Depth-adaptive methodologies for 3D image categorization

Submitted to Department of Electrical Engineering
in fulfillment of the requirements for the degree of

Doctor of Philosophy in Electrical Engineering & Electronics Research

by

Tsampikos Kounalakis

Department of Electrical Engineering
Brunel university London
June 2015

Depth-adaptive methodologies for 3D image categorization

by

Tsampikos Kounalakis

Submitted to Department of Electrical Engineering
on June 16,2015, in fulfillment of the requirements
for the degree of
Doctor of Philosophy in Electrical Engineering & Electronics Research

Abstract

Image classification is an active topic of computer vision research. This topic deals with the learning of patterns in order to allow efficient classification of visual information. However, most research efforts have focused on 2D image classification. In recent years, advances of 3D imaging enabled the development of applications and provided new research directions.

In this thesis, we present methodologies and techniques for image classification using 3D image data. We conducted our research focusing on the attributes and limitations of depth information regarding possible uses. This research led us to the development of depth feature extraction methodologies that contribute to the representation of images thus enhancing the recognition efficiency. We proposed a new classification algorithm that adapts to the need of image representations by implementing a scale-based decision that exploits discriminant parts of representations. Learning from the design of image representation methods, we introduced our own which describes each image by its depicting content providing more discriminative image representation. We also propose a dictionary learning method that exploits the relation of training features by assessing the similarity of features originating from similar context regions. Finally, we present our research on deep learning algorithms combined with data and techniques used in 3D imaging. Our novel methods provide state-of-the-art results, thus contributing to the research of 3D image classification.

Acknowledgments

First and foremost, I would like to thank my supervisor, Nikolaos V. Boulgouris, for his extraordinary support throughout my thesis. His guidance and encouragement all these years played an important role to the completion of this thesis, and my personal development as a researcher. I would also like to thank my second supervisor Georgios A. Triantafyllidis for his excellent advise and support during these past few years. Finally, I thank my family and friends for their support throughout all my studies.

Contents

List of Figures

List of Tables

List of Abbreviations

BTDI	Brunel Texture and Depth Image database
CNN	Convolutional Neural Network
SIFT	Scale Invariant Feature Transform
DoG	Difference-of-Gaussians
ShC	Shape context feature
ShCid	Shape-context with inner distance feature
EMK	Efficient Match Kernels
PCA	Principal Component Analysis
HMP	Hierarchical Match Pursuit
RGB-D	Red, Green, Blue and Depth images
RGB+D	Red, Green, Blue and Depth images
K-SVD	K-Singular Value Decomposition
NMF	Non-negative Matrix Factorization
GNMF	Graph-regularized Non-negative Matrix Factorization
VQ	Vector Quantization
SVD	Singular Value Decomposition
SPM	Spatial Pyramid Matching
BoW	Bag-of-Words
BoF	Bag-of-Features
SVM	Support Vectors Machine

DBN	Deep belief Network
RBM	Restricted Boltzmann Machine
CRBM	Convolutional Restricted Boltzmann Machine
GPU	Graphical Processing Units
ReLu	Rectifier Liner unit
SLAM	Simultaneous localization and mapping
BigBIRD	(Big) Berkeley Instance Recognition Dataset
LIDAR	LIght Detection And Ranging
ToF	Time-of-Flight camera
RGB	Red, Green and Blue images
HOG	Histogram of Oriented Gradients
SURF	Speeded Up Robust Features
ScSPM	Sparse Coding Spatial Pyramid Matching
KDES	Kernel DEscriptorS
SBSVM	Scale Based Support Vector Machine
LLC	Locality-constrained Linear Coding
ℓ-GP	ℓ-bounded Graph Partitioning
CAPM	Content-Adaptive Pyramid Matching
CA-GNMF	Context-Adaptive Graph regularized Nonnegative Matrix Factorization

List of publications

1. T. Kounalakis and N.V. Boulgouris, "Sparsity-based classification using texture and depth", *IEEE International Conference on Digital Signal Processing*, 2013

2. T. Kounalakis and N.V. Boulgouris, "Classification of 2D and 3D images using pyramid scale decision voting", *IEEE International Conference on Image Processing*, 2014

3. T. Kounalakis and N.V. Boulgouris, "Content-adaptive pyramid representation for three dimensional image classification", *UNPUBLISHED: in preparation of submission*, 2015

4. T. Kounalakis and N.V. Boulgouris, "Context-based Dictionary Training based on Depth for 3D Image Classification", *UNPUBLISHED: in preparation of submission*, 2015

Chapter 1

Introduction

Image classification is among the most popular research areas in computer vision. Image classification is the process where images are successfully categorized into classes according to their depicted contents. This topic is related to a variety of sub-topics which are differentiated regarding the content of the examined images. When the examined images depict objects then the task of identifying these objects, is known as object recognition. Further, when the image classification task deals with the identification of sceneries, it is termed scene understanding. Whatever the content of images, image classification approaches are very challenging because they are associated with large sets of realistic image data.

In recent years there has been great interest in image classification, because many current and evolving applications are based on the understanding of image content. The first implementations of image classification were applied in the field of robotics. Robotic vision, used on robots and autonomous vehicles, consists of multiple tasks of computer vision which are responsible for navigation, object recognition, human interaction, and others. Object recognition and scene understanding, allow robots to understand and interact with objects and their surrounding environment. Similar applications will eventually find their way to consumer uses such as computer systems and smartphones. The ability of these devices to recognize objects will provide new possibilities to many areas such as computer-human interaction, security, human assistance, or even providing new methods for advertising. Currently, such systems are

used for image retrieval, an application applied to many known web search engines.

Most proposed methods concern the categorization of 2D images, i.e, color or grayscale images depicting image texture. However, 3D image acquisition sensors have recently become commercially available as game console controllers and digital cameras. These sensors acquire 3D images depicting the information of depth. Depth information consists of range values representing the distance between the depicted content and acquisition sensor forming a depth image. The resulting depth image, also known as *depth map*, can capture shapes, silhouettes and surface textures from the depicted content. In this thesis, we study the attributes of depth information and research its effectiveness on modern image classification problems.

1.1 Current challenges in 3D image classification

3D image classification is a novel research topic exploiting depth information and its attributes in order to reach conclusions about the examined image. However, the integration of depth information in image classification systems comes with many challenges.

In the past, research on 3D imaging was limited by the lack of 3D image acquisition sensors. Those available were either too expensive and development for dedicated applications or bulky with calibration and synchronization issues. These reasons adversely affected research efforts and, as a result, the nature and uses of depth were not fully studied and exploited. The lack of reference methods was reversed only recently with the introduction of modern 3D acquisition sensors. At the beginning of our research only some methods had been published, including a 3D dataset [7] and the introduction of several feature extraction methods [8, 9]. However, these 3D image classification systems have inherent similarities with 2D methodologies [10, 11], that can not be applied to all 3D image data, as described in Section 2.1.3. As a result, these methods do not approach the task of 3D image classification effectively.

In this thesis, we report our research on 3D image data and its application for 3D image classification. We investigate the attributes and limitations of depth infor-

mation. Our goal is also to highlight the use of depth, not only as an image feature method but as an information that could unveil other image characteristics. The full exploitation of depth can lead to improved performance, thus contributing to 3D image classification efficiency. Furthermore, research areas such as object recognition and scene understanding can benefit from the integration of depth in image classification. In addition, proposed techniques for 3D image classification can be implemented in other areas such as 3D video classification. Methods that use depth for contextual image information will also be able to efficiently detect and describe objects/sceneries found in videos.

Improvements achieved though 3D research will bring widespread benefits to everyday systems used by the general public. 3D image classification will not only have an impact on robotics and autonomous vehicles but also in other applications. These applications may include real-world navigation, assistive devices for people with disabilities, future human-computer interaction and even advertising through object recognition.

1.2 Contributions

Our research includes a general examination of depth information which advances throughout our novel methodologies. Our proposed methodologies constitutes, gradually improve our knowledge in 3D image classification. We propose general improvements in every process of an image classification system, starting from data acquisition all the way to classification. We also implement novel deep learning methodologies that will contribute to future improvements in the field. All improvements are optimized exploiting attributes but also considering the limitations of depth information.

Our first great consideration was about benchmarks that correspond to realistic data, ensuring that our future methodologies can be applied to realistic scenarios. To this end, we introduce our novel **Brunel Texture and Depth Image database (BTDI)**, a dataset collected by a stereoscopic digital camera. This database focus on realistic data, easy expandability, ease of use while setting a solid benchmark for

multiple problems including image/object categorization, image segmentation and stereo correspondence.

We used the aforementioned dataset to benchmark our novel 3D image classification method that combined the individual representations of texture and depth. For the purpose of depth description, we introduce a novel **depth feature extraction methodology** that separates or combines different scales of depth values. This method allows the extraction of object shapes from depth maps, which are used for shape feature description. These shape features provide an image representation from depth features. The representation of depth is concatenated with the representation of texture, creating a common image representation. This method achieves top-performing results with small complexity and low computational costs compared to other competing methodologies. This work motivated us to further develop image representations and research their relations with classification algorithms.

Feature extraction methods can benefit by the use of encoding methods that provide more discriminant feature representations. However, feature encoding is gradually affected from dictionaries, i.e., matrices describing image features, used by the encoding methodologies. Current dictionary learning methodologies do not take into account the context from which each of their training features is collected. So, the relationship between features forming a dictionary is usually described only from their distance similarity in the feature space. We proposed a **dictionary learning constrain formulating the context relationships** of dictionary training features. This novel similarity measure considers as similar features only those coming from similar context. Our proposed method outperforms other competing dictionary training methodologies and achieves top-performing results in one 3D dataset.

Image representation is a key element in the overall performance of image classification systems. Most image classification systems use the representation in [2] which describes an image using spatial information. However, this representation method is fixed and does not take into consideration the content of each image. In this thesis, we propose a novel method that constructs a **content-adaptive pyramid representation** creating scales from regions of contents. The combination from encoded

features by content scales result in more efficient and discriminative image representations. Our novel representation outperforms the state-of-the-art methodologies in two 3D image datasets.

The process of classification is very important for image classification systems because the algorithm provides a decision defining the identity of each image. The performance of every classification algorithm is directly related to image representations which consist the data used for training and testing. To this end, we propose a novel classification which is more suitable for modern pyramid representations. Our suggested method studies the discriminative capabilities of each scale vector consisting a pyramid representation. We introduce a novel **scale-based classification** which favors individual scale representations. The scale decisions are processed by a majority voting, thus resulting to an improved classification decision. This methodology achieves state-of-the-art performance in two 3D datasets.

In the final part of this work, we deal with the development of **deep learning algorithms for 3D image classification**. This part of our thesis, studies the classification capabilities of Convolutional Neural Networks (CNNs) when they are used with 3D image data. In our experiments, we study the performance of CNN depth features when used individually or in combination with texture information. We also introduce a novel method of formulating known image representations as channels of input data to further enhance classification rates. The proposed methodology achieves improved results compared to CNN models that do not exploit the information provided from representations.

1.3 Thesis Outline

The following outline of this thesis describes in detail the course of our research as well as the resulting novel methodologies that contribute to the topic of 3D image classification.

In Chapter 2, we present a **literature review of current image classification methodologies**. We describe current challenges in image classification and the design

of efficient 2D and 3D methodologies. The chapter is divided regarding the individual processes comprising an image classification system including a separate section about deep learning.

Chapter 3, presents the process of **3D data acquisition** where we describe the compilation of our 3D imaging dataset. This chapter refers to most modern 3D datasets and focus on those used for image classification/object recognition. We also describe the design, acquisition process and other experimental tasks that can be examined using our dataset.

Our research on **shape feature extraction over depth maps** is presented in Chapter 4. This chapter introduce a feature extraction methodology using shape information extracted from depth maps to describe an image. We also introduce a 3D image classification system which performs a combination of the aforementioned features with those coming from texture using their individual sparse representations.

In our search for additional image discrimination, we propose a **context-based dictionary training methodology**, described in Chapter 5. Most dictionary learning methods are relating their training features with a distance similarity measure that does not take into account the context from which each feature was extracted. In this chapter, we formulate a context similarity measure that constrains the dictionary learning methodologies.

In Chapter 6 we propose **content-adaptive image representations**. Our proposed method captures content of images using image characteristics such as color, grayscale intensities and depth. The proposed multi-level representation, combined the encoded features of each image with regard to each image content.

In Chapter 7, we present a **scale-based classification algorithm**. The relation between image representation and a classification algorithm can affect the classification performance. Our proped classification methodology adapts to the needs of modern pyramid representations acquiring and combining the decisions of individual scales.

Chapter 8 presents our research on **deep learning algorithms** and their combination with 3D image data and techniques. We use depth information to enhance the

performance of the Convolutional Neural Networks (CNN) model. We also propose a novel formulation of image representation methodologies in order to enrich CNN input data with spatial or content information.

Conclusions and contributions are summarized in Chapter 9. The chapter also includes future extensions of the presented methods for further improvements on 3D image classification.

Chapter 2

Literature review of 2D and 3D image classification

As described in Chapter 1, the research topic of image classification is contributing to many applications. More recently, novel 3D image acquisition sensors were made publicly and commercially available. This fact justifies the need for 3D applications, such as image classification.

In this chapter, we describe the design of 2D and 3D image classification systems, their relations and advances over recent years. Image classification is a highly regarded and competitive research topic with newer methodologies presented each year. The task of image classification systems is to successfully recognize the depicted content and describe the image with the appropriate class label. This approach is also related to the research topic of object recognition. In fact, these topics can be considered the same when using images that do not need an object detection phase first. This means that when categorizing images depicting objects with little or no background clutter there is no separation between topics.

The design of modern image classification methodologies can be easily described by its comprising sub-tasks. These include *feature extraction*, *dictionary learning*, *feature encoding*, *image representation* and *classification*. These processes can be seen in Fig 2-1, depicting the flowchart of training/testing image classification systems.

The first task of feature extraction addresses the challenge of capturing content

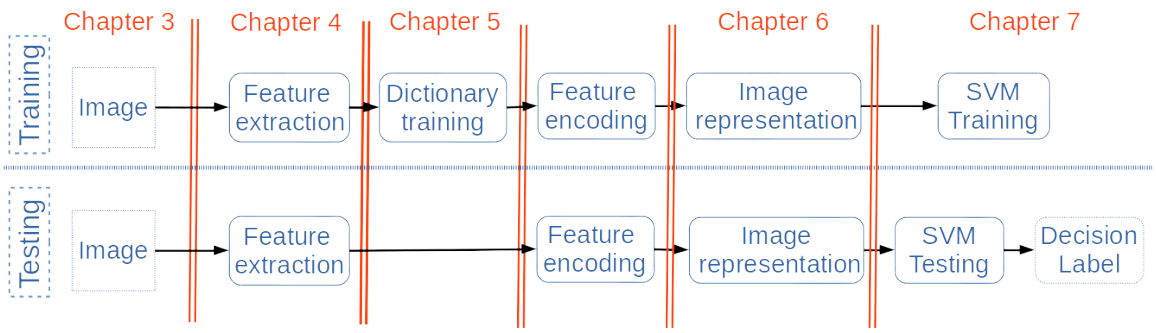


Figure 2-1: The comprising sub-tasks of image classification systems during training and testing phases.

information from raw image data. Feature descriptors, are vectors or matrices that describe small regions inside an image. These regions commonly describe values of rectangular pixel neighbourhoods. The type of image information captured by features regularly describes intensities of color or texture in a discriminant manner. The discrimination between features is a valuable asset for feature extraction methods aiming on the creation of similar features describing similar content. Feature extraction can also be considered as a first level image representation. But the significant number of extracted features from each examined image, makes the management and classification of individual features an inefficient process especially for large datasets.

Due to this fact, image classification research focuses on an efficient way of combining image features. This would create a discriminative representation of an image, favoring current classification algorithms. To this end, researchers developed a series of processes including dictionary learning, feature encoding, and image representation techniques.

Dictionary or codebook is the designation of a matrix computed in order to present patterns of image features. Dictionaries aim in a detailed description of all features that could be extracted from images found in a dataset. The computations of dictionaries help the discriminant representation of examined features. This is achieved by describing each examined feature as a combination of individual codewords consisting a dictionary.

The combinations are computed using feature encoding methodologies. The encoded vectors no longer represent feature information but the relation of features in

respect to the learned dictionary. This provides a more discriminant representation of features, favoring classification rates. Unfortunately, feature encoding can not be used directly for classification for similar reasons mentioned for feature vectors.

So, the forth task of image classification concludes with the use of image representations. These methods describe the way in which features or their encodings should be combined and described in a common representation vector. In their most basic form, image representations are used to capture the frequencies of features occurrences inside an image. In other applications of the same principle, feature encodings are combined in a histogram to create discriminant feature representation vectors suitable for classification. These methods treat features as an unordered set of data or describe features by capturing the spatial layout inside an image.

Finally, we conclude with the classification phase which is responsible for the decision regarding the identity of each image. The class label provided by the classification algorithm should correctly correspond to the label that describes the content of an examined image. A desired attribute of a classification algorithm is its ability to provide correct decisions in difficult classification problems such as large image datasets.

More recent methods redefine the conventional model of image categorization by introducing deep learning algorithms. Deep learning rejects the use of research-defined image features and representations. The reason is that their efficiency is limited by their architectures which cannot fully comprehend the nature of data. Instead, deep learning introduces algorithms able to compute self-taught features, image representations and even provide decisions. Current research [4, 5, 12] yields state-of-the art results in many 2D image classification systems.

