**	95.48	**96.19**
Translational	73.08	74.62	94.05	95.24
Affine	74.62	**77.69**	93.10	94.05

Table 4.17: Dynamic scene recognition accuracy with LLC encoded features for different pooling and camera stabilization methods. The proposed dynamic max-pooling allows best performance on data with a high degree of coarse scale motion (Maryland), as well as on dynamic scene sequences captured from static cameras (YUPENN).

In Table 4.17 the averaged overall classification rate for variations in the camera stabilization method prior to feature extraction is listed. On both datasets, the novel dynamic max-pooling leads to best performance. Conventional max-pooling is outperformed by a margin of 8.46% and 0.71% for Maryland and YUPENN, respectively. This is a very encouraging result, since, on Maryland, the proposed dynamic pooling clearly outperforms camera stabilization methods and, for the YUPENN dataset, this shows that there is still a performance increase from an already almost saturated accuracy of 95.48%. Based on this outcome, the proposed BoSE system makes use of LLC encoding and dynamic max-pooling, without applying camera stabilization prior to spacetime feature extraction.

4.7.5 Varying the size of the codebook

The final analysis of parameter variations in this thesis investigates the impact the vocabulary size in the feature encoding. The number of visual words, K, that represents the number of centroids in the BOW representation, and K_{GMM}, which denotes the number mixtures used in the GMM for Fisher vectors, is varied.

Table 4.18 presents the average performance for various sizes of the codebook. The resulting dimension of the feature vector for a single slice is listed as well. When increasing the codebook size, performance decreases from a certain point. One can observe that on the Maryland dataset a more complex codebook leads to a performance decrease, while on the YUPENN dataset an increased vocabulary size yields better performance up to a feature vector dimension of around 50000. For lowering the dimension of the final feature vector, principal component analysis could be applied to the encoded features. Generally, a low size of the vocabulary decreases discriminativity between the classes. On the contrary, a large vocabulary makes it difficult to find similar codewords within instances of the same class, as features, describing similar visual input, will be mapped to different codewords. This explains the performance decrease for larger codebooks on the Maryland dataset, because it exhibits higher intra-class variations than YUPENN.

LLC encoding					
K	100	200	400	1000	2000
Feature dimension	2500	5000	10000	25000	50000
Maryland	69.23	77.69	75.38	75.38	73.85
YUPENN	94.52	96.19	95.95	97.62	97.38
IFV encoding					
K_{GMM}	10	20	50	100	150
Feature dimension	11760	23520	58800	117600	176400
Maryland	64.62	67.69	66.92	66.92	66.15
YUPENN	93.57	95.00	96.19	95.95	95.24

Table 4.18: Recognition accuracy for different codebook sizes when using LLC encoded features pooled via the proposed dynamic max-pooling (*i.e.* BoSE), as well as IFV encoded features.

4.7.6 Comparison with the state-of-the-art

The proposed approach is compared to the CSO+STRF approach from the previous chapter and additionally to several alternative methods that have shown best performance previously [27, 90]. The methods are GIST [73] + (histograms of flow) HOF [67], GIST + chaotic dynamic features (Chaos) [90], spatiotemporal oriented energies (SOE) [27] and slow feature analysis (SFA) [95].

Tables 4.19 (Maryland dataset) and 4.20 (YUPENN dataset) compare the performance of the proposed Bags of Spacetime Energies (BoSE) system with the state-of-the-art. The BoSE consists of densely extracted local oriented spacetime energies (4.23) and colour distributions (4.22) that are encoded by LLC and pooled via the proposed dyn-max-pooling. Detailed description and parameter choices are given in Section 4.6.3. Note that the performance of SFA differs from that reported in their original paper [95]. According to the authors, this discrepancy is caused by a bug in the original implementation and the results presented here are the correct ones. An error report and the correct recognition rates can be found on the SFA website [2].

For both datasets, BoSE performs considerably better than the CSO+STRF approach from Chapter 3, which has defined a new state-of-the-art in classification and execution rates. On the Maryland dataset, the novel BoSE representation achieves a competitive average accuracy of 78% when coupled with a simple linear SVM classifier. When comparing to other approaches, one striking result is the 100% recognition accuracy for the Waterfall class. The proposed BoSE approach's 96% accuracy on the YUPENN suggests that performance is saturated on this dataset. One remarkable result on this dataset is the 87% recognition rate for the Fountain class, which exhibits huge intra-class variations in the background and only a small amount of common foreground (*i.e.* the fountain itself). Overall, BoSE is able to best represent the classes, by modelling the visual words with locally encoded spacetime energies that are pooled based on their dynamics.

[2] `http://webia.lip6.fr/~theriaultc/sfa.html`

Maryland "In-The-Wild"									
Features	HOF+ GIST	Chaos+ GIST	SOE		SFA	CSO (proposed)		BoSE (proposed)	
Classifier	NN	SVM	NN	RF	SVM	STRF	STRF	SVM	SVM
Temporal τ	*all*	*all*	*all*	*all*	*all*	1	*all*	1	*all*
Avalanche	20	60	10	40	60	60	60	50	60
Boiling Water	50	60	50	50	70	80	80	60	70
Chaotic Traffic	30	70	80	60	80	80	90	70	90
Forest Fire	50	60	40	10	10	80	80	70	90
Fountain	20	60	10	50	50	90	80	60	70
Iceberg Collapse	20	50	10	40	60	60	60	40	60
Landslide	20	30	50	20	60	20	30	50	60
Smooth Traffic	30	50	70	30	50	60	50	60	70
Tornado	40	80	60	70	70	90	80	90	90
Volcanic Eruption	20	70	30	10	80	50	70	60	80
Waterfall	20	40	20	60	50	50	50	100	100
Waves	80	80	80	50	60	60	80	70	90
Whirlpool	30	50	40	70	80	80	70	70	80
Overall	**33**	**58**	**42**	**43**	**60**	**66**	**68**	**65**	**78**

Table 4.19: Classification accuracy for different video descriptor and classifier combinations on the Maryland dataset.