The classic approach of image classification was developed for 2D images [2, 10, 11, 13–30], i.e, images with no information about depth. Most 3D image classification methodologies [7–9] are designed based on other known 2D systems [11, 16]. However, the existence of depth information, creates different challenges than the ones found in 2D systems. For instance, there is room for research in feature extraction methods over depth maps, where different attributes of images can be found. Different

types of features require different approaches in dictionary training and feature encoding that could be favored by the ability of depth to capture image content. Depth information may also lead to advanced image representations, that either combine texture and depth feature representations or use depth in order to combine features. All types of data and representations may require different classification approaches thus leading to better classification rates. Depth information could also contribute to the research of novel deep learning algorithms by enhancing input features, creating better representations and change the architecture of current models.

In the ensuing sections, we describe the current work of 3D image classifications [7–9] and the elements that are sharing with other known 2D methods [1, 2, 11, 16]. In Section 2.1, we present the advances of feature extraction for texture and depth. In Section 2.2, modern dictionary methods are analyzed. Then all feature encoding methods used in 3D image classification are described in Section 2.3. Current methods of image representation are described in Section 2.4 Section 2.5 presents the significance of classification algorithm in image classification systems. Finally, deep learning algorithms and their application to image classification are presented in Section 2.6.

2.1 Feature extraction methods in two and three dimensional imaging data

The first process of image classification methodologies is feature extraction, where several significant regions of an image are represented by corresponding vectors or matrices known as *descriptors*. By using these features, we hope to describe efficiently an image and its contents. To achieve this purpose, image features are using a variety of image information. The first attempts of feature design used color intensity and grayscale intensities in order to describe an image. However, most successful approaches use image gradients for detecting and creating image features.

An important trait of all feature extraction methods is their discrimination ca-

pabilities. Feature methodologies must not compute similar descriptors for unsimilar regions, thus guaranteeing the efficient description of image content. Other attributes of image features cover different aspects of use, such as dimensionality reduction. This means that an image and its contents, can be sufficiently represented by a few high-dimensional vectors thus reducing data space and computational cost. The feature vector form is important for linear operations such as transformations and combinations also making them suitable for use in modern classification algorithms.

Great improvements of recent years make the research topic of feature extraction very active and important. The discrimination capabilities of features, despite current efforts, are still far from perfect. Even modern methods, compute similar feature vectors representing different image regions and contents. It is understandable that in complex situations, i.e, large datasets with great magnitude of image data, feature extraction discrimination faults can occur. Improvements of feature design can be studied regarding feature dimensionality and computational cost, which are also an important factor for real-life applications.

2.1.1 Features on 2D data

Texture based feature extraction

The research of feature extraction, until recently, regarded only 2D images [1,31,32]. 2D implementations were most common due to the unavailability of 3D information, i.e, depth information of images. Current state-of-the-art feature extraction methods [1, 31, 32] use image gradient as data in order to describe image regions. In general, image gradients are very robust to scale and rotation changes and also capture efficiently image information. All feature extraction methods, use some kind of representation histogram of image gradients providing the desirable vector form and discriminative capabilities. The main differences between those methods is the computation of histogram representation and feature vector dimensionality. However, in this thesis we do not provide a comparison between feature methodologies because the purpose of our research does not include the comparison of individual features

and their characteristics. But we studied the general specifications of image features, in order to propose our own depth image features.

The most used image feature [2,10,11,13–30] is the Scale Invariant Feature Transform (SIFT) that was presented in [1]. SIFT feature provides invariance to rotation and scaling and is also partially invariant to image distortions and illumination conditions. This feature computes region representations using image gradients. In order to find interesting image regions, known as *keypoints*, it uses scale-space extrema of differences-of-Gaussians (DoG) within a difference-of-Gaussians pyramid. This process finds part of images that are not affected by the use of difference-of-Gaussians pyramid providing scale invariant keypoints. However, most applications in 2D image classification [2,10,11,13–30] disregard the keypoint detector. Instead, SIFT descriptors are computed over a dense grid representing the whole image into individual features. The computation of SIFT descriptors for each point of interest, i.e, keypoint or grid patch, is achieved using a local histogram of image gradients. Usually, a SIFT feature describes a four by four neighborhood of image pixels containing the gradient magnitudes and directions. Image gradients are quantized in eight directions, thus creating respectively eight gradient weighted histogram bins. This leads to a final representation feature of $4 \times 4 \times 8 = 128$ dimensional vector. An illustration of the computation of a SIFT descriptor is shown in Fig 2-2.

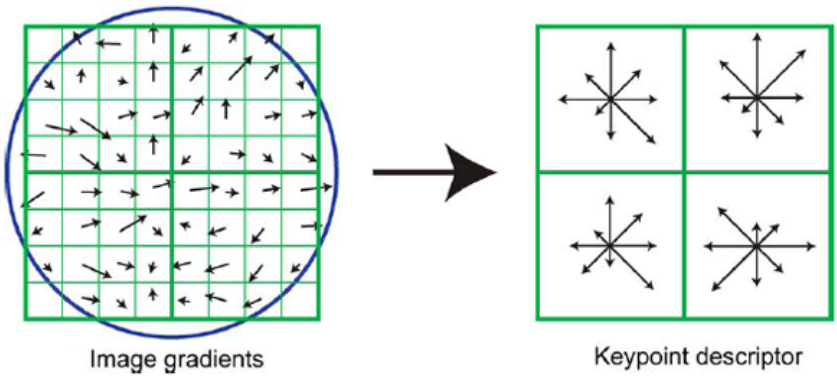


Figure 2-2: The creation of a SIFT descriptor as presented in [1].

In our work, we use the SIFT feature for texture description throughout our experiments due to its wide acceptance from researchers in image classification. The

examination of this well established feature, contributed to the design of our novel features over depth information. Our novel depth features are presented in Chapter 4.

Shape based feature extraction

Shapes can provide very discriminative features, because objects can be distinct due to their unique silhouettes. The use of shape is considered, in most cases, unreliable for some applications including image classification. The problem is found in classes describing unsimilar objects with similar shapes. For example, an image depicting an orange will provide a similar shape feature with an image depicting a ball. Also, shapes can not be accurately extracted without the need of user annotation, which is very time consuming for most image datasets. However, shape features are used in applications such as shape classification [33, 34].

A known methodology for shape representation is the Radon transform [35]. This transform is a special case of image projection operations. Radon transform computes the integral of a function along straight lines, thus describing shapes in a projection. It is very popular in image tomography (for reconstruction) but also found application in areas such as image segmentation, invariant image analysis [36], Arabic character recognition [37], filtering and restoring images [38, 39]. The advantages of Radon features are their low-dimensionality and invariance to shape deformations.

Shape context feature (ShC) [33] was firstly introduced in object recognition methodologies. The ShC computes histogram-based features describing shapes by creating a representation among individual shape points. Using the contours of an object, the ShC method extracts a number of reference points which correspond to image features. For each examined feature point, a descriptor is created using a log-polar histogram of limited range divided in multiple bins. The histogram describes the orientation of other points, found inside its range, in relation to the examined point. The description also includes the distance between each point regarding the examined one by using an intensity measure. The closest the distance between examined points, the greater the intensity value. As a result, each of the examined points is described by a feature histogram providing the relationship between points

of contour. The nature of this feature makes it invariant to rotation and scale by normalizing all distances with the mean distance. Furthermore, ShC provides some robustness to deformations and image noise.

In a more recent work, researchers in [34] introduced a feature based on the aforementioned Shape context descriptor. The presented Shape-context with inner distance feature (ShCid) uses a different distance measure creating a more discriminative representation among shapes. The authors proposed the use of the inner distance metric, which denotes the distance between two points of the same contour within the shape boundary. This means that the description of each examined point is relevant only to points that are close with respect to shape formation. This feature has all the advantages of the ShC feature improving its discrimination capabilities.

2.1.2 Features on 3D data

Feature extraction over depth image data was not an active research topic until recent advances in 3D acquisition technology. In previous years, range images, i.e, 3D images, were very hard to acquire due to the complexity of sensors. Furthermore, the need for such implementations was limited to specific applications such as robotics. However, new sensors are easier to operate and commercially available.

Following these advances, the research topic of 3D image classification, i.e, image classification using both texture and depth, is now an active research. As described earlier the first step of image classification is feature extraction. Some works [7–9] cover this topic for 3D image categorization. The challenges of feature extraction on depth information are mostly associated to the nature of depth maps, and all of the following methods [7–9] provide unique solutions.

Texture and shape features of [7]

The work in [7] introduces a 3D image dataset consisted of texture and depth images from the same objects. That paper also includes implementations showcasing the dataset in various 3D imaging problems, with object recognition being one of those

applications. As described earlier, image classification and object recognition are similar problems. When object detection techniques are not involved, then the two topics theoretically represent the same problem. So, authors proposed a multiple feature approach in order to achieve better classification rates.

Texture information was obtained using three types of features. SIFT features on a dense grid of eight by eight blocks, an implementation typically found in 2D image classification methods. Also, texton histograms [40] were extracted, in order to capture the texture of each object. Texton feature extraction has two parts including feature dictionary learning and feature creation. In dictionary learning, a training set of images pass through predefined gaussian filter providing *textons* which are then clustered by k-means algorithm. The texton feature is then created by convoluting with the filter and labeling. So, each filter corresponds with the texton closest in the feature-space. The final feature is created by a histogram describing the frequency of textons found in majority. A texton feature is a process of feature extraction, dictionary learning and feature representation. From the above, one could conclude that texton features can also be seen as a regional representation. For their experiments authors in [7] used the pre-built texton dictionary that was created using LabelMe dataset [41]. Their final feature used for texture was the three color histograms using the mean and standard deviation of each color channel.

The information of depth was captured using shape features combined with 3D bounding boxes. For shape features, the spin image features [42] were used, a feature usually found in 3D model classification. In order to capture those features authors in [7] use points of 3D image, also known as *3D point clouds*. They randomly select a set of 3D points from which spin features are computed. Spin features use 3D points that describe objects by its shape properties. This is done by "spinning" the object around the axis of an examined point thus collecting the contributions of neighboring points, a process somewhat similar to SIFT descriptors. The extracted spin features are used to compute Efficient Match Kernels (EMK) representation [16] using random Fourier sets. Authors enrich the extracted feature by using a 3D bounding cube of $3\times3\times3$ grid which captures that spatial information of features.

Finally, a simple solution is proposed for the combination of depth and texture features. The individual features create different image representations, one for each feature. The individual representation from texture and depth are combined to a common representation vector. It was experimentally shown that the combination of depth and texture representations provided the best classification results.

Depth kernel features of [10]

In [10], authors consider that similarity of features computed from histogram represented methods, such as [1, 31, 32], negatively affect the classification rates due to quantization errors of the binning process. This problem is addressed by using kernel functions in order to capture similarities between features. The advantages of their method are presented in a 2D image classification framework.

Motivated by that work, the method in [8] proposed several descriptors based on depth information that also use the kernel descriptors method. The introduced features represent size, shape and edges of depth maps.

Size of objects is described by measuring the distance between all points consisting a 3D point cloud describing an object. Then, a Gaussian kernel is created to describe these distances in relationship to a precomputed size feature dictionary. For shapes, authors in [8] consider two 3D point cloud shape features, such as kernel PCA features and spin kernel descriptors. For the first descriptor, kernel matrices are evaluated computing eigenvalues shown to capture efficiently shapes of objects. The spin kernels are computed using the aforementioned spin features [42], combined with the Kernel descriptors method [10]. Finally, authors proposed two edge-based descriptors over depth maps, a match kernel computed by gradients of depth maps, and a kernel from local binary descriptors.

For the final experimental setup, authors combined the image representations from texture features [10] and those introduced depth. The combination was similarly done to the aforementioned method of [7], by concatenating different feature image representations. Once again, it was experimentally shown that the combination of feature and depth provided superior results in comparison with the individual

representations.

Hierarchical Matching Pursuit in three dimensional image data

The 2D method of Hierarchical Matching Pursuit (HMP) [11] is an unsupervised feature learning which occurs in two layers. The first layer, describes features extracted from small image regions (typically 16 $\times$ 16 pixels). Each image, is separated in small image patches (typically 5 $\times$ 5 neighborhoods) containing pixel values. A first level dictionary is computed with K-Singular Value Decomposition (K-SVD) using the aforementioned patches. Each pixel in these image regions is represented by its sparse codes which are calculated using the Orthogonal Matching Pursuit (OMP) algorithm. Finally, each sparse code is combined by a spatially predefined max pooling process that concludes to the first level features/representations.

The second layer uses the extracted first layer features, sampling them into second class image features (which typically consisted of 4 $\times$ 4 first level features). The final image representation is created by OMP sparse coding, max pooling and spatial pyramid process applied on the second layer features.

In [11], this process is applied by only using the grayscale pixel values. However, authors in [9] used color and depth pixel values to compute the aforementioned features and their representations. These two image features provide individual image representations. Their combination is experimentally shown to improve the classification rates in the examined dataset.

2.1.3 Disadvantages of current 3D feature methodologies

All the aforementioned feature extraction methodologies [7–9] were designed for depth map information. In their experimental evaluation these methods use the RGB-D dataset [7] where all depth maps were acquired using a RGB-D camera. This camera has the advantage of capturing detailed range values for every pixel, thus creating detailed depth maps which allow the generation of 3D point clouds.

However, not all depth maps are produced using such sensors with high fidelity.

For instance, stereoscopic cameras can provide depth maps more robust to outdoor conditions in exchange to less detailed depth maps. Also, the creation of detailed 3D point clouds is more difficult using this acquisition sensor. This fact was not taken under consideration in the design of the aforementioned feature methods. As a result, some of these feature extraction methods are not applicable or efficient in all types of depth information.

The features constructed by 3D point clouds such as the spin images [7], and the shape and size feature of [8] are not applicable when multiple views are not available. The extraction of 3D point clouds need multiple views taken under a variety of angles. All views must be processed in order to find corresponding object parts forming the 3D cloud. So features extracted from 3D clouds are useful, but very difficult to implement on every 3D data.

Image gradients can provide texture information with great success thus contributing to a variety of feature extraction methods as presented in Chapter 2.1.1. Depth gradients used in [8] were inspired by a similar methodology in [10]. The depth gradients were computed over a RGB-D acquired depth map that depicts variations of depth in great detail. However, as described above, not all depth maps are of high detail. So, computing gradients,i.e, finding variations, is not so important, and does not provide information that could improve classification.

In the ensuing Chapter 4, we present our work in depth feature extraction methods based on shape, eliminating some disadvantages of the aforementioned methodologies.

2.2 Discriminative dictionary training methodologies

Dictionary learning (or codebook training) is the process of image classification systems where a matrix is computed using many sample image features. A dictionary is learned using a sample of random features extracted from all or a subset of images found in a dataset. All training features are processed by an algorithm which describes them in a smaller group of feature vectors called *codewords*. The collection of these codewords is called a dictionary (or codebook). The purpose of a dictionary is

to describe all possible features extracted from all images comprising a dataset using a subsampling process. The performance of image classification systems is favored by the computation of dictionaries. The use of dictionaries from encoding methods provides discriminative feature representations. More information about feature encoding methodologies is presented in the ensuing Section 2.3.

Dictionaries are used by the majority of 2D [2, 10, 11, 13–30] and 3D [7–9, 43, 44] image classification and object recognition systems. The popularity of dictionaries is based on their ability to create more discriminative feature representations. Dictionaries, when combined with an encoding method, create a feature representation for each extracted feature known as *feature encoding vector*. In most cases the dimensionality of the resulting vector is higher but more sparse. In this way, representing an image feature by encoded vectors leads to improved discrimination. When all encoded features are combined by an image representation method, as described in Section 2.4, the result leads to better classification results.

There are many methods for dictionary construction. The first method is based on the k-means algorithm, a very popular dictionary method among image classification methodologies [2, 7–11, 13–30, 43, 44]. Nevertheless, more recent works are oriented towards more elegant mathematical solutions for dictionary learning. So, methodologies such as K-Singular value decomposition (K-SVD) [45], Non-Negative Matrix Factorization (NMF) [46] and Graph-regularized Non-Negative Matrix Factorization (GNMF) [47, 48] were also considered for dictionary construction.

2.2.1 k-means

k-means algorithm is a classification algorithm which is able to create clusters of vectors, an ability also known as vector quantization (VQ). In general, this algorithm computes k number of clusters and groups each input data to the closest cluster center. Cluster centers are relocated computing the mean vector of all input data grouped by that cluster. This algorithm is used for many machine learning methodologies, but is also established for computing dictionaries in image classification. As mentioned earlier, a dictionary is computed from a random set of training feature vectors, here

denoted as $X = [x_1, x_2, \ldots, x_N] \in \mathbb{R}^{D \times N}$, where N is the number of features and D representing their dimensionality. These image features were extracted from all or a subset of the available images. k-means use the available feature vectors in order to compute the centroid vectors, i.e, cluster centers, describing the mean vector of a particular cluster. After an iterative process all examined features are grouped by cluster centroids. Each centroid is considered as a codeword consisting a dictionary. The computation of codewords is shown in the following equation:

$$\min_{U} \sum_{n=1}^{N} \min_{q=1,\ldots Q} \|x_n - u_q\|^2 \tag{2.1}$$

where $U = [u_{dk}] \in \mathbb{R}^{D \times Q}$ with Q the number of codewords, i.e, the number of centroids comprising the dictionary. Centroids consisting a dictionary are based on computations of distances between feature vectors in the feature space, thus achieving small computational cost. This method also creates dictionaries producing discriminative feature encodings, leading to improved classification rates. In some cases, further constraints such as those proposed in [13, 17] can be applied creating even more discriminative dictionaries.

k-means may be sufficient for many image classification methodologies, although the limit of dictionary efficiency is quickly reached with small margin for further improvement even with the application of constraints. This fact has resulted in further advances in dictionary learning research.