YUPENN Dynamic Scenes dataset									
Features	HOF+ GIST	Chaos+ GIST	SOE		SFA	CSO (proposed)		BoSE (proposed)	
Classifier	NN	NN	NN	RF	SVM	STRF	STRF	SVM	SVM
Temporal τ	*all*	*all*	*all*	*all*	*all*	1	*all*	1	*all*
Beach	87	30	90	93	93	100	100	97	100
Elevator	87	47	90	100	97	97	100	97	97
Forest Fire	63	17	87	67	70	76	83	90	93
Fountain	43	3	50	43	57	40	47	80	87
Highway	47	23	73	70	93	67	73	93	100
Lightning Storm	63	37	90	77	87	93	93	97	97
Ocean	97	43	97	100	100	90	90	100	100
Railway	83	7	90	80	93	90	93	93	100
Rushing River	77	10	90	93	87	97	97	93	97
Sky-Clouds	87	47	93	83	93	100	100	93	97
Snowing	47	10	50	87	70	57	57	97	97
Street	77	17	87	90	97	97	97	100	100
Waterfall	47	10	47	63	73	80	76	83	83
Windmill Farm	53	17	73	83	87	93	93	100	100
Overall	**68**	**23**	**79**	**81**	**85**	**84**	**86**	**94**	**96**

Table 4.20: Recognition rates for the best performing descriptor and classifier combinations on the YUPENN dataset.

Since BoSE builds on very similar features as CSO+STRF, the improvement of 10% accuracy on both datasets indicates the importance of a local mid-level BoW representation and the dynamic pooling procedure. However, on Maryland, CSO+STRF performs better when just a small amount of temporal information is used (*i.e.*, when just $\tau = 1$ slice is used for prediction), and it allows for overall faster classification, since it does not require clustering (*i.e.*, K-Means) and coding (*i.e.*, LLC) steps. Nevertheless, both proposed approaches are able to incrementally classify the videos in an online manner.

Confusion tables for the proposed BoSE approach are shown in Table 4.21. It can be observed that most of the confusions are between visually similar scene classes. For example, on Maryland, Avalanche is confused with Landslide, or Iceberg Collapse with Volcanic Eruption. On YUPENN, confusions only occur between highly similar classes, *e.g.*, classes showing dynamic water textures, *i.e.*, Fountain, Rushing River and Waterfall. Some instances of these classes are illustrated in Figure 1.1(a).

Maryland "In-The-Wild"

Actual \ Classified	*Avalanche*	*Boiling Water*	*Chaotic Traffic*	*Forest Fire*	*Fountain*	*Iceberg Collapse*	*Landslide*	*Smooth Traffic*	*Tornado*	*Volcanic Eruption*	*Waterfall*	*Waves*	*Whirlpool*
Avalanche	**6**					1	1			1			1
Boiling Water		**7**		1		1				1			
Chaotic Traffic			**9**					1					
Forest Fire				**9**						1			
Fountain					**7**		1	1			1		
Iceberg Collapse	1					**6**				2		1	
Landslide	1					1	**6**	1					1
Smooth Traffic	1		2					**7**					
Tornado									**9**	1			
Volcanic Eruption	1					1				**8**			
Waterfall											**10**		
Waves					1							**9**	1
Whirlpool			2										**8**

YUPENN Dynamic Scenes dataset

Actual \ Classified	*Beach*	*Elevator*	*Forest Fire*	*Fountain*	*Highway*	*Lightning Storm*	*Ocean*	*Railway*	*Rushing River*	*Sky-Clouds*	*Snowing*	*Street*	*Waterfall*	*Windmill Farm*
Beach	**30**													
Elevator		**29**										1		
Forest Fire			**28**		1		1							
Fountain				**26**									3	1
Highway					**30**									
Lightning St.			1			**29**				1				
Ocean							**30**							
Railway								**30**						
Rushing River								1	**29**					
Sky-Clouds	1									**29**				
Snowing				1							**29**			
Street												**30**		
Waterfall			1	1					3				**25**	
Windmill Farm														**30**

Table 4.21: Confusion matrices for BoSE on both datasets. The columns show the predicted labels of the classifier, while the rows list the actual ground truth label.

Confidence interval for codebook based approaches

To investigate the degree of performance variation originating from the randomization in the feature coding step, Figures 4.9 (Maryland dataset) and 4.10 (YUPENN dataset) compare the proposed dynamic scene classification framework to the SFA algorithm of Theriault *et al.* [95], in several subsequent experiments with a fixed set of parameters. To produce those results the implementation kindly provided from the authors of [95] was used. The SFA parameters were set to the optimal ones for the Maryland dataset listed in [95]. The bar plots show the average classification performance for each class as well as the standard deviation, indicated by the confidence interval of the corresponding error bar. Compared to SFA, which gives an overall accuracy of 54.92% $\pm$ 1.60% on Maryland and 76.90% $\pm$ 1.81% on YUPENN, the proposed method obtains competitive recognition accuracies of 75.69% $\pm$ 1.17% and 96.10% $\pm$ 0.21%, respectively. This corresponds to a substantial absolute performance gain of 20.77% and 19.20% over the recently published SFA approach of Theriault *et al.* [95].