2.2.2 K-Singular value decomposition

The K-Singular Value Decomposition (K-SVD) [45] is a dictionary learning method similar to k-means algorithm. The algorithm tries to find the most optimal dictionary from the examined dictionary training features.

In k-means algorithm, each of the training feature vectors corresponds to its closest centroid. However, K-SVD method allows training features to be represented as a linear combination of the codewords found in the dictionary. The aforementioned optimization problem of k-means becomes:

$$\min_{U,Z}\{\|X - UZ\|_F^2\} \text{ subject to } \forall n, \|z_n\|_0 \leq T_0 \tag{2.2}$$

where $U = [u_{dk}] \in \mathbb{R}^{D\times K}$ is the dictionary and $Z = [z_{nk}]^T \in \mathbb{R}^{N\times K}$ is the representation matrix. N is the number of features in dictionary training set X. T_0 is the sparsity term which defines the nonzero entries of z_n which can be more than one but less than the number T_0. This term guarantees the sparsity which this method applies to the dictionary. This penalty term can be expanded reformulating the problem as:

$$\|X - UZ\|_F^2 = \|X - \sum_{j=1}^{K} u_j z_T^j\|_F^2 = \|(X - \sum_{j\neq k} u_j z_T^j) - u_k z_T^k\|_F^2 = \|E_k - u_k z_T^k\|_F^2 \tag{2.3}$$

E_k stands for the error for all N examples when the k_{th} example is removed. The multiplication of UZ is decomposed to K rank -1 matrices that can be solved using Singular Value Decomposition (SVD). The error E_q would also minimize using SVD, however eq 2.3 does not enforce sparsity.

By using ω_k we describe the indices pointing to z_n examples that use the atom u_k.

$$\omega_k = \{n\|1 \leq n \leq K, z_T^k(n) \neq 0\} \tag{2.4}$$

Ω_k is a $N \times |\omega_k|$ matrix where $(\omega(i), i)_{th}$ entries are ones, and zeros elsewhere. This matrix can sufficiently describe sparsity and can be applied to eq 2.3.

$$\|E_k\Omega_k - u_k z_T^k \Omega_k\|_F^2 = \|E_k^R - u_k z_R^k\|_F^2 \tag{2.5}$$

The dictionary and error E_k^R can be successfully decomposed by SVD in respect to sparsity. As a result, the optimal sparse dictionary is created by the sparse representation of the examined training feature vectors.

The algorithm uses an iterative process in which initially the coefficient matrix of each training feature vector is computed keeping the dictionary fixed. The second step uses the coefficient matrices to update the dictionary. Columns of the dictionary are individually updated, minimizing the error, as in k-means algorithm. This can

be achieved by K-Singular Value Decompositions, factorizing dictionary columns to coefficient vectors, which also provide the name of this dictionary learning method.

As in k-means, the output of this method is a matrix representing all examined training features, although K-SVD can achieve better feature encoding, thus yielding better image classification results. The codewords no longer present a "mean" from a group of features, but a linear combination creating more discriminative codewords comprising the dictionary. Another benefit of this algorithm comes from the computation process of dictionaries. The simultaneous update of dictionary and coefficient matrices leads to rapid convergence to the optimal sparse dictionary. However, limitations are found in the computation of the dictionary which favors sparsity, making this method non-optimal for all feature types.

2.2.3 Non-Negative Matrix Factorization

Non-Negative Matrix Factorization (NMF) [49] is a matrix factorization algorithm. NMF can compute two non-negative matrices which are the linear combination of a non-negative input matrix . For dictionary training feature set $X = [x_1, x_2, \ldots, x_N] \in \mathbb{R}^{D \times N}$, where N is the number of D-dimensional feature vectors, two non-negative matrices are computed. The resultant non-negative matrices U and Z, denoting the dictionary and representations respectively, provide a fine approximation X

$$X \approx UZ^T \tag{2.6}$$

where $U = [u_{dk}] \in \mathbb{R}^{D \times K}$ and $Z = [z_{nk}] \in \mathbb{R}^{N \times K}$, with K being the number of codewords.

Compared to other matrix factorization techniques such as Vectors Quantization (VQ), Principal Component Analysis (PCA) and Singular Value Decomposition (SVD), NMF is the only method that permits additive combinations of its factorized matrices. All other methods also permit subtractive combinations. As a result, all the above matrix factorization techniques provide different approximation results from the same data. NMF achieves great results in applications such as face recogni-

tion [50] and document clustering [51] outperforming other competing methods such as SVD factorization.

The performance of NMF opposed to other factorization methods made it a suitable candidate for dictionary learning in image classification. NMF is trained through an iterative process presented in [52], which tries to minimize a cost function. The cost function is the square of Euclidean distance between the matrices presenting the initial data. In [52], it is proven that NMF will reach the local minima of the cost functions thus providing better dictionaries.

Despite the NMF's advantages over other dictionary learning methodologies, this algorithm will only work with non-negative features. This means that not all available features are compatible. However, most modern image features are consisted of non-negative values.

2.2.4 Graph-regularized Non-Negative Matrix Factorization

As described earlier, the NMF methodologies provide top performance in many topics and also became popular for dictionary learning. Nontheless, this method has limitations which have become a point of interest for researchers. In [47], authors proposed the creation of a constraint that describes the relationship between dictionary training feature data. The proposed method is called Graph-regularized Non-Negative Matrix Factorization (GNMF).

The conventional NMF fits the training data in the Euclidean space thus failing to discriminate them in the feature space. Though in case of two features are close in the feature space then their representations should also be close. In GNMF, the relationship between feature data is presented by a Laplacian graph L which consists of the pairwise distances between all examined features in matrix W and a diagonal matrix D. The distance matrix W is sparse using only a number k closest features, and the construction of D is defined as $d_{ii} = \sum_{j=1}^{N} w_{ij}$. The final Laplacian matrix can be described as:

$$L = W - D \tag{2.7}$$

The computed Laplacian regulation graph can be applied on an update rule as a weighting matrix. The update rule tries to minimize the representations of features which are related to the computed weight matrix.

$$O = \min_{U,Z} \|X - UZ^T\| + \lambda Tr(Z^T L Z) \tag{2.8}$$

By minimizing the representations Z the dictionary is trained by keeping the representations fixed and optimizing the dictionary U. The regularization parameter in eq 2.8 can be further expanded:

$$Tr(Z^T L Z) = Tr(Z^T D Z) - Tr(Z^T W Z) \tag{2.9}$$

By using this constraint, GNMF has more discriminative power that the conventional NMF. The training feature data can be favored by describing the relationships between features from different classes. Thus, the discriminant capabilities of GNMF can improve feature encoding resulting in improved classification results.

However, the greatest advantage of this method is its ability to describe relationships between dictionary training features. The formulation of the regularization graph is not limited to relationships in the feature space, but it can also represent class label information [48] and other structures.

In Chapter 5, we present our work based on this dictionary learning method. We have developed the formulation of a constraint, describing similarities of context from which the training features were extracted.

2.2.5 Disadvantages of current dictionary methodologies

A major parameter regarding the performance of an image classification system is the combination between image feature type, dictionary learning and feature encoding methodologies.

As described earlier, k-means algorithm is one of the most used dictionary learning algorithms in 2D and 3D image classification. However, this does not rely on classification efficiency provided by the computed dictionary. The nature of k-means

algorithm makes it more compatible with a variety of feature types and encoding methods. As a result, k-means is not the optimal solution but a good compromise between performance, computational cost and an exhaustive optimization process.

Multiple examples of this problem are shown in the experimental section of Chapter 5. We present a comparison between k-mean, K-SVD, NMF, GNMF and our proposed dictionary training method applied to multiple features and different feature encodings.

2.3 Algorithms used for feature encoding

Feature encoding is the intermediate process between feature extraction and image representation. This process is found in most image classification systems [2, 7–11, 13–30]. The purpose of this process is to represent each extracted image feature in relation to a precomputed dictionary.

The dictionary, theoretically, is a general description of all possible features found inside a database, as described in Section 2.2. Its purpose is to represent each feature as a combination of codewords found in the dictionary. The encoded vector contains a number of coefficients corresponding to each codeword creating an multidimensional vector for each image feature. The encoded feature vectors usually have bigger dimensionality than the original feature vector. Nonetheless, the performance does not deteriorate due to the increased computational cost of a larger vector. This relies on the fact that the description provided by an encoding feature vector is far more discriminative than the feature itself. Therefore, the overall performance of image classification systems is significantly improved.

After the feature encoding process, all encoded feature vectors of an image are combined by an image representation technique which provides the data for classification. Feature encoding is in fact a special case of image representation methods. More details for image representations are presented in the ensuing Section 2.4. The research topic of feature encoding is very active constantly improving the recognition rates in various datasets. In this section, we describe the most used feature encoding

methods in 3D image classification systems [17].

2.3.1 Vector quantization

As we described in Section 2.2.1, k-means is a vector quantization method that is able to compute dictionaries. However, the dictionary training problem can be reformulated for feature encoding. Vector quantization (VQ) [2] is applied to feature encoding by keeping dictionary U fixed and solving for the encoding Z. This is achieved by reformulating the optimization problem into a matrix factorization problem.

$$\min_{U,V} \sum_{n=1}^{N} \|x_n - Uz_n\|^2 \text{ subject to } Card(z_n) = 1, |z_n| = 1, z_n \geq 0, \forall n \tag{2.10}$$

where $U = [u_{dk}] \in \mathbb{R}^{D \times K}$ and $Z = [z_{nk}]^T \in \mathbb{R}^{N \times K}$, K being the number of codewords. The cardinality constraint $Card(z_n) = 1$ means that one element of z_n is nonzero and defines all elements as nonnegative. So, each of the examined features, is represented as a linear combination of the dictionary providing an encoded feature. These encoded features contain one non-zero element indicating the codeword which corresponds to the quantization class of a feature vector. The non-zero element can be also found when searching for the closest codeword corresponding to the examined feature in the feature space. The vector quantization codes are easy to compute but generally produce great reconstruction errors when compared to other feature encoding methodologies.

2.3.2 Sparse coding

Authors in [17], proposed an improvement on feature encoding based on vector quantization codes. This is achieved by introducing a constraint, which relaxes the coefficient regularization which is computed using the l_1 norm [53]. The coefficients of the examined feature are presented by several non zero elements forming a sparse code.

$$\min_{U,V} \sum_{n=1}^{N} \|x_n - Uz_n\|^2 + \lambda |z_n| \text{ subject to } \|z_n\| \leq 1, \forall k = 1, 2, ..., K \tag{2.11}$$

Sparse coding can be computed using the feature-sign search algorithm [49] constraining the optimization problem. In [17], sparse feature encoded vectors are combined using the spatial pyramid matching image representation (SPM) [2]. For each scale vector consisting the spatial pyramid, a max pooling function was applied. Authors have experimentally shown that only positive coefficients are beneficial to image representation. This method can be also used for dictionary training using a constrained k-means.

This encoding method achieves smaller reconstruction errors than vector quantization, thus achieving better representations. Also, sparsity is able to capture the salient patterns of local features from each image. The method achieved top-performing results for many image classification datasets.

2.3.3 Disadvantages of feature encoding

Encoded vectors have higher dimensionality than the original feature, because they describe the relation between an image feature and each codeword consisting a dictionary. Nevertheless, regardless their higher dimensionality, encoded feature vectors provide more efficient representations than the image features which they describe.

The performance of these methods have a direct relation to the type of dictionary learning techniques used for its construction. As described in Section 2.2.5, the use of k-means is applied as a solution to this problem. However, the optimization for the best performing combination is achieved through intense research, as shown in Chapter 5. Feature encoding is often a computational intense process due to its complex formulation. This fact makes certain methods unsuitable for real-time applications, but more modern methods such as [13] are aiming towards computational efficiency.

2.4 The most used image representation methodologies

As described in previous sections, each image is presented by a number of individual features. But the direct use of features as training/testing data for classification will provide insignificant results. This is caused by the discriminatory issues as well as the magnitude of features. Each image commonly produces a large number of features that cannot be directly applied to conventional classification algorithms, especially for large datasets. Image features also have discrimination problems, i.e, features which will describe unsimilar image content with similar features. This is also the case even when using more discriminative feature encodings.

Therefore, researchers resorted to combinations between features, or their encodings, in order to provide one representation for each image thus making the classification more efficient. The representation of features can be achieved by several methods [2,16,54]. The output image representation vectors are then used as data for classification algorithms. These methods are *similarity functions* representing data extracted from an image, often called as *kernel functions*.

As described in Section 2.3, the encoding process is a special case of feature representation which helps the discrimination of image representations. Further improvements [13,16,17], proposed the encoding process of features before their combination by a representation methodology. As a result, each image representation is consisted of feature encoding vectors providing even more discriminative image representations.

The conception of these methods is very challenging, however image representation techniques [2,16,54] yield great improvements for image classification and are widely-used in many 2D or 3D systems [2, 7–11, 13–30]. The progress of these methods is described in the ensuing sections.

2.4.1 Bag-of-words(BoW)

The Bag-of-words (BoW) [54], or Bag-of-features (BoF), was one of the most popular representations for image classification and object recognition. BoW model represents the features of an examined image as a representation histogram. The number of bins consisting the histogram is equal to the number of codewords found in the dictionary. In its most basic form, the histogram describes the occurrence frequency of feature vectors being similar to a particular codeword, i.e, closest in the feature space. This model is very simple and computationally efficient but not very discriminative.

A more advanced approach of the BoW model, constructs a histogram of encoded feature coefficients using a *pooling* method. This method collects all the encoded vectors of an examined image and computes the mean value of each coefficient, a method known as *average pooling*. Eq 2.12 describes the average pooling function computing an image representation s from feature encodings z_n, $n = [1, ..., N]$ where N is the number of extracted features.

$$\mathbf{s} = mean[|\mathbf{z}^1|, |\mathbf{z}^2|, \ldots, |\mathbf{z}^N|] \tag{2.12}$$

A similar method creates image representations of max values of encoded vectors resulting to a histogram of max values, a method known as *max pooling*.

$$\mathbf{s} = \max[|\mathbf{z}^1|, |\mathbf{z}^2|, \ldots, |\mathbf{z}^N|] \tag{2.13}$$

The advanced model of BoW, outperforms the more simplistic approach. But, because of dictionary learning and feature encoding processes, it requires greater computational resources. However, the greatest limitation of BoW model is the representation of features in an orderless form.

2.4.2 Spatial pyramid matching(SPM)

The spatial layout of image features was completely disregarded by the BoW model, which treats features with no particular order. However, spatial layout can be very

useful when describing objects or image content. The ideal image representation should capture the content of each examined image and represent objects, scenes and its comprising parts. However, this is a rather difficult process because the content can be found in various regions of an image. We propose a solution to this problem in Chapter 6. So, other methods were proposed exploiting the available spatial information.

A first implementation of spatial information in image representation was introduced in [55]. This method creates a pyramid of multiple resolution histogram over unordered feature sets extracted from an image. In each resolution, a coarse grid is placed over the feature space creating a weighted match process for image features. Features that belong to individual sets, are considered to match when found in the same grid cell.

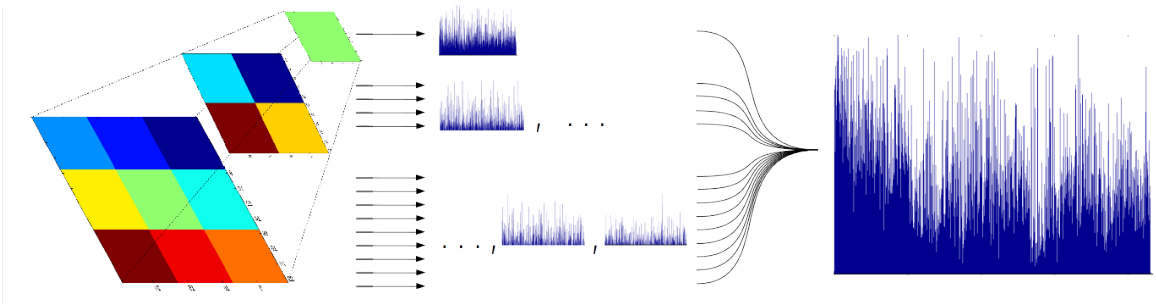


Figure 2-3: The spatial pyramid presented in [2].

In [2], the approach of incorporating spatial information into representation goes one step further. Using the proposed multi-resolution histogram approach of [55], this method represents each image into individual representation histograms, unlike the matches used in [55]. In detail, this method proposes a pyramid construction of multiple levels that subdivide the image in multiple rectangular regions. The regions are denoted as *scales* or *spatial bins* and create a BoW representation histogram describing each particular region. The scale representation histograms are concatenated in a final image representation vector. The method can also be considered as a special case of BoW collecting content information in a predefined spatial manner. This method is called Spatial Pyramid Matching (SPM) and is the most popular im-

age representation for image classification, achieving state-of-the-art results in many datasets . The architecture of pyramids is described by $l \times l$ scales in L different pyramid levels where $l = 1, 2, \ldots, L$.

$$\kappa(\mathbf{p}) = \sum_{l=1}^{L} \sum_{r=1}^{l^2} \kappa(\mathbf{s}_{lr}) \tag{2.14}$$

$\mathbf{s}_{lr}$ here denotes the scale representation vector ,i.e, the pyramid scale, on the l-th pyramid level and r-th rectangular region. At level $l = 0$ all features are represented from a single histogram, as a result SPM produces the same representation vector as the BoW model.

The SPM method was further improved by the methods presented in [13, 17]. These methods implement feature encoding and pooling processes creating more discriminative scale histograms thus leading to improved classification results. These methods are currently highly regarded and are often referenced in image classification methods.

2.4.3 Efficient Match kernels(EMK)

The work in [16] presents an image representation sharing similarities with feature encoding methodologies. Authors claim that BoW model is too simplistic and can not really represent the similarities between local features. Therefore, they propose a representation that captures similarities of features inside the feature space.

This is achieved by projecting feature data from each examined image to a dictionary learned using the K-SVD method. The low dimensionality projection coefficients, i.e, the encoded feature vectors $\mathbf{z}_n$ are calculated as:

$$\mathbf{z}_n = (U^T U)^{-1}(U^T \mathbf{x}_n) \tag{2.15}$$

The encoded feature vectors provide the following local kernel:

$$k(x, y) = [U\mathbf{z}_x]^T [U\mathbf{z}_y] = \mathbf{k}_U(x)^T K_{UU}^{-1} \mathbf{k}_U(y) \tag{2.16}$$

where $\{\mathbf{k}_U\}_i = k(\mathbf{x}_n, \mathbf{u}_i)$ is a $W \times 1$ vector and $\{k_{UU}\}_{ij} = k(\mathbf{u}_i, \mathbf{u}_j)$ a $d \times d$ matrix. For $K_{UU}^{-1} = G^T G$ forming a local feature map:

$$\phi(\mathbf{x}_n) = G\mathbf{k}_U(\mathbf{x}_n) \tag{2.17}$$

the final full feature map is $\phi(X) = \frac{1}{|X|} G[\sum_{\mathbf{x}_n \in X} \mathbf{k}_U(\mathbf{x}_n)]$. Implementing the proposed encoding on the SPM representation of eq 2.14, every scale vector is presented:

$$\mathbf{s}_{lr} = [|\phi(\mathbf{x}_{lr}^1)|, |\phi(\mathbf{x}_{lr}^2)|, \ldots, |\phi(\mathbf{x}_{lr}^{N_{lr}})|] \tag{2.18}$$

So, the resulting representation vector can be perceived as a combination of feature encoding and image representation methods. This representation method results to a histogram comprised by encoded vectors, like the advanced model of BoW method. This method achieves top-performing results when combined with a SPM pyramid image representation.

2.4.4 Disadvantages of image representation methods

Image representation methods have been successfully applied to many image classification systems. However, these methods also have disadvantages and limitations that have not been sufficiently studied.

All image representations result to a long vector that is complementing the use and performance of classification algorithms. Even more complex representations such as pyramid representations are concatenating their individual scales, i.e, regional representations, to produce a vector suitable for classification. Although, there is no guarantee about the discriminatory representational value of each individual scale vector. So scales with small discriminatory capabilities that adversely affect categorization are still used in the representation of an image. Another fact is that scales are collected in a predefined spatial manner thus capturing the spatial layout of an image. This method can provide great results when all examined images depict an object or the main content in the center of the image with little or no background clutter. This makes pyramid representations somewhat restrained when used in realistic

data, because the content of images is not always found under the aforementioned conditions.

In Chapter 7, we studied the success of current image representations, providing solutions to some of the aforementioned problems.

2.5 Classification algorithms of image classification systems

Classification is the final stage of an image classification system providing the decision for the identity of each examined image. The task for every classification algorithm is to make a correct decision for unknown input data. In the case of image classification, image features or image representations consist the data used for classification.

Classification algorithms are categorized by three types of learning, regarding the training process. The first category, is the unsupervised learning in which classification algorithms have no knowledge about the labels of data, i.e, the class from which each image is derived. In most cases, these algorithms are computationally efficient but do not provide great classification rates for large collection of data. The second category is the semi-supervised learning where only few labeled data are available from a large set of unlabeled data. This method achieves better classification rates than unsupervised methods benefiting from the extra data label information. The last type is supervised learning where every data is labeled in each class. The labels of data are available for use only during the training phase of the system. So, the classification algorithm is able to create class-specific attributes. Testing data labels are unknown to the classification algorithm, and are only used for validation purposes. In general, this kind of learning is the most successful for classifying large datasets of labeled data with the trade of higher computational demands.

In image classification, the majority of systems use supervised learning for their classification stage providing decisions for every image. Image datasets are usually very large, containing multiple images organized in several classes.

The classification is separated in two phases. The first phase is the training, where a number of images is exclusively used for this process. The training data are consisted of the representation vectors describing training images, and their respective class labels. The training set must be strictly randomly selected in order to ensure that data are not deliberately selected to present better performance. The training set will allow the classifier to learn patterns of image representation data. The training method of each classification algorithm differs in execution. But in most cases the methodology includes an iterative process that allows the algorithm to learn and minimize the training error.

The testing process uses the rest of the images that were not selected by the training process. This creates a testing set from image features or image representations and their identities, that are not known to the classification algorithm. The class labels of each testing image is not used by the classification algorithm during testing but they will be later used for validation. Since the training set was randomly collected, the testing set is also a random collection of image data. The process of testing, use the unknown data provided by the test set in order to evaluate the performance of the classification system. So for each test sample the classification algorithm will provide a decision label, i.e, a class label assigned to the unknown sample. If the decision provided by the classification algorithm describes the same label as the true label of the testing sample then the sample is correctly classified. By measuring the successful decisions of the classification algorithm with respect to the number of the testing samples we are able to calculate a success rate.

The efficiency of an image classification system is measured by the success rate achieved in multiple datasets. The success rate is based on two parameters, the data used for classifications and the classification algorithm itself. The most used data for image classification are the final image representations, presented in detail in Section 2.4. For classification, the Support Vectors Machine (SVM) algorithm is used in most image classification methodologies. This algorithm and its advantages are described in the ensuing section.

2.5.1 Support vector machines

Support Vectors Machines (SVM) [56], is the most used classification algorithm in image classification [2, 7–11, 13–30, 43, 44]. In its most simple example, the SVM algorithm is trained to find a hyperplane separating the data forming two classes. The hyperplane is placed among the data thus representing the separation between each class. This is done by finding the intermediate distances between data found by the boundary samples of each class. The SVM algorithm decides the label of an unknown sample by finding in which side of the hyperplane that sample is found.

Linear classification is possible using the SVM when the separation of data is linearly possible. Although, in most cases data is not easy to separate because the classification problem is more complex. An advantage of the SVM algorithm is that it can handle both linear and non-linear classifications. The non-linearity on classification is achieved by using of the *kernel trick* which projects the original data to a higher-dimensional data space. The implementation of the kernel trick is easy and can provide separable data in higher-dimensional spaces.

For multiple classes the training process is little more complex since a number of hyperplanes must be computed. The first categorization strategy is the *one-against-all* [57] which uses the data samples of one class and tries to separate them from all other samples. This method constructs j number of hyperplanes for j number of classes. The decision function of one-against-all strategy is presented by the following equation.

$$f^j(\mathbf{p}) = (\sum_{i=1}^{M} a_i^j \mathbf{p}_i)^T \mathbf{p} + b^j = \mathbf{w}_j^T \mathbf{p} + b^j \tag{2.19}$$

which separate the jth class from the rest. In eq. (2.19) $\mathbf{w}$ is the normal vector to the separating hyperplane, a the Lagrange multiplier, and b the y-intercept of the border line created by the support vectors. If Λ is the number of classes and M the number of available training images, the SVM learns Λ linear functions using the representation vectors $\mathbf{p}_i$ as training data, $\{(\mathbf{p}_i, y_i)\}_{i=1}^{M}, y_i \in \Upsilon = \{1, ..., \Lambda\}$. Each linear function $f^j(\cdot)$ is defined by $\{\mathbf{w}_j^\top \mathbf{p} | j \in \Upsilon\}$.

The second strategy is called *one-against-one* [58] in which the hyperplanes are computed between class pairs. So, for each class, multiple hyperplanes are computed separating it from all other classes. These methods provide similar to identical results with the only difference being the computation process of each method.

The above specifications of the SVM algorithm show that it can solve complex classification problems. These attributes are highly regarded by researchers of image classification which use this algorithm as their default classification methods.

2.5.2 Disadvantages of classification algorithms

The SVM algorithm is used for classification in most image classification systems. It does not present any serious disadvantages, except the computational cost of the algorithm when used in large datasets.

However, researchers "force" image representations to be presented in vector form in order to benefit from the efficiency of this algorithm. This setup limits the design of representations and adversely affect classification performance. Especially image representations comprised by individual vectors, can lose their discriminatory capabilities when concatenated in a vector form. A research about the effects of image representations on classification, and a solution to the aforementioned problem is presented in Chapter 7.

2.6 Deep learning research on image categorization problems

Deep learning is topic of machine learning describing multilayer architecture algorithms that learn representations of input data. Conventional machine learning approaches rely on hand-crafted features that represent input data in order to achieve a recognition task. However, the construction and use of these research-defined features is limited by their shallow architecture which cannot comprehend and efficiently describe the nature of input data. Deep learning algorithms overcome this problem

by applying multiple layer, i.e, a deep architecture, on feature extraction and data representation. Deep architecture allows the creation and representation of features with respect to the nature of input data.

The implementations of deep learning algorithms have been successfully applied to text recognition [59], object detection [4, 6], object recognition [5, 60], face recognition [61], scene parsing/labeling [62]. Deep learning algorithms benefit from their architecture, learning features more efficiently thus achieving state-of-the-art results in all aforementioned research topics. We continue this section by presenting two of the most commonly implemented deep learning algorithms.

2.6.1 Deep Belief Networks

Deep belief networks (DBNs) [63] is a generative graphical model separated in a layer of visible units and multiple layers of hidden units which are stochastic. The layers present directed or undirected connections between variables of the previous layer. Layer architecture consists of stacked Restricted Boltzmann Machines (RBMs), which learn high-level feature representations in an unsupervised manner.

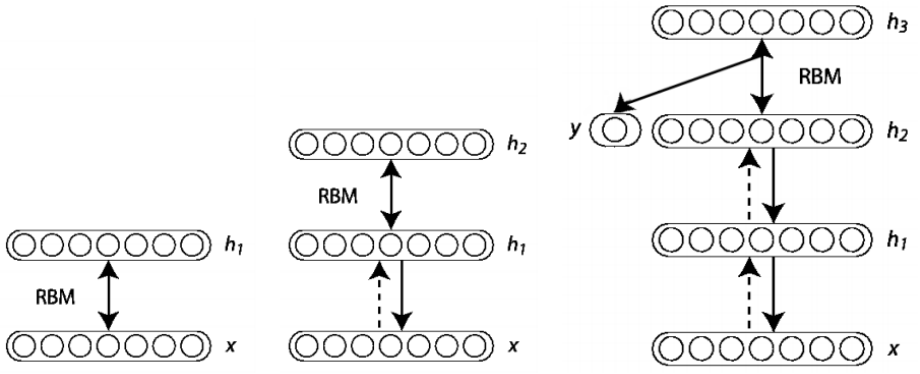


Figure 2-4: The iterative pre-training model construction of a Deep Belief Network as presented in [3].

The most important contributions in the development of DBNs was presented in the work of [63], where a layer wise greedy training was proposed thus achieving better performance and reduced computational cost. More recently [64] proposed a

Convolutional RBM (CRBM) where layer connections are available among all image locations. By creating multiple layers of CRBMs, followed by a max pooling process, this architecture can learn hierarchical representations of images achieving top-performing results. However, DBNs have the disadvantage of all unsupervised training algorithms where the lack of training labels affect the classification performance.

2.6.2 Convolutional Neural Networks

Convolutional Neural Networks (CNN) is a variant of feed forward neural networks where each neuron corresponds to overlapping regions that mimic receptive fields. The first implementation of CNN was introduced in 1980 by Fukushima in [65]. A very influential model for the development of CNN was presented in [59]. The LeNet-5 CNN, is a fully connected multilayer network that extracts features from input data using convolutional and subsampling layers. This model can effectively classify numbers with successful applications on recognizing checking numbers. However, this model was prone to overfitting and could not be applied to solve more complex problems. In [12], authors proposed an improvement of LeNet-5 that introduces a more deep architecture (7 layers) consisted of 60 million different parameters. The training and testing of this CNN was made possible by implementing software optimized for graphical processing units (GPUs). A dropout process that prevents overfitting was introduced making this model very successful in image categorization, achieving state-of-the-art results in very large image datasets.

The architecture of CNNs can be separated in two sections that include a *feature section* and a *decision section*. As feature section we denote those layers that either use the input data or the output data of other feature layers in order to create image features. Features that come from the last feature layer are passed to the decision section. This section consists of layers degrading features in a final representation creating decision labels for each input data.

The individual types of layers consisting a CNN are the convolutional layers, the pooling layers, the ReLu layers and the Loss layers. These layers can be found in

different order, with different parameters and can be found in the feature or decision section of a CNN. This leads to the creation of specific architectures that encounter different machine learning problems.

The **convolutional layers** are the main type of layers consisting a CNN. This type of layer use either the input of raw data (first layer) or the output of previous layers. The data is collected using an overlapping process that creates feature maps used in a discrete convolution with the convolution kernels of this layer. So, for input data x, the k_{th} feature y^k of a given layer can be computed as follows.

$$y_{ij}^k = \tanh((W^k * x)_{ij} + b_k) \tag{2.20}$$

where W^k is a linear filter containing weights, known as the convolutional kernel and b_k is that layer's bias parameter and i, j are indexes corresponding to columns and rows. Eq. 2.20, shows the feature extraction over a single input channel. However, these equations are applicable in multidimensional data such as color images and combinations between color and other information as shown in Chapter 8.

Pooling layers are used to perform subsampling process on input data, usually the output of convolutional layers. The pooling process extracts one value from a patch of input pixels partitioning input features in non overlapping neighborhoods. The described value is either the average or max value of all the examined pixels inside each neighborhood. This layer reduces the dimensionality of hidden units and provides invariance to local translations, especially in the case of max pooling. Pooling is an important process for complex problems, including object detection and image classification.

The rectifier liner unit or **ReLu layer** uses an activation function usually $f(x) = \max(0, x)$ which increases the non-linearity of a network without affecting the overall process. The CNN models using ReLu layers, usually between the convolutional and pooling layers, have less computational requirements in their training phase. Other functions that can be described by a ReLu layer include the hyperbolic tangent $f(x) = tanh(x)$, its absolute value $f(x) = |tanh(x)|$ and the sigmoid fucntion $f(x) =$

$(1 + e^{(-x)})^{-1}$.

Finally, the **Loss layer** is the final layer consisting a CNN model which is the one responsible for making decisions. These layers represent several loss functions that perform different tasks. The most commonly found is the softmax loss function that can estimate the probability of each class. Sigmoid cross-entropy predicts the independent probabilities corresponding to each class with values ranging from zero to one. And last is the Euclidean loss function where predictions are presented from values ranging from $[-\infty, +\infty]$.

2.6.3 Disadvantages of deep learning techniques

Despite their advantages, the use of deep learning algorithms comes with a great computational cost. The proposed multilayer architecture requires a great magnitude of training data and exhaustive training. The exacting iterative training process is caused by the back-propagation phase (or up-down process of DBNs) in order to minimize the training error. Researchers have resorted to multi-processing techniques in order to resolve this issue. But even with modern techniques, using graphical processing units (GPUs) performing faster computations than processors, the training phase can take days up to several weeks.

Chapter 3

3D data acquisition for dataset composition

In this chapter, we present the process of designing and compiling our novel 3D image dataset. We compiled this dataset in order to create a solid benchmark for our ongoing research on 3D image classification.

The common way of scientifically evaluating an image classification method requires an experimental procedure, where the success of a method is measured by classification rates on several datasets. Modern theories in learning and evaluation suggest that datasets cannot reach the variety of images and objects found in the real world. As a result, some proposed methods may perform well in fixed datasets but may not be applicable in real world scenarios. So, researchers seek datasets representing real-world data in order to evaluate and present the validity of their methods.

Image classification and object categorization for 2D images is a well established problem and an area of active research. This research is supported by a large multitude of image databases that are publicly available. These datasets are consisted of a large number of images divided in several classes. While several datasets [2,66–71] are available for testing 2D image classification algorithms, 3D databases have been less widely used. That is because 3D image sensors were either not publicly available or required complex construction and operation.

Modern 3D acquisition sensors became publicly and commercially available in

recent years. The most popular acquisition sensors are range cameras, including 3D scanners and stereoscopic cameras. Both sensors have the ability to capture depth images, i.e., images showing distance values per pixel. Depth information, depending on sensor and application, can be provided as a depth map or a 3D point cloud. In image classification, both these formats can be useful providing shapes, surfaces, and other information about the scene or depicted objects.

The structure of databases used for image classification or object categorization must have a variety of specifications. A good example, should include numerous images with a wide variety of classes. Images should be of sufficient quality and include depth information in the form of a precomputed depth map. They also should not require any further preprocessing before usage.

Early multi-view databases [72, 73] were largely unsuitable for image classification because they were compiled for different research purposes. An early effort on 3D object categorization was presented in [74]. That work was also accompanied by a small dataset, considered unsuitable by modern standards. Novel 3D datasets used for object categorization [7, 75] consisted of objects. They are captured using a turntable setup where multiple views of the depicted object are collected.

There are also datasets compiled for other problems of 3D imaging. These datasets propose individual architectures and setups according to their intended use. Other datasets were designed regarding segmentation and pose estimation problem [7,76–79] under controlled conditions, where multiple objects are found in a scene. There are also datasets created for scene annotation and object detection [7, 80–87] which are collected from modern 3D scanners under less controlled conditions. Other datasets are used in recognition and mapping of surrounding environment (SLAM) as well as camera pose estimations [88–94]. Finally, there are datasets using human activity as data for various applications. Some of these applications include tracking objects and human activity [95–99], person identification [100], human gestures [101,102] and face detection/recognition [103, 104]. Of course, all the above datasets and topics are a fraction of the possible applications that can be researched by the emerging topic of 3D imaging.