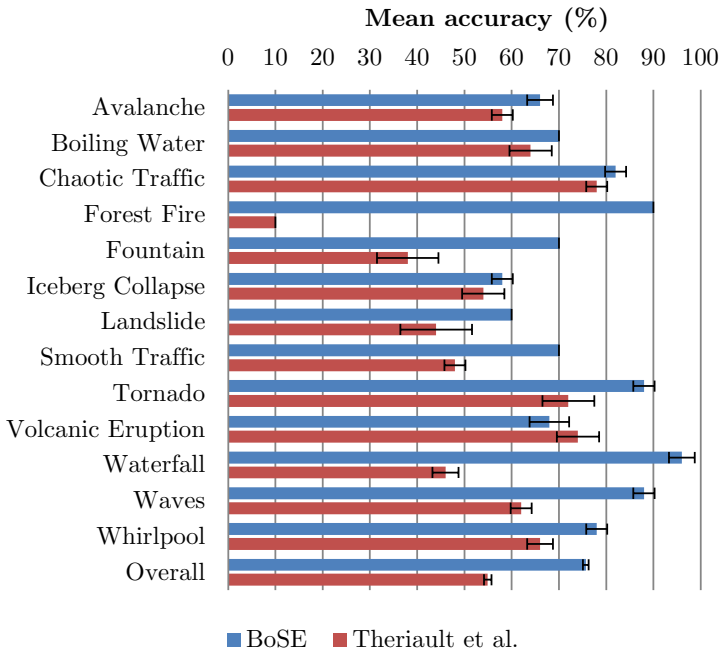


Figure 4.9: BoSE vs SFA [95] on the Maryland dataset. The class specific recognition accuracy as well as the average accuracy (Overall) is shown. The bar widths correspond to the average and the error bars indicate the respective standard deviations for each class.

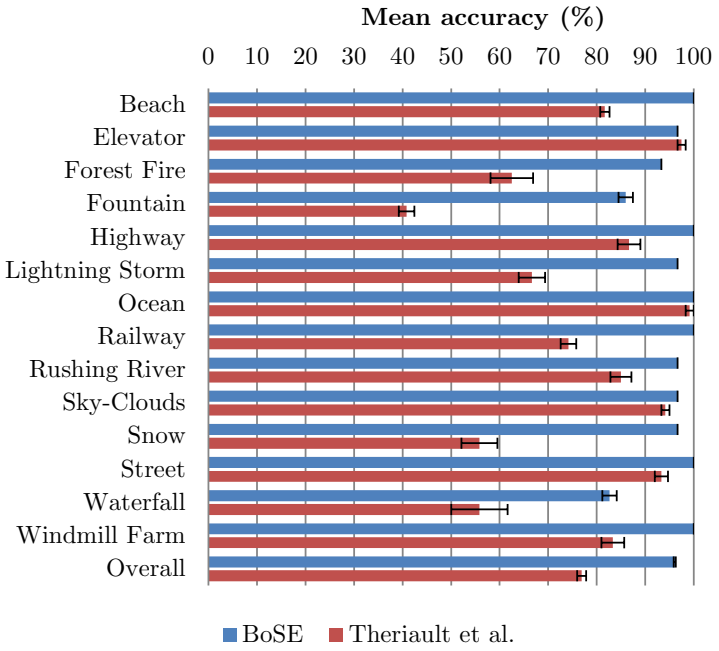


Figure 4.10: Comparison of the proposed BoSE method to the SFA approach [95] on the YUPENN dataset. Classification performance for each class and averaged over classes (Overall) is shown. The bar widths correspond to the mean of the recognition rates for 5 subsequent experiments and the error bars show the respective standard deviations.

4.8 Conclusion

This chapter has proposed BoSE, a generic BoW framework for dynamic scene recognition. Local spacetime orientation structure is extracted via application of multiscale, multiorientation filters and weighted aggregation of the resulting energy responses coupled with multiscale colour cues. Based on an evaluation of several popular feature coding methods, the local spacetime energies are projected into a mid-level representation by using a learned visual vocabulary. It has been shown, that the application of image stabilization leads to better performance on data confounded with camera motion, however, it degrades performance on scenes captured from a static camera. Finally, a novel spatiotemporal pooling strategy has been introduced, that aggregates the encoded spacetime features in a spatiotemporal pyramid representation, based on their dynamics in the frequency domain. The performance of the proposed framework has been verified in rigorous evaluations, where it has been shown that a carefully designed BoW model outperforms the state of the art significantly. The outstanding performance of the presented spacetime recognition framework for dynamic scene classification suggests that it could also be used in a variety of other areas, such as event retrieval, video indexing, or object and activity localization.

5

Summary and Outlook

In this chapter, a summary of the findings presented elsewhere in this thesis is given, followed by thoughts for future work. This thesis has addressed the problem of computational dynamic scene understanding. Recognition of dynamic scenes is relevant to several machine vision tasks such as the retrieval and ranking of video in search engines. Furthermore, scene understanding can be very useful when used as contextual information, for example, when reasoning about actions or activities in videos

This thesis provided several significant contributions, including two generic frameworks for visual recognition that were tuned specifically for dynamic scene classification. The proposed representations rely on oriented spacetime energy features, computed by applying oriented filters to the video. During spatiotemporal filtering, the filter scale has been matched to intrinsic scene dynamics, which allows for recognition that is robust to camera motion. A novel concept of temporal slicing has been introduced that temporally samples local descriptors from the video with a constant temporal spacing. Slicing allows for efficient, incremental online classification of video as well as treatment of temporal alignment as latent during classification.

The recognition approach presented in Chapter 3 is focused on fast, online processing of video. Temporal slices of the input sequences have been described by spatiotemporal aggregation of complementary histograms of efficiently extracted spatial and temporal filter responses, as well colour distributions. These complementary spacetime orientation (CSO) features were directly pooled in a vector representation. Next, a spacetime random forest (STRF) classifier was introduced that allows the complementary components of the CSO descriptor to be exploited during classification. In empirical evaluation, this approach has shown highly competitive recognition performance in nearly real time, with the ability of accurate recognition when only a very short amount of temporal information is processed.

The second dynamic scene classification framework presented in this thesis is based on bags of spacetime energies (BoSE). This approach models the local spatiotemporal orientation structure, as well as the local chromatic distribution by a sparse, over-complete spacetime dictionary. An extensive evaluation has shown the benefit of video stabilization and feature coding for dynamic scene classification. The outcome of these investigations is that a general spacetime BoW model is able to significantly outperform all previous approaches to dynamic scene recognition. Finally, the introduction of a novel spatiotemporal pooling method, which directly builds on the dynamics of the aggregated features, has further improved recognition accuracy, especially when large temporal diversity is present.

For ongoing work, it is planned to automatically learn the first-level feature descriptors from video for producing even more discriminative and robust representations. Following the current progress in the deep learning literature, highly discriminative filters should be learned to represent the input by an over-complete dictionary. Similar as the hand-crafted Gaussian derivative filters employed in this work, these learned filters should also be separable to allow for computational convenience. Furthermore, an important follow-on to the currently learned intermediate-level representations (*e.g.*, those documented in Chapter 4), as well as any learned primitive features, would be the development of an analysis that explains exactly what has been learned. In particular, from a scientific point of view it is essential to understand what properties of the world and/or image signal are being abstracted by these learned representations that enable the classifier to improve recognition in comparison to not making use of these representations.

A

Image stabilization with global motion estimation

Assuming constant brightness over adjacent frames, with I^t denoting the frame at time t, the transformation between frames I^t and I^{t-1} is expressed by

$$I^t(\mathbf{x} + \mathbf{u}^{t,t-1}) = I^{t-1}(\mathbf{x}), \tag{A.1}$$

where $I^t(\mathbf{x})$ is the image intensity at position $\mathbf{x}$ and time t, $\mathbf{x} = (x, y)^\top$ are the spatial coordinates and $\mathbf{u}^{t,t-1}$ is the computed motion field between the adjacent frames. The least squares solution to this problem is to find the motion field which minimizes the sum of squared differences (SSD)

$$\epsilon_{SSD}(\mathbf{u}^{t,t-1}) = \sum_{\mathbf{x}} [I^t(\mathbf{x} + \mathbf{u}^{t,t-1}) - I^{t-1}(\mathbf{x})]^2. \tag{A.2}$$

For small transformations, the brightness constancy assumption may be linearised by applying a Taylor series expansion and omitting higher order terms:

$$\begin{aligned}\epsilon_{SSD}(\mathbf{u}^{t,t-1}) &\approx \sum_{\mathbf{x}}[I^t(\mathbf{x}) + \nabla I^t(\mathbf{x})^\top \cdot (\mathbf{u}^{t,t-1})^\top - I^{t-1}(\mathbf{x})]^2 \\ &= \sum_{\mathbf{x}}[I_t^t(\mathbf{x}) + \nabla I^t(\mathbf{x})^\top \cdot (\mathbf{u}^{t,t-1})^\top]^2 \end{aligned} \tag{A.3}$$

with $\nabla I = (I_x, I_y)$ being the partial derivatives of the intensity function I with respect to the spatial coordinates $(x, y)^\top$, and I_t being the partial derivative of I with respect to the temporal coordinate t.

A.1 Translational motion model

The least squares problem (A.3) can be rewritten to a set of linear equations by setting the SSD to zero. For a translational motion model the estimate of the motion can be found by solving the following set of linear equations:

$$\mathbf{A}\mathbf{u} = \mathbf{b} \tag{A.4}$$

with

$$\mathbf{A} = \begin{pmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{pmatrix}, \mathbf{u} = \begin{pmatrix} u \\ v \end{pmatrix} \text{ and } \mathbf{b} = -\begin{pmatrix} \sum I_x I_t \\ \sum I_y I_t \end{pmatrix}. \tag{A.5}$$

A.2 Affine motion model

If the camera motion is approximated by an affine transformation

$$\begin{aligned} u(x, y) &= a_x + b_x x + c_x y \\ v(x, y) &= a_y + b_y x + c_y y, \end{aligned} \tag{A.6}$$

the motion is modelled by a local affine transformation in the image plane, consisting of rotation, dilation, shear and translation.