At the beginning of our research, we noticed the lack of datasets suitable for image classification. To this end, any proposed method could only be tested by one dataset [7]. That fact motivated us in the designing and collecting of a novel dataset, with respect to the aforementioned attributes regarding image classification datasets. Our novel database differs from other modern approaches, introducing a novel compilation framework and acquisition process. To our knowledge, the proposed acquisition format and organization, was never used in any other modern 3D dataset. We used a digital stereoscopic camera, that is not greatly affected from illumination changes under natural conditions. Unlike other modern datasets [7, 75], that sensor type gave us the advantage of capturing both indoor and outdoor images. As a result, we provide a more realistic benchmark for every image classification applications.

In the ensuing Section 3.1, we present the related work of other datasets that can be used for 3D image classification and 3D object categorization applications. Section 3.3, presents our novel dataset describing the design, the acquisition process as well as the uses of our proposed dataset from other research topics. Finally in Section 3.4, we draw conclusions and share our future plans regarding the development of our novel database.

3.1 Related work

An early work in 3D object categorization [74] introduced the **3D object categories** dataset. That dataset consists of images representing 8 objects including bikes, shoes, cars, irons, computer mice, cellphones, staplers and toasters. Each class, contains 10 different objects captured in multiple views. Due to the use of a common digital camera, authors had to develop an acquisition process including 8 viewing angles, 3 heights and 3 scales resulting to approximately 7000 images. Interestingly, the database was compiled using indoor and outdoor images forming a dataset with modern design. Some images found in that dataset are shown in Fig 3-1.

However, the proposed framework in [74] was unable to recover full 3D object geometry and depth values. This was caused by the use of a single camera and the

Figure 3-1: Example images from the 3D object categories database.

predefined capturing process. In 2007, when [74] was published, the most sufficient way to capture depth was by using two individual cameras on stereoscopic setup. However, this method presented 3D objects by using mutual homographic transformations. Nevertheless, the setup capturing the 3D geometry plus the small size of that dataset makes it not suitable for benchmarking modern 3D methods.

The first large scale 3D image dataset suitable for categorization was the **RGB-D dataset** [7]. That dataset was designed for testing multiple problems of 3D imaging including object recognition, object detection, video annotation and segmentation algorithms.

The database consists of 300 everyday objects divided in 51 classes with images taken from multiple angles. This was possible by placing the objects on a turntable turning at a slow constant speed. All images were captured indoors and under controlled illumination setup. Image data were also collected at three different height of 30, 45 and 60 degrees angle to the horizon created by the turntable. The number of images is approximately 250 frames per object, also organized in video sequences.

Image acquisition was done using a RGB-D camera capable of capturing both texture and depth information simultaneously. Using the aforementioned setup the dataset consists of 250000 frames depicting texture and depth information. The texture images contain background, showing the white turntable. But they are also available in a cropped form, removing the background and highlighting the object. Each object has additional depth information in the form of depth maps and 3D point clouds. Example images from the dataset are shown in Fig 3-2.

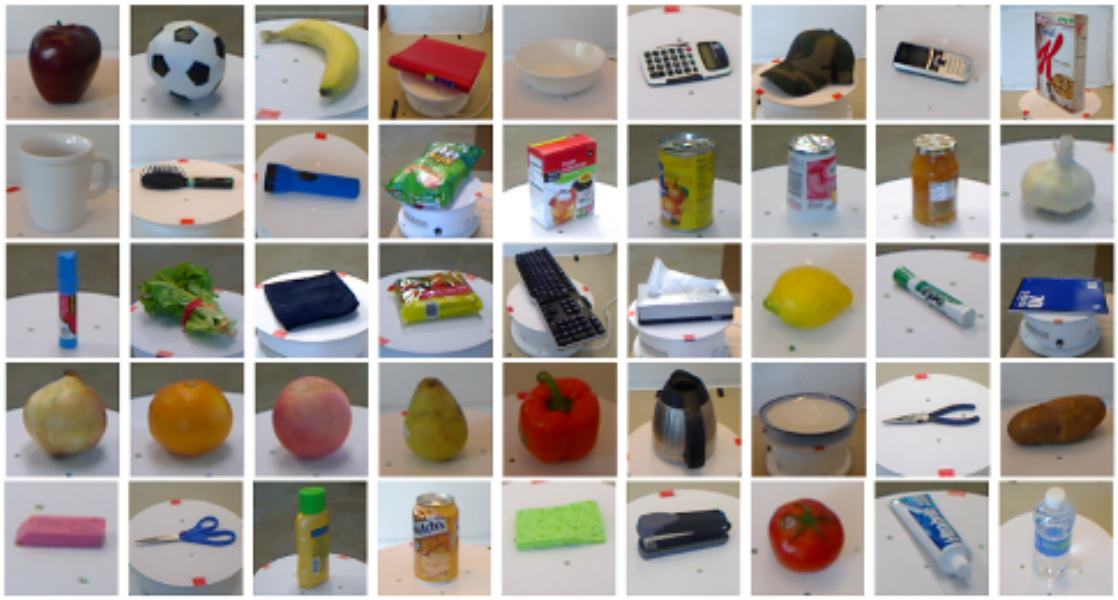


Figure 3-2: Sample images from the RGB-D dataset

As described in Chapter 1, the categorization of objects makes that dataset also suitable for testing image classification methodologies since these topics are related. Overall, the large number of images makes that database a quite demanding benchmark. That dataset provides medium resolution images combined with qualitative depth information, an attribute needed for a modern benchmark. Nevertheless, the specific acquisition setup and sensor does not cover outdoor scenarios and uses.

More recently the **BigBIRD** [75], i.e, (Big) Berkeley Instance Recognition Dataset dataset was introduced, proving the growing need for 3D datasets. That dataset was created in order to address shortcomings of other 3D image datasets, such as low image quality and less detailed depth maps.

That database (in its current version) consists of 125 household objects which are not registered in classes but organized as different instances. For each object, 600 RGB-D style images are acquired using a glass turntable with white background providing texture and depth information. Those images are depicting the object from five different height angles ranging from zero to 90 degrees and every three degrees horizontally concluding to 120 views. All images were acquired using a combination of high-resolution digital cameras and RGB-D sensors. The equipment was placed on a rig with five different bases corresponding to the five different height angles. All texture images are of high resolution with accurate calibration, also provided with object segmentation. Depth information is provided by a depth map and full-object 3D meshes. Example images from the dataset are shown in Fig 3-3.

Figure 3-3: Sample images from the BigBIRD dataset

Despite its detail and quality, this dataset is far from suited for benchmarking image categorization methods. The main issue is that objects are not organized into classes, and the variety of objects leave no room for such organization. Also, as mentioned earlier, the acquisition framework used in that database makes it unsuitable for benchmarking methods for outdoor scenarios and uses.

3.2 Review of 3D acquisition sensors

As described above, the design of our proposed database focuses on describing realistic acquisition conditions, thus creating a challenging benchmark for 3D imaging problems. To this end, we capture objects in realistic conditions of lighting, background and shading. Our goal was the extraction of depth information from objects, thus capturing object properties such as shapes, surfaces, etc.

Figure 3-4: A collection of modern 3D sensors.

Depth can be acquired by range imaging sensors, cameras able to create images depicting distance values in each pixel. The selections of available sensors when compiling our dataset was limited to structured-light 3D scanners (Microsoft Kinect, Asus Xtion) and digital stereoscopic cameras. A collection of these sensors can be seen in Fig 3-4. Structured-light 3D scanners are also referred to as RGBD cameras. Other selections such as LIDARs, Time-of-flight cameras (ToF) and other types of 3D scanners were disregarded due to cost and difficultly of use.

The Microsoft Kinect sensor is a **RGB-D camera**, i.e, a structured-light 3D scanner combined with a RGB sensor. It was firstly introduced as a motion sensor for the Microsoft Xbox gaming console. This sensor has the ability to capture depth information by using an infrared laser projector that constructs a specific light pattern. When combined with a CMOS sensor and a dedicated firmware, Kinect can translate deformations on the laser pattern into depth information. An example of

this process can be seen in Fig 3-5.

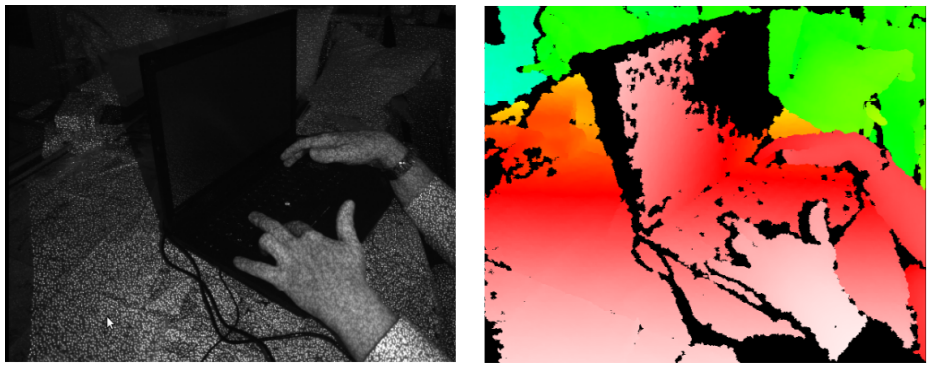


Figure 3-5: The process of making a depth map from infrared light patterns of Microsoft Kinect sensor.

This sensor was initially used for detecting motions of users in order to control some functions of the gaming console. Researchers and computer vision enthusiasts became interested in the abilities of Kinect and "hacked" the sensor by using custom software. The growing uses of Kinect caught the attention of Microsoft which provided drivers and a software development kit. The sensor is able to provide RGB images, i.e, texture images, and image streams using an 8-bit VGA with 640×480 resolution and is even capable of collecting 1280×1024 on lower frame rates. The structured-light sensor provides detailed depth maps with low deformations ranging from 1.2 to 3.5 meter approximately. The combination of texture and depth images is often referred to as RGBD or RGB+D images. The aforementioned specifications made Kinect very popular among researchers in 3D object recognition, resulting in the first database presented in [7].

This sensor comes with several limitations that should be taken under consideration. In order to capture an image comprised by texture and depth, the sensor requires a computer connection and capturing software thus affecting portability. It also requires an independent power supply because the computer connection can only provide data transfer. Finally, the infrared ranging sensor can be easily corrupted by natural lighting. As a result, the use of this sensor constitutes a bad choice for outdoor applications.

Stereoscopic cameras are acquisition devices built with two individual lenses

emulating the binocular vision of humans. Binocular vision is the ability of creatures to perceive depth using a pair of eyes. Each eye receives an image slightly shifted from the other. The brain can perceive depth by measuring the disparity between images from each eye.

Depth information of stereo digital cameras is extracted using stereo triangulation. The calculation of depth pixel values is done by finding the horizontal disparity of corresponding regions between images. This problem can be solved by stereo correspondence algorithms that provide a depth map for the depicted image scenery.

The most common setup of previous years required two individual digital cameras mounted in pairs, in order to capture stereoscopic images. These setups were bulky, required perfect alignment and in most cases synchronization between cameras. In recent years, the advances on 3D camera design developed two main camera types. The first camera type consists of two lenses corresponding to one image sensor. The second camera design includes a pair of independent lenses corresponding to individual acquisition sensors. Essentially the first type is less complex but provides lower resolution images and difficult image stereo processing, since two views share a common image sensor. The second type of stereoscopic cameras has the advantage of eliminating alignment, synchronization and low resolution issues since two independent image sensors are controlled by the same camera. Both types of stereoscopic cameras are found commercially as portable handheld devices and provide ease of operation.

3.3 Brunel Texture and Depth Image database (BTDI)

In recent years, the use of 3D image acquisition sensors became more popular among researchers. Research of new 3D imaging methodologies required more 3D image datasets. At the beginning of our research, we realized the lack of datasets capable of testing 3D image categorization methodologies. The need for challenging benchmarks in which we could present our 3D image classification methods became apparent.

Most publicly available datasets were either created for different implementations

or outdated by modern standards [72, 73]. It became apparent that in order to cover the needs of our research we had to compile our own dataset. This research provided a better understanding about the nature of 3D acquisition sensors and acquisition techniques. We wanted to compile a database with no limitations in lighting, image quality and resolution leading to less challenging benchmarks. Therefore, we introduced a novel 3D image dataset called *Brunel Texture and Depth Imaging dataset (BTDI)* [43] consisting of stereoscopic images.

3.3.1 Designing a dataset

Dataset composition starts by carefully planning the acquisition requirements and intended uses. In order to complete the design of our novel dataset, we took under consideration the format of other known 2D and 3D databases.

Known 2D image classification datasets [2,66–71] usually include images collected from web image searches. Images consisting these datasets contain one object per image, usually found in the center of each image with little background clutter. These datasets have an easier composition process often using data available online. However, the magnitude of collected images create problems in data organization. The aforementioned image specifications, i.e, little background clutter and objects depicted in the center, may lead to less efficient methods for real life applications. This is also the case in modern 3D databases [7, 75] where images and depth maps are, in their majority, acquired under controlled environments. We consisted a dataset from images captured under controlled conditions and realistic imaging scenarios. BTDI dataset is a good mixture of controlled and uncontrolled photographic environments targeting at a wider range of applications and more challenging benchmarks.

All of the aforementioned datasets present objects which are separated in multiple classes. The contents of these image sets are in their majority common objects, such as household objects. The identification of images containing common objects have a direct application on robotics and other applications. We wanted the same indented uses and benefits of this design for our proposed dataset. So, we captured common objects in natural environment and under controlled indoor conditions. The

main indoor image theme is common household objects, like clothing and cutlery. Outdoor images mainly represent buildings, trees and plants, some of them under difficult illumination conditions. Some objects corresponding to the same class are depicted both indoors and outdoors, thus creating a challenging intra-class format. For instance, one of the categories, clocks in particular, represents both indoor and outdoor images, depicting table clocks as well as building clocks. The collection of all classes can be seen in Fig 3-6.

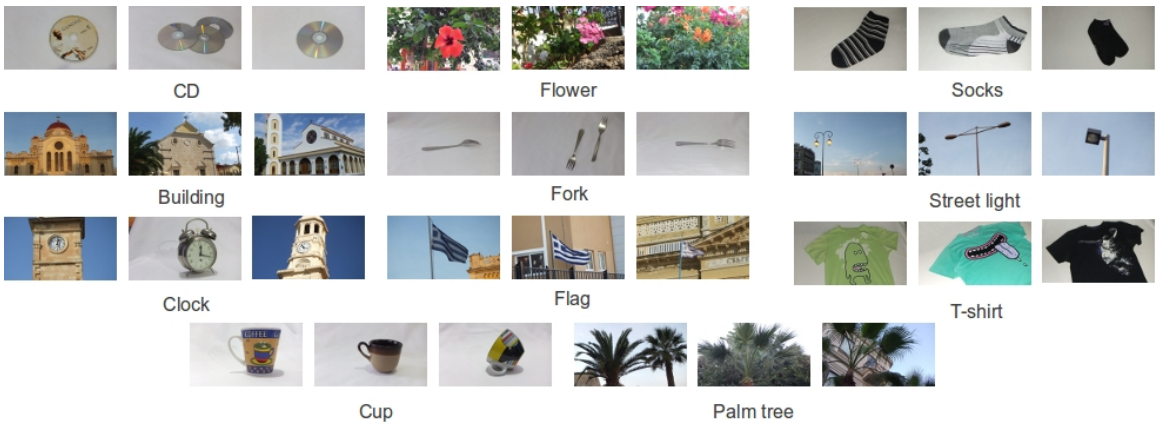


Figure 3-6: Example images from the BTDI database.

An important attribute for all dataset designs, is a proper number of classes and images. In some cases [70, 75], the number of classes was not fixed but is constantly expanding. The collection of 3D image data is a very complex process requiring human validation in order to preserve the good quality of a dataset. This validation process reduces the initial number of acquired images, because some of them are not of sufficient quality both in texture and depth. But the number of images per class is a very crucial issue in the design of a database. So, the more images per class the better a database becomes. The first edition of our database consists of 98 stereo image pairs organized in 13 classes, a challenging benchmark used throughout our experiments. However, we are still improving and expanding our novel dataset with new classes and more images per class. More information about the future expansion and the second editions of this dataset are presented in Section 3.4.

3.3.2 The database acquisition process

For the acquisition process of our database we used the Fujifilm FinePix Real 3D W3 stereoscopic digital camera. This camera has individual lenses corresponding to two ten megapixels 1.2/3-inch CCD image sensors. The device also provides 3× optical zoom and a 3D macro function for closer shots. In normal 3D mode, the minimum distance between the lens and the depicted object can be approximately one meter. In 3D macro mode the minimum distance for real stereoscopic images is approximately 38 centimeters. This device can achieve high-resolution images up to 3648×2736. Nonetheless, all images in our dataset have a resolution of 1920×1080 pixels. All images were formated in .JPEG containing the RGB image and .MPO files containing the pair of RGB images captured from both lenses. For the stereoscopic settings we used the auto parallax setting of the camera, a function that automatically controls the disparity between the image pair. Depth information for each image consisting our dataset, can be extracted by using any stereo correspondence algorithm. To our knowledge, this was the first attempt using a stereoscopic digital camera for the collection of 3D images comprising a modern image database.

Figure 3-7: The process of taking indoor pictures of objects using a mini studio setup.