For an affine motion model the motion estimate can be found by solving the following

set of linear equations [6]:

$$\begin{bmatrix} \sum I_x^2 & \sum xI_x^2 & \sum yI_x^2 & \sum I_xI_y & \sum xI_xI_y & \sum yI_xI_y \\ \sum xI_x^2 & \sum x^2I_x^2 & \sum xyI_x^2 & \sum xI_xI_y & \sum x^2I_xI_y & \sum xyI_xI_y \\ \sum yI_x^2 & \sum xyI_x^2 & \sum y^2I_x^2 & \sum yI_xI_y & \sum xyI_xI_y & \sum y^2I_xI_y \\ \sum I_xI_y & \sum xI_xI_y & \sum yI_xI_y & \sum I_y^2 & \sum xI_y^2 & \sum yI_y^2 \\ \sum xI_xI_y & \sum x^2I_xI_y & \sum xyI_xI_y & \sum xI_y^2 & \sum x^2I_y^2 & \sum xyI_y^2 \\ \sum yI_xI_y & \sum xyI_xI_y & \sum y^2I_xI_y & \sum yI_y^2 & \sum xyI_y^2 & \sum y^2I_y^2 \end{bmatrix} \begin{bmatrix} a_x \\ b_x \\ c_x \\ a_y \\ b_y \\ c_y \end{bmatrix}$$

$$= - \begin{bmatrix} \sum I_xI_t \\ \sum xI_xI_t \\ \sum yI_xI_t \\ \sum I_yI_t \\ \sum xI_yI_t \\ \sum yI_yI_t \end{bmatrix}. \tag{A.7}$$

Image motion is estimated in a hierarchical alignment procedure as described in [5]. The motion is estimated within an image pyramid, starting at the coarsest resolution in order to recover large motions and ending at the finest resolution for small displacements, by applying the velocity estimates at each level.

For the translational motion model, the frame to warp is transformed by the estimated constants $(u, v)^\top_{t,t-1}$ at all coordinates $(x, y)^\top$. In case of the affine motion model, the coordinates are transformed according to the estimated affine parameters $a_x \dots c_y$. After applying the transformations to all frames of a slice, missing image regions due to the warping are trimmed to generate the final stabilized sequence. For the case of large estimated camera motion, the video is only trimmed to a minimum width or height of 100 pixels.

Before applying a transformation to the slices, a reliability measure for the flow estimate is computed. The reason is that video frames without sufficient gradient structure, for example, black frames in lightning sequences, would provide unreliable motion estimates. The reliability measure is calculated on the second moment matrix by considering

the minimum eigenvalue of the structure tensor

$$\mathbf{A} = \begin{pmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{pmatrix}, \tag{A.8}$$

where eigenvalues scale in proportion to the gradient energy. Note that, by definition, $\mathbf{A}$ has non-negative eigenvalues, *i.e.* it is symmetric positive semi-definite. By inspecting the eigenvalues of matrix $\mathbf{A}$, it is possible to find image regions with a lack of texture. Large eigenvalues of $\mathbf{A}$ indicate highly textured image frames. In the present work the minimum eigenvalue criterion of Shi and Tomasi [88] is used to discard structureless images. This distinction has been previously used for finding good features to track [88]. A motion estimate is incorporated in the proposed stabilization only if the minimum eigenvalues of adjacent frames are both above a threshold:

$$\min\left(\lambda_x^t, \lambda_x^{t-1}\right) > T, \tag{A.9}$$

where λ_x^t and λ_x^{t-1} are the smallest eigenvalues of frame I^t and I^{t-1}, respectively.

Bibliography

[1] Adelson, E. and Bergen, J. (1985). Spatiotemporal energy models for the perception of motion. *Journal of the Optical Society of America*, 2(2):284–299. 7, 75

[2] Amit, Y. and Geman, D. (1997). Shape quantization and recognition with randomized trees. *Neural Computation*, 9:1545–1588. 32

[3] Berg, A. C., Berg, T. L., and Malik, J. (2005). Shape matching and object recognition using low distortion correspondences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 10

[4] Bergen, J. R. (1991). Theories of visual texture perception. In Regan, D., editor, *Spatial Vision*, pages 114–133. Macmillan, London, UK. 7

[5] Bergen, J. R., Anandan, P., Hanna, K. J., and Hingorani, R. (1992a). Hierarchical model-based motion estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 68, 69, 117

[6] Bergen, J. R., Burt, P. J., Hingorani, R., and Peleg, S. (1992b). A three-frame algorithm for estimating two-component image motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(9):886–896. 117

[7] Boiman, O., Shechtman, E., and Irani, M. (2008). In defense of nearest-neighbor based image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 10

[8] Bosch, A., Zisserman, A., and Munoz, X. (2007). Image classification using random forests and ferns. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 32

[9] Bosch, A., Zisserman, A., and Muoz, X. (2008). Scene classification using a hybrid generative/discriminative approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 30(4):712–727. 10

[10] Boureau, Y., Bach, F., LeCun, Y., and Ponce, J. (2010). Learning mid-level features for recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 11, 56, 97

[11] Boureau, Y., Le Roux, N., Bach, F., Ponce, J., and LeCun, Y. (2011). Ask the locals: multi-way local pooling for image recognition. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 11, 56, 59

[12] Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32. 13, 32, 36, 40, 43

[13] Brown and Lowe (2003). Recognising panoramas. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 68

[14] Cao, L., Ji, R., Gao, Y., Yang, Y., and Tian, Q. (2012). Weakly supervised sparse coding with geometric consistency pooling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 11, 56, 59, 73