All collected images are captured under indoor and outdoor conditions. A great

advantage of stereoscopic camera is that they do not require special illumination conditions and they are not affected by natural light. However, the main limitation for both indoor and outdoor setups was shades created from scenery or depicted objects. Shades create a correspondence region according to most stereo correspondence algorithms. So, shades can be falsely assigned with depth values that corrupt the overall depth map.

For our indoor setup, shown in Fig 3-7, some images were captured using a white sheet and artificial lighting forming a mini studio. The white sheet provides a good background without clutters and directed artificial lighting provides less shades from the depicted object. However, some objects were captured indoors with background clutter and under room illumination conditions.

Figure 3-8: The process of taking outdoor pictures of objects.

Outdoor images are more realistic, because they were captured under natural illumination conditions. Most images were captured during sunlight but there are some taken at dusk when natural light was limited. All outdoor images have background clutter from the surrounding environment. Finally, shades of natural scenes were avoided when possible. An example of our outdoor setup can be seen in Fig 3-8.

The overall collection process was not very difficult, and did not require any advanced photographic skills. This means that our database is easily expandable and does not require any special skills to collect more images and classes.

3.3.3 3D imaging uses for our proposed dataset

We compiled this dataset in order to test our 3D image classification methods. Nevertheless, it became apparent that collected data could also be used to benchmark other types of imaging problems as well.

The task of **image classification** is to correctly categorize unknown test images with respect to a number of known reference categories. With new 3D sensors now becoming publicly and commercially available, there is a need for extensive research of 3D image classification techniques. The challenges of this emerging topic include the study of depth information and its contributions to improve recognition results.

This dataset includes depth information captured by 3D sensors representing shapes, surfaces or other visual features of objects and scenery. The combination of depth and texture information can improve existing methods or create novel methods for feature extraction, dictionary learning, image representations and classification.

Segmentation is the process that successfully partitions an image into individual regions. These regions are usually exploited from other methods for further image analysis. Therefore, such regions must have a meaning according their intended use. Most commonly segmented regions may include objects, scenery or multiple parts of them. Separation into regions is commonly based on color, texture and intensity pixel values.

Still, this dataset can also provide depth information that can be used interdependently or combined with other image information. Interesting applications that could benefit from the use of depth are 3D object/ scenery segmentation and automated 3D object/region annotation on segments considered interesting for further processing.

As described above, **stereo correspondence** is the process where a pair of stereoscopic images produces a depth map. The correspondence algorithm creates a process of finding corresponding regions in both images. Then, the horizontal disparity be-

tween the corresponding regions is measured, and the result of this process provides a depth map. The corresponding regions are usually found by correlation between pair regions or corresponding image features in both images.

But the problem of stereo correspondence is far from solved. As we described in earlier sections, shades create depth map corruption which are very hard to eliminate. This dataset can provide a good benchmark for testing stereo correspondence methods, mostly because our dataset is consisted of natural images.

3.4 Conslusions

In this chapter we presented our novel dataset, a collection of 98 stereoscopic images organized in a 13 class. This dataset is compiled from high resolution images depicting common everyday objects under a variety of illumination conditions. Therefore, it can provide a great benchmark for a variety of 3D imaging problems like image/object classification, stereo correspondence and segmentation. Advantages of our database are:

- The acquisition method, which is robust to illumination conditions.
- Our novel database depicts realistic scenes and objects under natural environment with good image and depth map quality.
- Easily expandable database format and ease of use due to stereo image format.

One disadvantage of our dataset's framework is the problem of shades in natural images. Shades in sceneries may be considered as difficult to encounter, but in our opinion, the improvement of future correspondence methods will provide a solution to these issues.

Future improvements for this dataset, include a large expansion both in number of classes and image magnitude. At the moment, there are 12 new classes and a significant increase of 430 additional images depicting different objects. This increase

will make the second edition of our novel database even more challenging. Furthermore, we plan a public distribution for our database making it more popular among researchers.

Chapter 4

Depth feature extraction and combination with texture features

Although 2D image classification methods [2, 10, 11, 13–30] have reached some maturity, smaller progress has been made in the field of classification of images that consist of both texture and depth [7–9]. Such images offer additional information and can be captured with commercial 3D cameras. Considering the growing number of 3D sensors in our everyday life, the efficient exploitation of depth information is an emerging challenge in image classification.

In methodologies [7–9], authors introduce features extracting information from depth exploiting depth information from 3D images. They all use multiple features extracted from texture and depth. Individual image representations, corresponding to each feature, are combined in order to enhance 3D image classification performance. However, these methods have problems on wider applications regarding less detailed depth maps. In [8], authors create features from depth gradients, a technique compatible only with depth maps captured from 3D scanners. Furthermore, they use 3D point cloud-based features, depth information that can only be extracted using 3D scanners. Other significant attributes of depth, such as shape and content information are also disregarded. Finally, the combination of multiple feature representations presented in [7–9], is a computational intense procedure from using high-dimensional vectors in their image representation.

In this chapter, we present a novel depth feature extraction method used for 3D image classification, a work presented in [43]. This part of our research studies the effects of depth information as feature describing image content.

We also introduce a novel image classification method that exploits depth information, if such information is available. The proposed method uses conventional SPM and sparse coding to represent texture images. For depth representation, we chose the description of object shapes, a method suitable for all types of depth information. Specifically, shapes extracted from depth maps include discriminative information which can be harvested using our proposed slicing methods. We introduce four different slicing methods which partition the examined depth map and enhance the efficiency of each shape feature. Shape features are represented using sparse encoding and the BoF image representation method, providing very efficient recognition even when used independently. The combination of texture and depth using our proposed methodology yields improved results for 3D image classification. The flowchart of our proposed methodology can be seen in Fig 4-1.

The rest of the chapter is organized as follows: In Section 4.1, we describe the recent work in 3D feature extraction. The processing of the depth map, the slicing methods and the shape features used are described in Section 4.2. In Section 4.3, the encoding of each feature and data fusion is presented. Experimental results are presented in Section 4.4 and Section 4.5. Conclusions are drawn in Section 4.6.

4.1 Related work

The first stage of describing an image is achieved by the extraction of image features. Image features, are vectors representing significant parts of an image. An important specification of image features is their capability to represent identical or similar image content. So, image features have to be discriminative and not allow different image content to be described by similar vectors. Other important feature specifications include invariance to rotation and scale changes, as well as robustness to illumination and image noises. Using keypoint detectors image regions robust to

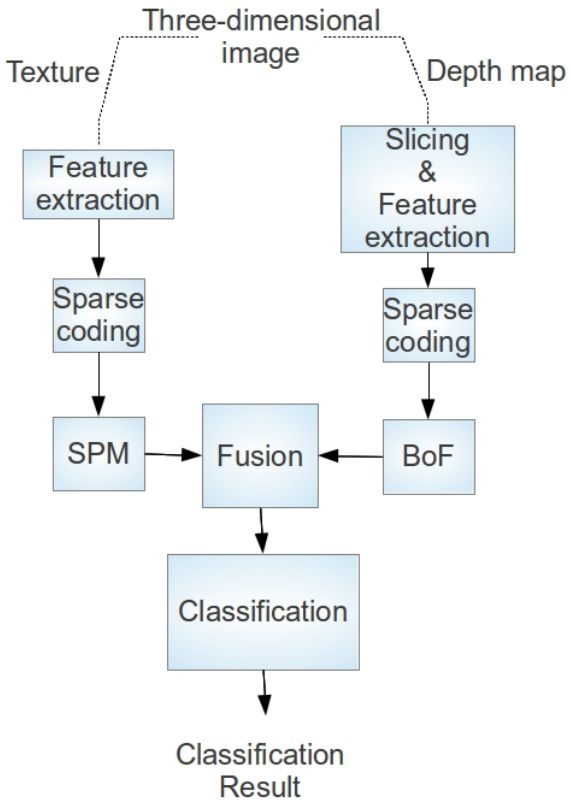


Figure 4-1: Block diagram of the proposed method.

the above conditions can be found. However, the majority of image classification systems computes texture features over dense grids in order to collect more content information from each examined image.

The most famous among feature extraction methods are the Scale-invariant feature transform (SIFT) [1], Histogram of Oriented Gradients (HOG) [32] and Speeded Up Robust Features (SURF) [31]. All the above features are computed by image gradients capturing directional changes of color intensities. Their main differences can be found in their individual representation methodologies for texture gradients. The aforementioned texture feature extraction methods are used in identical or similar applications, with each one having advantages over the others. Nonetheless, most 2D image classification methods use the SIFT feature as their preferred feature extraction method [2, 10, 11, 13–30].

The representational value of depth information had not been studied until re-

cently [7–9, 43]. As described in Chapter 3, acquisition of depth information was challenging thus limiting the demand for 3D image applications. Depth information has a different nature from texture, making traditional approaches of feature extraction not applicable, as seen in Chapter 2.1. As a result, the research for depth feature extraction became necessary in regard to newer 3D applications. The methods in [7–9] presented features extracted from depth information. Their research led to the creation of discriminative features respecting depth information as well as to the combination with texture features for optimized recognition results.

In [7], authors compile a dataset consisted from RGBD images, i.e, images that include both texture and depth information. Showcasing their dataset's potential uses, among other methods, they created an object recognition setup capable of classifying objects to their individual classes. This object recognition system can be considered as an image classification problem since the objects are found in the center of the image and do not need a detection process first.

In order to describe each object, authors proposed multiple image representations derived from texture and depth. For texture information, that method used SIFT features [1] over a dense grid combined with color histograms and texton histograms [40] features. For depth, they used a shape retrieval feature known as spin images [42] applied on 3D bounding cubes each composed from a $3\times3\times3$ grid. All features are encoded using the Efficient Match Kernels (EMK) method [16] and represented by an Spatial Pyramid Matching (SPM) [2] architecture.

As a result, each image has two representation vectors one for texture and one for depth. The representations are classified separately using a Support Vector Machine (SVM) algorithm comparing the recognition results from texture and depth representations. However, better results were achieved from using the combination of image representations in a common vector used for the classification process. More details about this method can be found in Chapter 2.1.2.

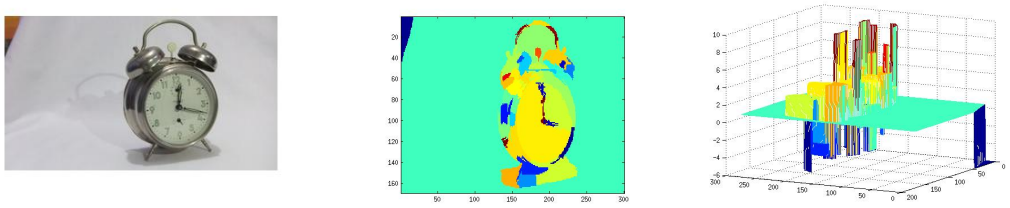
A recent work in [8], introduced a variety of descriptors extracted from depth information. Features are computed using the 2D feature extraction method of Kernel descriptors [10]. In that method image patches are described by a kernel function

instead of a representation histogram used in other feature extraction methods [1, 32]. Authors apply that methodology on depth information, such as depth maps and 3D point clouds. As a result, authors introduced size and shape descriptors from 3D point clouds and edge descriptors from depth maps. The depth map edge features include gradient and local binary pattern kernels. Information regarding these feature extraction methods can be found in Chapter 2.1.2. All images are represented using a pyramid of Efficient Match Kernels [16] for each feature type, thus creating multiple representations describing the same image. In their experimental results, authors create an object recognition system that can be also considered as an image classification problem. The experiments present the recognition performance of each feature type and show that the combination of all feature representations yields the best recognition results.

More recent improvements on depth features and feature combination can be found in [9]. Feature extraction and representation method is based on [11], where a similar method for 2D image classification is presented. Hierarchical Matching Pursuit (HMP) [11] learns image features in an unsupervised manner by collecting and represents image regions in multiple-layers.

The method introduces a two-layer feature extraction each corresponding to its individual image representation. The first layer generates features representing small neighborhoods of image pixels. Raw pixel information, i.e, color or grayscale intensities, comprise the pixel neighborhood creating a first-level feature patch. These features are encoded using the Orthogonal Matching Pursuit (OMP) algorithm and via a spatial pyramid combined with max pooling they result in the first-level image representation. The second layer, generates a representation by sub-sampling features extracted from the first layer. The second layer feature patches are also encoded and represented using a spatial pyramid with max pooling providing the second-level image representation. Finally, both level representations are then combined in a common representation vector suitable for classification.

In [9], instead of using only grayscale intensity values, authors use color and depth information to represent all examined images. Once again, experiments in [9] present



(a) (b) (c)

Figure 4-2: Demonstrating the extraction of depth maps. (a) the original image, (b) the extracted depth map, (c) a different view-point of the same depth map.

interdependently texture and depth recognition rates. However, the best performance comes from the combination of both representations.

As seen above, all methods try to create suitable feature extraction methods in order to efficiently represent depth information. They also proposed the combination of texture and depth information yielding better recognition rates for the RGBD dataset. However, as presented in Chapter 2.1.3, these methods present curtain disadvantages regarding the representation of depth information. Our research focused on novel features with simplified computation that respect the nature of depth information.

4.2 Depth map preprocessing and feature extraction

The most widely-used feature in image classification is the SIFT feature. It was introduced in [1] and it is generally considered to be the most efficient feature for a variety of applications. The SIFT feature provides uniqueness, rotation invariance, scale invariance, and robustness to deformations. It is typically used in conjunction with a keypoint detector that enables the extraction of SIFT information from selected key locations in an image.

Despite their successful application to texture, SIFT features and gradient-based features cannot contribute to the representation of depth information. In most case, depth maps do not exhibit large variations suitable for describing depicted image content. For this reason, we did not consider designing a feature extraction methodology based on depth map gradients. In this work we use SIFT features only to represent

texture but not depth maps.

Depth maps depict depth information of an image. A depth map can be obtained from a variety of sensors such as stereo cameras or 3D scanners. By using depth, an image can be partitioned to regions or objects describing objects and scenery. Therefore, suitable depth features can be extracted using shape analysis. The shape features that we use in our proposed methodology include Radon transform [35], Shape context (ShC) [33], and Shape context with inner distance (ShCid) [34].

Radon transform [35] is a known method used in tomographic reconstruction. It can successfully represent the projection data from tomographic scans, and can be implemented in combination with our proposed slicing process over depth maps (Section 4.2.2).

ShC [33] feature is used for shape retrieval and object recognition. Each object is separated in points of interest which provide a feature histogram created from uniform log-polar bins. This feature is very discriminative and provides invariance to translation, scale and small shape deformations.

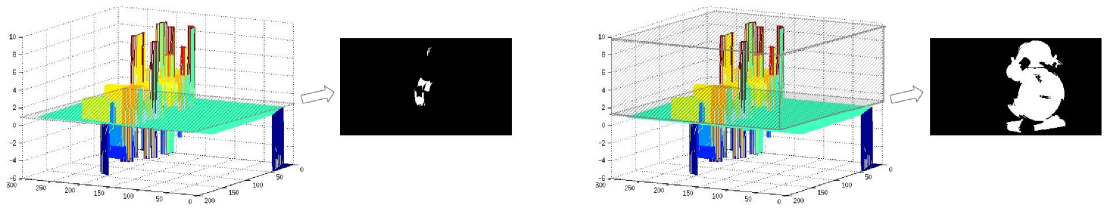
Finally, the ShCid [34] was based on the ShC also describing the distance between interest points. The distance between interest points of ShCid is computed by finding the shortest path connecting two points using only the area inside the shape silhouette. This makes the ShCid more discriminative in shape representation in some cases. All the above features, can represent shapes using shape regions either as contour or finite contour points with great fidelity. Detailed information about these shape-extraction feature methods is presented in Chapter 2.1.1.

4.2.1 Feature normalization

Feature normalization is a simple procedure that contributes to the efficiency of the classification system. Shape-context features, are based on a log-polar histogram represented using decimal numbers. However, it is proven by common practice that mathematic complications and classification algorithms perform better with homogenized data in the range 0 to 1. In this work we use a simple procedure that normalizes every shape context feature $f = [f_1, f_2, ..., f_n]$ as follows:

$$f' = f / \sqrt{\sum_{i=1}^{n} f_i^2} \tag{4.1}$$

where f' is the normalized feature and n is the size of the shape-context feature. This normalization procedure will be experimentally seen to boost the classification performance of shape context feature and also to reduce the complexity of sparse coding and codebook generation.



(a) (b)

Figure 4-3: The two main slicing methods. (a) Tomographic depth slicing: the slice is extracted from one depth layer only, in (b) Progressive depth slicing : the slice is the combination from all depth layers up to current depth layer.

4.2.2 Slicing methods for depth maps

The depth map of an image represents the range of depth levels in the image. In order to facilitate the extraction of depth features, we partition each depth map into areas of different depths. Henceforth, this process shall be referred to as *depth slicing*. Each depth slice is the set of areas on the depth map that correspond to a specific range of depth values.

A depth slice is a binary map that indicates which areas on the depth map are within the prescribed depth range. These slices are likely to capture the shape details of the object or scene depicted in the image, with more detailed regions being extracted when the width of the depth intervals is small. This is true even for classes that are not necessarily characterized by a uniform common shape. We tried four different slicing methods. These are explained below.

Tomographic depth slicing

For every depth value, a slice of space can be extracted. A slice may contain objects, background of a scene or just a parts of them. This method using slices of depth to represent a scene as seen in Fig 4-3(a), a method that resembles tomographic imaging. Similar to the intensity slicing descried in [105], we define the following.