[15] Capel, D. and Zisserman, A. (1998). Automated mosaicing with super-resolution zoom. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 68

[16] Censi, A., Fusiello, A., and Roberto, V. (1999). Image stabilization by features tracking. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*. 68

[17] Chang, C.-C. and Lin, C.-J. (2011). Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27. 82

[18] Chang, H.-C., Lai, S.-H., and Lu, K.-R. (2004). A robust and efficient video stabilization algorithm. In *IEEE International Conference on Multimedia and Expo (ICME)*. 68

[19] Chatfield, K., Lempitsky, V., Vedaldi, A., and Zisserman, A. (2011). The devil is in the details: an evaluation of recent feature encoding methods. In *Proceedings of the British Machine Vision Conference (BMVC)*. 11, 52, 56, 57, 83, 85

[20] Chen, Q., Song, Z., Feris, R., Datta, A., Cao, L., Huang, Z., and Yan, S. (2013). Efficient maximum appearance search for large-scale object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 52

[21] Cinbis, R. G., Verbeek, J., and Schmid, C. (2013). Segmentation driven object detection with Fisher vectors. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 52

[22] Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297. 13, 53

[23] Criminisi, A. and Shotton, J., editors (2013). *Decision Forests for Computer Vision and Medical Image Analysis*. Springer Publishing Company, Incorporated. 13, 32

[24] Criminisi, A., Shotton, J., and Konukoglu, E. (2012). Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning. *Foundations and Trends in Computer Graphics and Computer Vision*, 7:81–227. 32

[25] Csurka, G., Dance, C., Fan, L., Willamowski, J., and Bray, C. (2004). Visual categorization with bags of keypoints. In *ECCV workshop on statistical learning in computer vision*. 10

[26] Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 11, 25

[27] Derpanis, K., Lecce, M., Daniilidis, K., and Wildes, R. P. (2012a). Dynamic scene understanding: The role of orientation features in space and time in scene classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2, 5, 6, 7, 9, 16, 24, 29, 30, 32, 37, 39, 50, 52, 61, 65, 71, 83, 106

[28] Derpanis, K., Sizintsev, M., Cannons, K., and Wildes, R. P. (2012b). Action spotting and recognition based on a spatiotemporal orientation analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 35(3):527–540. 7

[29] Derpanis, K. and Wildes, R. P. (2012). Spacetime texture representation and recognition based on a spatiotemporal orientation analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 34(6):1193–1205. 5, 7, 23

[30] Doretto, G., Chiuso, A., Wu, Y. N., and Soatto, S. (2003). Dynamic textures. *International Journal of Computer Vision (IJCV)*, 51(2):91–109. 6

[31] Engel, S., Zhang, X., and Wandell, B. (1997). Colour tuning in human visual cortex measured with functional magnetic resonance imaging. *Nature*, 388(6637):68–71. 18

[32] Fei-Fei, L. and Perona, P. (2005). A Bayesian hierarchical model for learning natural scene categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 1, 5, 10

[33] Feichtenhofer, C., Pinz, A., and Wildes, R. P. (2013). Spacetime forests with complementary features for dynamic scene recognition. In *Proceedings of the British Machine Vision Conference (BMVC)*. 16

[34] Feng, J., Ni, B., Tian, Q., and Yan, S. (2011). Geometric ℓ_p-norm feature pooling for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 11, 56, 59, 73

[35] Freeman, W. and Adelson, E. (1991). The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 13(9):891–906. 23, 24, 35

[36] Freund, Y. and Schapire, R. (1999). A short introduction to boosting. *Japanese Society For Artificial Intelligence*, 14(771-780):1612. 13

[37] Gorea, A., Papathomas, T. V., and Kovacs, I. (1993). Motion perception with spatiotemporally matched chromatic and achromatic information reveals a "slow" and a "fast" motion system. *Vision Research*, 33(17):2515 – 2534. 18

[38] Granlund, G. and Knutsson, H. (1995). *Signal Processing for Computer Vision*. Kluwer Academic Publishers Norwell, MA, USA. 7

[39] Grauman, K. and Darrell, T. (2005). The pyramid match kernel: Discriminative classification with sets of image features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 13, 54, 71

[40] Grossberg, S. and Huang, T. R. (2009). Artscene: A neural system for natural scene classification. *Journal of Vision*, 9(4):1–19. 37

[41] Itti, L., Koch, C., and Niebur, E. (1998). A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 20(11):1254 –1259. 18

[42] Jaakkola, T., Haussler, D., et al. (1999). Exploiting generative models in discriminative classifiers. *Advances in Neural Information Processing Systems*, 11:487–493. 56

[43] Jähne, B. (2005). *Digital Image Processing, Sixth Edition*. Springer, London, UK. 35

[44] Jain, M., Jegou, H., and Bouthemy, P. (2013). Better exploiting motion for better action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 68

[45] Jégou, H., Perronnin, F., Douze, M., Sánchez, J., Pérez, P., and Schmid, C. (2012). Aggregating local image descriptors into compact codes. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*. 58, 60

[46] Jia, Y., Huang, C., and Darrell, T. (2012). Beyond spatial pyramids: Receptive field learning for pooled image features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 11, 56, 59, 73

[47] Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning*. 10

[48] Juneja, M., Vedaldi, A., Jawahar, C. V., and Zisserman, A. (2013). Blocks that shout: Distinctive parts for scene classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 52, 58, 85

[49] Jurie, F. and Triggs, B. (2005). Creating efficient codebooks for visual recognition. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 10

[50] Kläser, A., Marszałek, M., and Schmid, C. (2008). A spatio-temporal descriptor based on 3d-gradients. In *Proceedings of the British Machine Vision Conference (BMVC)*. 11

[51] Koenderink, J. (1984). The structure of images. *Biological Cybernetics*, 50(8):363–370. 30, 60

[52] Laptev, I. (2005). On space-time interest points. *International Journal of Computer Vision (IJCV)*, 64(2-3):107–123. 11

[53] Laptev, I., Marszalek, M., Schmid, C., and Rozenfeld, B. (2008). Learning realistic human actions from movies. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 6

[54] Larlus, D., Jurie, F., et al. (2006). Latent mixture vocabularies for object categorization. In *Proceedings of the British Machine Vision Conference (BMVC)*. 10

[55] Lazebnik, S. and Raginsky, M. (2009). Supervised learning of quantizer codebooks by information loss minimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 31(7):1294–1309. 56

[56] Lazebnik, S., Schmid, C., and Ponce, J. (2005a). A maximum entropy framework for part-based texture and object recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 10

[57] Lazebnik, S., Schmid, C., and Ponce, J. (2005b). A sparse texture representation using local affine regions. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 27(8):1265–1278. 10

[58] Lazebnik, S., Schmid, C., and Ponce, J. (2006). Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 1, 5, 7, 10, 11, 13, 30, 50, 60, 71, 73

[59] Lepetit, V. and Fua, P. (2006). Keypoint Recognition using Randomized Trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 28(9):1465–1479. 32

[60] Leung, T. and Malik, J. (2001). Representing and recognizing the visual appearance of materials using three-dimensional textons. *International Journal of Computer Vision (IJCV)*, 43(1):29–44. 10

[61] Li, F. F., VanRullen, R., Koch, C., and Perona, P. (2002). Rapid natural scene categorization in the near absence of attention. *Proceedings of the National Academy of Sciences*, 99(14):9596–9601. 1

[62] Liu, J. and Shah, M. (2007). Scene modeling using co-clustering. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 5

[63] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 60(2):91–110. 11, 61

[64] Lucas, B. D., Kanade, T., et al. (1981). An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence*. 74

[65] M. Elfiky, N., González, J., and Roca, F. X. (2012). Compact and adaptive spatial pyramids for scene recognition. *Image and Vision Computing (IVC)*, 30(8):492–500. 5

[66] Mairal, J., Bach, F., Ponce, J., Sapiro, G., and Zisserman, A. (2008). Discriminative learned dictionaries for local image analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 56

[67] Marszalek, M., Laptev, I., and Schmid, C. (2009). Actions in context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 5, 6, 37, 106

[68] Matsushita, Y., Ofek, E., Ge, W., Tang, X., and Shum, H.-Y. (2006). Full-frame video stabilization with motion inpainting. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 28(7):1150–1163. 68

[69] Matsushita, Y., Ofek, E., Tang, X., and Shum, H.-Y. (2005). Full-frame video stabilization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 69

[70] Moosmann, F., Nowak, E., and Jurie, F. (2008). Randomized clustering forests for image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 30(9):1632–1646. 10

[71] Moosmann, F., Triggs, B., and Jurie, F. (2007). Fast discriminative visual codebooks using randomized clustering forests. In *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*. 32

[72] Muja, M. and Lowe, D. G. (2009). Fast approximate nearest neighbors with automatic algorithm configuration. In *Proceedings of the International Conference on Computer Vision Theory and Applications (VISAPP)*. 55, 81

[73] Oliva, A. and Torralba, A. (2001). Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision (IJCV)*, 42:145–175. 1, 5, 7, 11, 37, 106

[74] Oneata, D., Verbeek, J., and Schmid, C. (2013). Action and event recognition with Fisher vectors on a compact feature set. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 52

[75] Oppenheim, A. V., Schafer, R. W., Buck, J. R., et al. (1999). *Discrete-time signal processing*, volume 5. Prentice Hall, Upper Saddle River, New Jersey, USA. 24

[76] Papathomas, T. V., Gorea, A., and Julesz, B. (1991). Two carriers for motion perception: Color and luminance. *Vision Research*, 31(11):1883–1892. 18

[77] Park, D., Zitnick, C. L., Ramanan, D., and Dollár, P. (2013). Exploring weak stabilization for motion feature extraction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 68

[78] Perronnin, F. and Dance, C. (2007). Fisher kernels on visual vocabularies for image categorization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 11, 52, 56, 57

[79] Perronnin, F., Liu, Y., Sánchez, J., and Poirier, H. (2010a). Large-scale image retrieval with compressed Fisher vectors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 58

[80] Perronnin, F., Sánchez, J., and Mensink, T. (2010b). Improving the Fisher kernel for large-scale image classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 57, 58, 60

[81] Poppe, R. (2010). A survey on vision-based human action recognition. *Image and Vision Computing (IVC)*, 28(6):976–990. 6

[82] Potter, M. C. and Levy, E. I. (1969). Recognition memory for a rapid sequence of pictures. *Journal of experimental psychology*, 81(1):10. 1

[83] Rasiwasia, N. and Vasconcelos, N. (2008). Scene classification with low-dimensional semantic spaces and weak supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 5