Let D be a depth map with L depth levels ranging in $[0, L-1]$ provided by the sensor or the disparity estimation algorithm. A slice S is defined as:

$$S_l(x,y) = \begin{cases} 1 & \text{if } D(x,y) = l_n \\ 0 & \text{otherwise} \end{cases} \tag{4.2}$$

where l is the depth level index.

Progressive depth slicing

This slicing method combines several depth levels on a single map. Essentially, each slice includes all previous slices. In this way, regions or objects that are found at various depth levels can be captured in a single slice (as seen in Fig 4-3(b)). Therefore, a slice S is defined as:

$$S_l(x,y) = \begin{cases} 1 & \text{if } D(x,y) \leq l_n \\ 0 & \text{otherwise} \end{cases} \tag{4.3}$$

Object-oriented tomographic depth slicing

Simple slices of depth, as defined in eq. (2), will most certainly contain segregated regions. Using the connected components algorithm [106], each depth slice can be partitioned in labeled regions. Independent regions extracted from a slice can be separately analyzed and features can be produced from regions shape.

Object-oriented progressive depth slicing

Using the same methodology as the above method, regions are now extracted on a progressive depth map, such as that expressed in eq. (3). The independent regions

extracted by using the connected components algorithm [106] consist of the union of all previous depths, adding new details on each one.

4.3 Sparse feature encoding and data fusion

Let I be an image comprising texture and depth. Let also T denote the set of *texture* feature vectors in the F_1-dimensional space, where $T = [t_1, t_2, ..., t_N] \in \mathbb{R}^{\mathrm{NxF1}}$. Similarly D is the set of *depth* feature vectors in the F_2-dimensional space extracted from the depth map, where $D = [d_1, d_2, ..., d_M] \in \mathbb{R}^{\mathrm{MxF2}}$. Using the K-means algorithm, a texture codebook $V_t = [V_{t_1}, V_{t_2}, ..., V_{t_h}]$ and a depth codebook $V_d = [V_{d_1}, V_{d_2}, ..., V_{d_k}]$ are trained, where h and k are the number of cluster centers in the two codebooks respectively.

Using the above codebooks, each texture or depth feature vector can be approximated by a linear combination of the reference features in the codebook. Specifically, sparse vectors (codes) $C_t = [C_{t_1}, C_{t_2}, ..., C_{t_N}]$ and $C_d = [C_{d_1}, C_{d_2}, ..., C_{d_M}]$, for texture and depth respectively, are computed for texture by minimizing the l_1 norm [53]:

$$C_t = \min_{C_t} \sum_{n=1}^{N} \|t_n - C_{t_n} V_t\|^2 + |C_{t_n}| \tag{4.4}$$

and similarly for depth:

$$C_d = \min_{C_d} \sum_{m=1}^{M} \|d_m - C_{d_m} V_d\|^2 + |C_{d_m}| \tag{4.5}$$

where $||\cdot||$ denotes the l_2 norm and $|\cdot|$ the l_1 norm.

From that point on, each sparse code is treated separately. The encoded depth features are shown in Fig 4-4 for three different depth features. As seen, the vectors calculated when the Radon feature is used are not sparse, meaning that Radon transform in its present form is not suitable for the application considered. The shape context features, however, yield extremely sparse coded representations, a fact that highlights their appropriateness for sparse coding.

The resulting texture sparse codes C_t are processed by a linear Spatial Pyramid

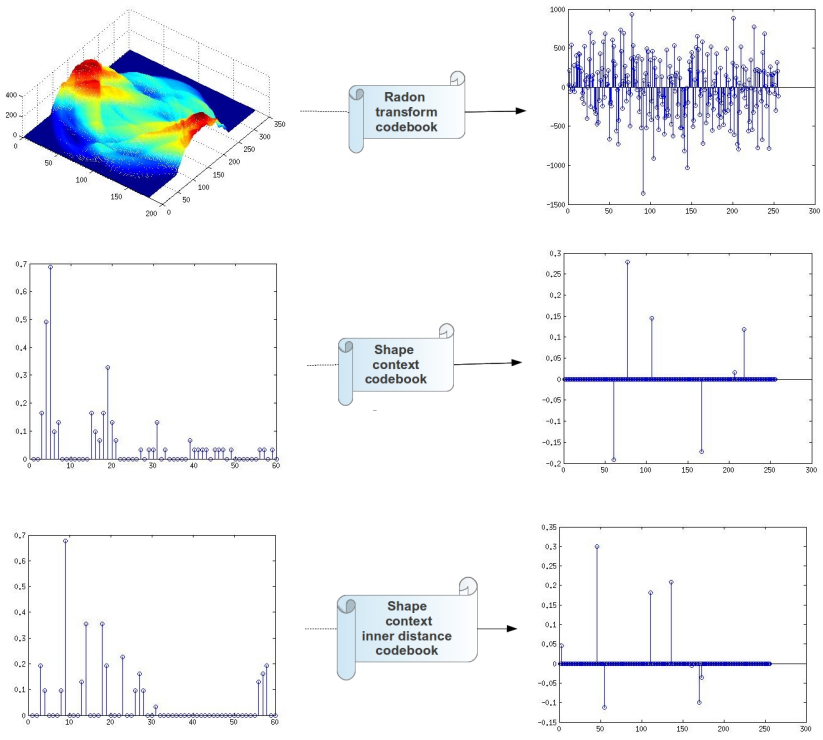


Figure 4-4: Depth shape feature l_1 encoding. Left column: extracted depth features. Right column: sparse codes.

Matching (SsSPM) kernel set as P found in [17]. The associated max-pooling function is defined as follows:

$$p_j = max\{|C_{t_{1j}}|, |C_{t_{2j}}|, ..., |C_{t_{Qj}}|\} \tag{4.6}$$

where p_j represents the j_{th} element of P, Q is the number of regional descriptors and the $|\cdot|$ is the absolute value of sparse vectors. The final representation is formed by combining the p_j vectors above into a spatial pyramid [17].

The depth sparse codes, denoted as C_d, are subjected to a max pooling process. Similar to the max-pooling operation expressed with eq. (6) for the texture feature, the sparse codes calculated using eq. (5) are also jointly processed. In the case of depth, however, sparse codes are not taken on local regions but instead they are combined all together as follows:

$$B = max\{|C_{d_1}|, |C_{d_2}|, ..., |C_{d_W}|\} \tag{4.7}$$

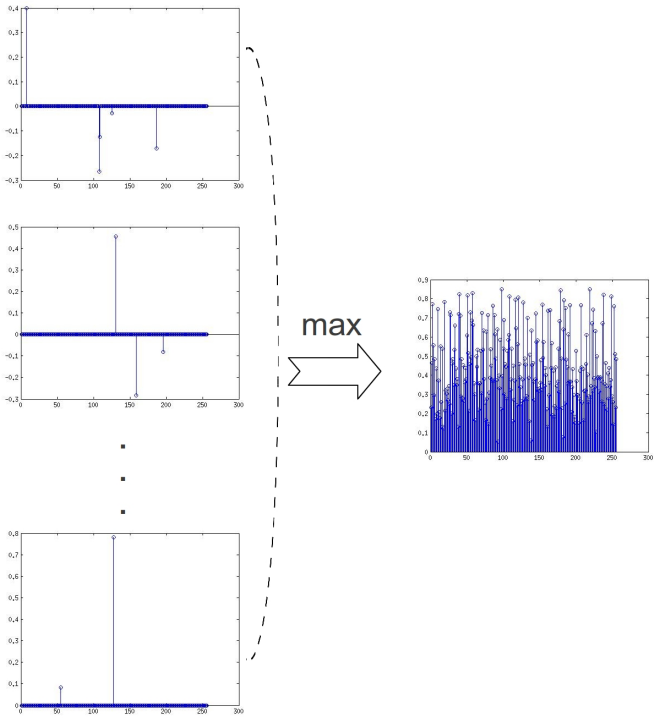


Figure 4-5: Max pooling using depth sparse codes.

where W is the number and $|\cdot|$ is the absolute value of sparse codes. The max pooling operation is graphically shown in Fig 4-5.

4.3.1 Data fusion

The outcome of the preceding analysis are two separate sparse representations, one for texture and another for depth. A fusion method should merge the representations in a simple way by taking into account the dimensionality of the resulting fused representation [7–9, 107].

The most straightforward method is the concatenation of the two vectors. In this way the final feature representing the image is:

$$U = [P, B] \tag{4.8}$$

4.4 Experimental Results

4.4.1 Experimental setup

For the experimental evaluation of our method we use the framework introduced in [17] and subsequently used in several other works on classification. The aforementioned framework was adapted for use with our database and the descriptors extracted from depth maps. Experiments were designed in order to assess shape descriptors and slicing methods. Our experiments where conducted on the BTDI database.

Codebook training took place by randomly picking 500 shape descriptors that yielded 256 centroids (codewords) after 50 K-means iterations. The final feature vectors were formed, as described in the previous section, by means of sparse coding and max pooling. Finally, each depth representation was used in a multi-class linear support vector machine (SVM) for classification purposes. For SVM training, only one image per class was used (13 images in total), while the rest of the images were used for testing. This took place for five different training-testing configurations and the associated results were averaged.

4.4.2 Slicing parameters

Our first experiment aimed to evaluate the efficiency of our depth slicing approaches and gain a better understanding of the impact of depth maps in our method. For this reason, we tried the same depth map feature with each of our slicing methods in order to access their efficiency. The measure that we used for slicing method evaluation is the resulting classification percentage.

Table 4.1: Depth map classification without feature normalization.

Descriptor Type	Slicing methods			
	1	2	3	4
Radon transform	12.90%	10.80%	8.83%	10.32%
Shape context	14.76 %	12.37 %	22.56 %	13.55 %
Shape context with inner distance	13.58%	13.54%	12.6%	18.81%

Slicing methods

Table 4.1 provides evidence about the way with which the slicing methods affect classification performance. As seen, Radon transform performs better in conjunction with tomographic depth slicing. As mentioned above, this method provides distinct regions that are not always connected and represent an image at increasing depth values. However, other slicing methods appear to perform better if other features are used. Therefore, the conclusion that follows is that slicing methods should be coupled with the appropriate feature in order to yield optimal performance.

Table 4.2: Experiments on depth map slicing rate.

Slicing depth step experiments on depth map classification				
Feature	Slicing method	Slicing step	step 1 dictionary	step 3 dictionary
ShC-BoF	Progr. depth slicing	1	16.99	15.35
ShC-BoF	Progr. depth slicing	3	15.01	17.09

Experiments on sampling rate

A further parameter that affects performance is the number of slices in which the depth map is partitioned. The experimental process was set up with progressive depth slicing and depth feature normalization, to reduce computational complexity. The experiment was calculated with a codebook which was trained using features extracted from a large number of depth slices, i.e., each slice represented a narrow depth interval. The second experiment tested classification performance using a dictionary trained with a smaller number of slices. Results for both cases are reported in Table 4.2. As seen, performance is better when the depth slicing strategy for extracting features from test images is the same as that used for codebook generation.

4.4.3 Descriptor efficiency

Shape descriptors extracted from depth maps should have the ability to classify images even when texture information is not used. For this reason, we assessed the distinc-

tiveness of each descriptor extracted from a depth map. These tests were conducted without feature normalization, in order to facilitate assessment of the independent application of features.

The first tested feature was the Radon transform with all the slicing methods. Results are reported in Table 4.1. Each reported result is the mean classification percentage for the entire database. It is seen that this feature is generally unable to represent efficiently shapes within depth maps. This was somewhat expected, because Radon transform usually performs better when describing clearly defined silhouettes and shapes.

Shape contexts (ShC) are more discriminative features. As seen in Table 4.1, the shape context feature outperforms Radon transform for classification based only on depth maps. Furthermore, shape-context features are preferable in terms of computational cost because they result in shorter feature vectors. The size of the descriptor is proportional to the number of bins and the range of the log-polar histogram, but based on experimentation we conluded that the best combination comes from 12 bins and range up to 5 pixels, resulting in a 60 dimensional vector. A big drawback of the ShC feature is that it permits the computation of features on every point of a shape contour, yielding a potentially unmanageable number of features. To overcome this problem, we randomly selected 100 features from the contour of each region. Another solution to this problem could be a contour point evaluation before the extraction of keypoints. Despite the above difficulties, ShC performs quite well as a depth map classification feature.

Shape context *with inner distance* (ShCid) was also tested. Although ShCid were introduced [34] out of a need to deal with shape context drawbacks, image classification based on depth maps did not improve. In general, ShCid is less efficient than the simplest ShC and it seems that, despite the additional complexity, no performance gains are generally achieved. However, when combined with texture, the ShCid with object-oriented progressive depth slicing is experimentally shown to attain competitive performance.

Table 4.3: Depth map classification with feature normalization.

Descriptor type	Slicing methods			
	1	2	3	4
Shape context	16.84%	15.30	24.58	18.84%
Shape context with inner distance	17.34%	16.99%	20.85%	19.52%

4.4.4 Feature normalization

Feature normalization, discussed in Section 4.2.1, is trivial procedure that has a significant impact on performance. As shown in Table 4.3, due to the less complex and more efficient codebook training, when normalized features are used performance improves in comparison to that reported in Table 4.1. The use of normalized features results in more accurate sparse coding and facilitates the subsequent SVM-based classification. The above provide evidence feature normalization is beneficial for recognition performance.

4.4.5 The combination of texture and depth

In order to assess the performance of our system for image classification, we combined our depth-based classification approach with the well established SPM method [2], which is one of the most efficient texture-based image classification methods. The objective in our case was to use depth information in order to improve the performance of systems that rely only on texture. A practical consideration is that both SPM and our method result to vectors of high dimensionality, a fact that may be preventing efficient classification. With our method, the increase in size is less than 5% of the original size of SPM vector.

Experimental results using our method, combining texture and depth, one reported in Table 4.4 for several combinations between depth features and slicing methods. Results are also reported for texture-only systems in order to allow the assessment of the advantages gained from the use of depth information. As seen, the depth slicing approach affects classification results. The best result with shape context feature without inner distance enhancement is achieved in combination with the

object oriented tomographic depth slicing method. This result narrowly outperforms the texture-based SPM and highlight the advantages of using depth information for image classification. However, when using the shape context features with inner distance, system performance improves further and becomes clearly superior to that of the texture-only systems.

Table 4.4: Image classification using feature combination.

Texture description	Depth description	Slicing method	Output
SIFT-BoF	-	-	30.94%
SIFT-SPM	-	-	39.23%
SIFT-SPM	ShC-BoF	1	39.02%
SIFT-SPM	ShC-BoF	2	32.56%
SIFT-SPM	ShC-BoF	3	39.82%
SIFT-SPM	ShC-BoF	4	38.93%
SIFT-SPM	ShC_InDist-BoF	1	39.66%
SIFT-SPM	ShC_InDist-BoF	2	40.88%
SIFT-SPM	ShC_InDist-BoF	3	40.67%
SIFT-SPM	ShC_InDist-BoF	4	42.49%

4.5 Comparison of 3D image classification methodologies

In this chapter we present a comparison between our novel methodology the competing 3D method in [8].

4.5.1 Experimental setup

The experimental framework regarding the following experiments was proposed in [108] and is similar to the one used in [8]. In our experiments, each examined method is assessed using the dictionary training methodology proposed in each respective work ([8,16,17,43]). However, all examined methods compute a codebook comprised of 1000 codewords, for each examined feature extraction method. In all experiments, we use a three-level pyramid with 14 pyramid scales (1 scale, 4 scales, 9 scales) in

total. For the classification stage, we use the training/testing protocol of Section 4.4.1 which, unlike the framework proposed in [8], uses more testing than training images. We randomly choose two images per class for training and use all the remaining images for testing.

4.5.2 Comparison between methodologies

As seen in Table 4.5, our proposed methodology is compared with the competing method in [8] in two 3D imaging datasets [7, 43]. The results on BTDI and RGBD dataset show that our method outperforms most methods proposed in [8] except the combination of all features. However, our methods exhibit advantages against the proposed features of [8] in respect to computational complexity.

In [10], authors admit that the computation of kernel descriptors are generally expensive. Their most computationally intense shape kernel features need about four seconds per image computed by using MATLAB. Furthermore, gradients kernel features take 1.5 seconds using identical settings with SIFT features which only need 0.4 seconds. Moreover, all image representations computed from each feature are combined together forming a long image representation method. The dimension of the final representation vector proposed in [8] is the length of the encoding vector multiplied by the number of pyramid scale vectors multiplied by the number of used features. So for BTDI, the method in [8] uses $1000 \times 14 \times 5 = 70000$ dimensional image representation vector. The RGBD dataset, uses $1000 \times 14 \times 7 = 98000$ dimensional image representation vector.

Our proposed feature extraction methodologies using the tomographic and progressive slicing takes 0.19 seconds, using identical settings and dataset with those used to compute the features in [10]. Feature extraction using our novel object-oriented slicing takes 0.22 seconds to compute using MATLAB. Furthermore, our novel methodology uses 1000×14 dimensional vectors for texture and 1000-dimensional vectors for depth resulting in a 15000-dimensional vector. Our proposed method uses smaller representation vectors providing competitive results.

Table 4.5: Results of BTDI and RGBD dataset (± standard deviation).