[84] Rousselet, G. A., Thorpe, S. J., Fabre-Thorpe, M., et al. (2004). How parallel is visual processing in the ventral pathway? *Trends in cognitive sciences*, 8(8):363–370. 1

[85] Scovanner, P., Ali, S., and Shah, M. (2007). A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the International Conference on Multimedia*. 11

[86] Serre, T., Wolf, L., Bileschi, S., Riesenhuber, M., and Poggio, T. (2007). Robust object recognition with cortex-like mechanisms. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 29(3):411–426. 7

[87] Shabou, A. and LeBorgne, H. (2012). Locality-constrained and spatially regularized coding for scene categorization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 56

[88] Shi, J. and Tomasi, C. (1994). Good features to track. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 118

[89] Shotton, J., Johnson, M., and Cipolla, R. (2008). Semantic texton forests for image categorization and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 32

[90] Shroff, N., Turaga, P., and Chellappa, R. (2010). Moving vistas: Exploiting motion for describing scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2, 5, 6, 7, 9, 16, 29, 32, 37, 52, 65, 71, 83, 106

[91] Simoncelli, E. and Heeger, D. (1996). A model of neuronal responses in visual area MT. *Vision Research*, 38(8):743–761. 7

[92] Simonyan, K., Parkhi, O. M., Vedaldi, A., and Zisserman, A. (2013). Fisher vector faces in the wild. In *Proceedings of the British Machine Vision Conference (BMVC)*. 52

[93] Sivic, J. and Zisserman, A. (2003). Video google: A text retrieval approach to object matching in videos. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 10

[94] Szummer, M. and Picard, R. (1998). Indoor-outdoor image classification. In *IEEE International Workshop on Content-Based Access of Image and Video Database*. 5, 7

[95] Theriault, C., Thome, N., and Cord, M. (2013). Dynamic scene classification: Learning motion descriptors with slow features analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 7, 37, 39, 50, 71, 106, 110, 111

[96] Vailaya, A., Figueiredo, M. A. T., Jain, A., and Zhang, H.-J. (2001). Image classification for content-based indexing. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 10(1):117–130. 5

[97] van de Sande, K. E. A., Gevers, T., and Snoek, C. G. M. (2010). Evaluating color descriptors for object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 32(9):1582–1596. 29

[98] van Gemert, J. C., Veenman, C. J., Smeulders, A. W., and Geusebroek, J.-M. (2010). Visual word ambiguity. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 32(7):1271–1283. 56

[99] Varma, M. and Zisserman, A. (2003). Texture classification: Are filter banks necessary? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 10

[100] Vedaldi, A. and Fulkerson, B. (2008). VLFeat: An open and portable library of computer vision algorithms. `http://www.vlfeat.org/`. 55, 81

[101] Vedaldi, A. and Zisserman, A. (2012). Efficient additive kernels via explicit feature maps. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 34(3):480–492. 13, 60

[102] Vogel, J. and Schiele, B. (2007). Semantic modeling of natural scenes for content-based image retrieval. *International Journal of Computer Vision (IJCV)*, 72(3):133–157. 5

[103] Walther, D. and Koch, C. (2006). Modeling attention to salient proto-objects. *Neural Networks*, 19(9):1395–1407. 18

[104] Wang, J., Yang, J., Yu, K., Lv, F., Huang, T., and Gong, Y. (2010). Locality-constrained linear coding for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 11, 52, 56, 59, 60, 81, 97

[105] Watson, B. and Ahumada, A. (1983). A look at motion in the frequency domain. In *Proceedings of the Motion Workshop*. 23, 66

[106] Wildes, R. and Bergen, J. (2000). Qualitative spatiotemporal analysis using an oriented energy representation. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 11, 75

[107] Willems, G., Tuytelaars, T., and Van Gool, L. (2008). An efficient dense and scale-invariant spatio-temporal interest point detector. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 11

[108] Winn, J. and Shotton, J. (2006). The layout consistent random field for recognizing and segmenting partially occluded objects. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 32

[109] Wiskott, L. and Sejnowski, T. J. (2002). Slow feature analysis: Unsupervised learning of invariances. *Neural computation*, 14(4):715–770. 7

[110] Wyszecki, G. and Stiles, W. (2000). *Color Science, Second Edition*. John Wiley and Sons, New York, USA. 29, 65

[111] Yang, J., Yu, K., Gong, Y., and Huang, T. (2009). Linear spatial pyramid matching using sparse coding for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 11, 13, 56, 59, 96

[112] Yang, L., Jin, R., Sukthankar, R., and Jurie, F. (2008). Unifying discriminative visual codebook generation with classifier training for object category recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 10

[113] Yao, B., Khosla, A., and Fei-Fei, L. (2011). Combining randomization and discrimination for fine-grained image categorization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 11

[114] Yeffet, L. and Wolf, L. (2009). Local trinary patterns for human action recognition. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 11

[115] Yu, H., Li, M., Zhang, H.-J., and Feng, J. (2002). Color texture moments for content-based image retrieval. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*. 11

[116] Zhu, Z., Xu, G., Yang, Y., and Jin, J. S. (1998). Camera stabilization based on 2.5 D motion estimation and inertial motion filtering. In *IEEE International Conference on Intelligent Vehicles*. 68

[117] Zoghlami, I., Faugeras, O., Deriche, R., and antipolis Cedex, F.-S. (1997). Using geometric corners to build a 2D mosaic from a set of images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 68