Features	**BTDI**		**RGBD**	
SIFT + ShC (sl.method 1)	48.67±6.17	[43]	35.59±2.84	[43]
SIFT + ShC(sl.method 2)	46.64±5.39	[43]	**38.03±3.09**	[43]
SIFT + ShC (sl.method 3)	50.68±3.83	[43]	34.19±2.86	[43]
SIFT + ShC (sl.method 4)	51.72±4.30	[43]	37.00±3.15	[43]
SIFT + ShCid (sl.method 1)	52.49±4.41	[43]	34.45±2.75	[43]
SIFT + ShCid (sl.method 2)	47.93±4.98	[43]	33.02±2.82	[43]
SIFT + ShCid (sl.method 3)	**57.17±5.30**	[43]	33.98±2.81	[43]
SIFT + ShCid (sl.method 4)	51.92±6.41	[43]	33.96±2.99	[43]
RGB Gradient KDES	57.01±5.15	[8]	34.69±2.86	[8]
RGB LBP KDES	54.88±4.88	[8]	32.44±2.60	[8]
RGB Normalized Color descr. KDES	47.04±4.97	[8]	28.11±2.89	[8]
Depth Gradient KDES	43.88±4.91	[8]	25.64±2.81	[8]
Depth LBP KDES	41.72±6.01	[8]	23.67±2.64	[8]
Point cloud Size KDES	-		29.94±2.88	[8]
Point cloud Normal KDES	-		27.18±2.29	[8]
Feature combination	**62.26±5.07**	[8]	**47.80±2.77**	[8]

4.6 Conclusion

In this chapter, we presented our method which introduces novel feature extraction techniques from depth maps. In order to showcase our feature extraction method we also introduced a 3D image classification system combining texture and depth image features. Depth features create individual image representations combined with conventional texture representations. The resultant method was tested on two 3D image database that contain images captured under a variety of conditions. The proposed experimental framework creates challenging conditions, providing solid conclusions from every examined method. Our proposed methodology achieved excellent results compared to other competing methodologies. Advantages of our method are:

- The novel feature extraction method that can be easily implemented on multiple types of depth maps.

- Simple, modular 3D image classification architecture for easy implementation and future development.

- Low computational cost, in comparison with other competing methods.
- Overall, good performance in classification rates in comparison with more complex methodologies.

Future improvements for this method include an optimized depth representation. We plan to represent depth using the SPM image representation technique which captures the spatial information of the extracted depth features. Furthermore, a better representation can be achieved using our novel representation presented in Chapter 6. We consider optimizing vector dimensionality and scale number for each representation pyramid thus affecting recognition performance. Moreover, depth feature design can be improved by using deep learning methodologies which are capable to provide better depth description.

This novel method was the beginning of our research on depth feature design and fusion of feature representations for better 3D image classification performance. This work has further motivated us to study relations between representation and classification, leading to two novel methodologies on classification and depth-driven image representations.

Chapter 5

Dictionary training using relationships derived from depth

Recent methods for 3D classification [7–10,43,44] utilize depth information in order to achieve more accurate recognition. These methods are limited in the extraction and description of physical depth information. As a result, information such as context relationship and content representation described from depth are not exploited.

Dictionary training methods play a significant part on the description of extracted features using feature encoded methodologies. Dictionaries, when used with encoding methods, describe each feature as a combination of codewords consisting the dictionary thus solving the discriminatory problem of feature similarity. More information about the contribution of dictionaries in image classification systems can be seen in Chapter 2.2.

A popular dictionary construction method for 3D image classification is the k-means algorithm [7, 8, 43]. However, most recent approaches in image classification are using more complicated dictionary learning techniques such as K-Singular Value Decomposition (SVD) [9, 45] and Non-negative Matrix Factorization (NMF) [49, 109, 110] leading to better classification performance.

In [47], a graph regularized NMF is presented which regulates approximation by taking into account the relationship between dictionary training features in the feature space. The relationship of features is determined by means of their similarity,

which can be measured using their pairwise distances in the feature space. The disadvantage of this approach is that by randomly using selected features for dictionary training, without regard to the context from which these features were extracted, leads to less discriminative feature encoding and less efficient classification. As image context, we mean regions separating an image with respect to its content, and with no regard to features extracted from the region. The importance of image context was also presented in [111], where a comparative evaluation of context-based techniques for semantic image analysis was presented. Furthermore, context-based techniques have shown significant improvements in image classification [112] and complex image queries [113].

In Fig. 5-1, the above disadvantage is exemplified using two images from different classes. The selected feature in the right image is similar, i.e., has small distance in the feature space, with four other image features from the left image. However, not all similar features come from the same context. This means that features that exhibit similarity in the feature space may not correspond to similar or relevant regions in their respective images. If those features are included in a training set, dictionary training may be adversely affected due to the fact that features from different regions are mixed in order to form a codeword.

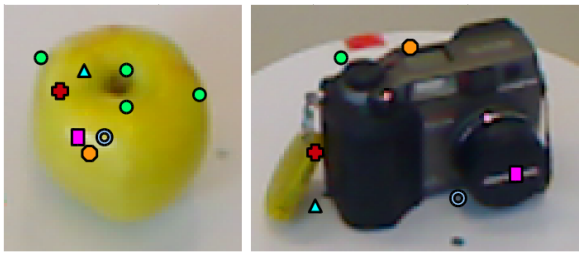


Figure 5-1: The pairs of features with the shortest distances in the feature space are presented by multi-color shapes. Some pairs exhibit short distances but no context similarity and, therefore, can be misleading for dictionary learning.

In this chapter, we present a novel dictionary training that integrates the relation of image context as described from depth information to improve dictionary learning. We propose a method that uses depth information in order to achieve more efficient dictionary training based on graph-regularized non-negative matrix factor-

ization (GNMF) [47]. In order to train dictionaries, we reformulate the similarity constraint of [47] using feature graphs that are constructed based on contexts. In this way, only the closest training features coming *from the same context* are considered to be related. This is made possible by partitioning each image into regions (contexts) by assinging a *context label* to each extracted feature. Then, we use these context labels to create a *context relationship matrix* that shows the relationship between features from the same context. This approach can be directly applied to 3D images, where contexts can be defined based on depth information.

The rest of the paper is organized as follows. Related work is presented in Section 5.1. Section 5.2, describes the purpose of context partitioning. In Section 5.3, the proposed methodology is formulated. The experimental methodology and results are shown in Section 5.4. Finally, conclusions are drawn in Section 5.5.

5.1 Review of NMF and GNMF

Non-negative Matrix Factorization (NMF) [110] is used in many applications. It analyses a given non-negative dictionary feature matrix $X = [x_1, x_2, \ldots, x_N] \in \mathbb{R}^{P \times N}$, where N is the number of P-dimensional feature vectors, into two non-negative matrices. The product of the resultant two non-negative matrices U and Z is a fine approximation of the training data matrix X, i.e,

$$X \approx UZ^T \tag{5.1}$$

where $U = [u_{pq}] \in \mathbb{R}^{P \times Q}$ and $Z = [z_{nq}] \in \mathbb{R}^{N \times Q}$. Q is the number of codewords and p, n are row indexes for those matrices. In NMF, matrix U is used as dictionary and Z as the encoded feature vectors.

In Graph regularized NMF (GNMF) [47], a graph is used in order to improve the efficiency of the NMF representation. The nearest neighbor graph is presented by a Laplacian graph adding a regularization constraint from the pairwise distances of features inside the dictionary training set.

$$L = W - D \tag{5.2}$$

where W is a $N \times N$ distance matrix and D a diagonal matrix with elements $d_{ii} = \sum_{l=1}^{N} w_{il}$. The geometric constraint using the Laplacian graph L is defined as:

$$Tr(Z^T LZ) = Tr(Z^T DZ) - Tr(Z^T WZ) \tag{5.3}$$

where $Tr(\cdot)$ denotes the trace of a matrix. By applying the constraint of eq. (5.3) into eq. (5.1), the GNMF approximation is formulated based on the following objective minimization function:

$$O = \min_{U,Z} \|X - UZ^T\| + \lambda Tr(Z^T LZ) \tag{5.4}$$

It should be noted that when $\lambda = 0$ then GNMF degenerates to conventional NMF.

5.2 Assigning context labels to features

As described in the introductory part, similar features do not always come from the same or similar parts of images. As a result, in a set of randomly collected features, similar features may describe different content due to feature discrimination problems. When two or more such features are used in a dictionary training set, the computed codewords lead to non discriminative image representation resulting in poor classification results.

In [47], attempts to calculate dictionaries that ensure discriminant encoded features are based strictly on the similarity measure of distance. However, this leads to falsely similar features as presented in Fig. 5-1.

In the proposed method, we determine the relationship between training features by context before dictionary training. This will lead to a relationship matrix that provide context similarity in combination with the distance matrix proposed by the conventional GNMF method. To this end, we partition images into regions.

This process assigns a context label to each extracted feature from every image in a predefined manner. Since image features are extracted via dense grids, we divide the image into blocks that correspond to feature patches. The block value is computed as the mean value of image characteristics, i.e, depth, color and their combinations, inside that block.

Region creation for each image can be achieved using scene understanding methodologies [50,114–117] based on partitioning techniques. These can partition each image depending on information such as depth for 3D images. In our method we use the blocks as vertices of an undirected graph, which is partitioned in K regions by solving the ℓ-GP problem [118]. As a result, each feature is assigned with a *context label* c showing the region from which it was extracted. An example of region labeling based on depth is shown in Fig. 5-2, where context labels are assigned according to the image characteristic values of the respective contexts. It must be noted that in order to guarantee the integrity of context labels, we first normalize all image information and assign the context labels after a value-based context sorting. So the context region with the greatest value is always denoted with the first context label, achieving a current label assignment for each examined image.

5.3 Context-Adaptive Graph regularized Nonnegative Matrix Factorization(CA-GNMF)

5.3.1 Context mapping

In [47], the similarity among N dictionary training features is captured in the distance matrix W, which is used for the construction of D and L in eq. (5.2). Matrix W is a $N \times N$ matrix that includes pairwise distances between all features in X. Finally, for each feature x_n only κ closest distances are kept creating a κ-nearest neighbor distance matrix. It must be noted that these features were randomly collected from different images in order to create a dictionary training feature set.

Contexts within images can be represented as regions, with the simplest example

being a two-region representation of foreground and background, i.e, $K = 2$. In 3D images, different depth levels can be used in order to define more regions. Due to the partitioning of images into K regions (contexts), the context from which each feature originates is known. Representation using more than two regions, i.e, $K > 2$, offers additional information about image content. For a set of N randomly selected image features X for dictionary training, we define the set of the respective context labels, denoted as $C = \{c_1, c_2, \ldots, c_N\}$, where each context label c_i is a value between 1 and K.

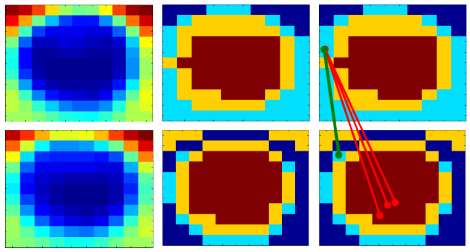


Figure 5-2: Partitioning of image blocks with graph partitioning. Each row shows an example, with feature similarities shown using lines in the last column.

The proposed training method regards features as similar only when they originate from similar context, as shown in Fig. 5-2. Therefore, feature vector similarity is assessed not only through a conventional distance metric but also by taking into account the context from which features are extracted. If features originate from different contexts, i.e., they have different context label, they are assumed to have zero similarity.

In order to formulate such relations among all training features, we create a *context relationship map* M. Map M is a $N \times N$ matrix that indicates context similarity between extracted features. The $(i,j)_{th}$ element m_{ij} of M indicates whether the context labels c_i, c_j of the i_{th} and j_{th} features in the training set are the same:

$$m_{ij} = \begin{cases} 1, & \text{if } c_i = c_j \\ 0, & \text{otherwise} \end{cases} \tag{5.5}$$

The context relationship map M is taken into account in combination with distance matrix W in order to create *context-weight matrix* B. Therefore, B is a distance

matrix with elements b_{ij} that are the pairwise distances of training feature pairs multiplied by their respective context relationship element m_{ij}, i.e., $b_{ij} = w_{ij} \cdot m_i$ for $i = 1 \dots N,\ j = 1 \dots N$.

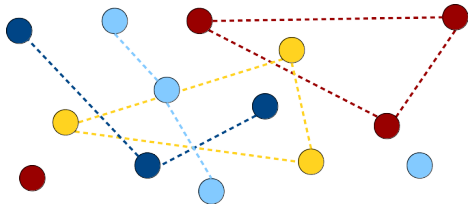


Figure 5-3: Construction of context-weight matrix B. Features from the same context (shown with the same color) are grouped together. Features from different contexts are not grouped even if they exhibit short distances in the feature space.

5.3.2 Implementation of context constraint on GNMF

Matrix B defines a graph with nodes that represent training features vectors. Connections between feature pairs with small similarity are disregarded. As in [47], we keep only the κ-nearest neighbors of matrix B for each of its comprising feature vectors. In this way, the new content similarity graph is converted into a nearest neighbor graph. The context-weight matrix B can contribute to more discriminative dictionaries and image representations, as well as improved classification rates. This can be achieved by means of introducing Laplacian matrix L_B, derived from B, which will serve as a constraint applied in the conventional NMF formulation. Laplacian matrix is defined as:

$$L_B = B - D_B \tag{5.6}$$

where D_B is a diagonal matrix with elements $d_{B_{jj}} = \sum_{l=1}^{N} w_{jl}$. By using L_B in eq. (5.3), the new regularizing term becomes:

$$Tr(Z^T L_B Z) = Tr(Z^T D_B Z) - Tr(Z^T B Z) \tag{5.7}$$

which can now be used in eq. (5.4) yielding a new objective function:

$$O = \min_{U,Z} \|X - UZ^T\| + \lambda Tr(Z^T L_B Z) \tag{5.8}$$

Using eq. (5.8) our proposed method for dictionary training takes into account not only the distance between feature vectors but also considers the context from which these features have been extracted. Therefore, features that would otherwise be considered similar and would be used together, are now seen as different and they contribute differently to dictionary construction. My proposed method minimizes O in eq. (5.8) through an iterative process based on the following update rules [47]:

$$u_{iq} \leftarrow u_{iq} \frac{(XU)_{iq}}{(UZ^T Z)_{iq}} \tag{5.9}$$

$$z_{jq} \leftarrow z_{jq} \frac{(X^T U + \lambda BZ)_{jq}}{(ZU^T U + \lambda D_B Z)_{jq}} \tag{5.10}$$

where $p = 1, \ldots, P$, $q = 1, \ldots, Q$ and $i = 1, \ldots, N$. The proposed method constructs dictionaries that improve the discriminatory capacity of the resultant image representation. Henceforth, we call the resultant method *Context-Adaptive Graph regularized Non-negative Matrix Factorization* (CA-GNMF). As will be seen in the experimental assessment section, CA-GNMF yields great improvements in classification rates on two 3D image classification datasets.

5.3.3 Classification algorithms

The discriminative power of an image classification system also depends on classification algorithm. Methodologies in [8, 16, 17, 43] use the Support Vector Machine (SVM) [56]. This combination is claimed to be among the most suitable because SVM is very efficient in dealing with long SPM representation vectors. A more recent classification method [119] delivers improvements by classifying each representation scale separately and reaching a final decision based on all partial decisions. In the ensuing experimental assessment we use the classification approach in [119], presented in Chapter 7, in combination with our dictionary training method.

5.4 Experimental results

5.4.1 Experimental framework

In our experimental assessment, we used the datasets and framework presented in [119]. Unlike [7,8], this framework uses more testing than training objects. In particular, two images per class were used for training while the rest were used for testing. Training images were randomly selected and used in every experiment. The pyramid architecture used for image classification was the same in all examined methods and the representation vectors comprised of 14 scale vectors, as proposed in [8].

All competing dictionaries comprised 1000 codeword vectors, as proposed in [8]. All compared methods were based on the same randomly selected image features for dictionary training. For the construction of distance matrix W and context-weight matrix B, in GNMF and CA-GNMF methods respectively, we use the five closest features per training feature. All experiments use the same dictionary training feature set.

5.4.2 Context-adaptive partitioning strategy

We conduct an experiment in order to determine which image attributes (or their combinations) yield the best context labeling and, as a result, lead to more discriminative dictionaries and better classification for both datasets. To this end, the same dictionary training feature set is used with five update iterations across all experiments. Multiple values are used for parameter K, which denotes the number of regions each image is partitioned. When $K = 2$, the resultant contexts essentially indicate the separation between foreground and background. However, when the number of regions increases, we observe that the separation also gains a spatial layout, as seen in Fig 5-2. From Table 5.1, we conclude that overall performance improves when more that two regions are used, i.e, $K > 2$.

Furthermore, as seen in Table 5.1, the proposed CA-GNMF performs best when the context-aware constraint in eq. (5.8) is created from regions formed by the com-

Table 5.1: Experiments on context partitioning image data. Comparison between all image data and their combinations in order to achieve a more discriminative dictionary.

K	Depth	Color	Combination
2	87.37±2.69	87.37±2.16	87.83±2.20
3	87.53±2.70	87.88±2.14	87.83±2.76
4	87.63±2.82	87.37±2.51	87.93±2.61
5	87.73±2.52	88.18±2.26	88.13±2.35
6	88.05±2.85	87.73±2.88	87.78±2.18
7	87.83±2.52	87.88±2.76	87.98±2.76
8	88.23±3.00	87.63±2.52	87.83±2.75

Best	**K = 6**	K = 5	K = 5
	89.94±2.91	89.19±2.72	88.13±2.35

bination of color and depth attributes. However, the optimized results for each experiment show that depth is the best image characteristic for describing context similarity. Exploiting the findings of these experiments, we use depth information for the extraction of context regions throughout our experimental assessment.

5.4.3 Comparison of encoding methods and dictionary training methodologies

In order to assess the benefits of the proposed methodology against competing dictionary learning techniques, we conduct a series of experiments. Table 5.2 demonstrates the combination of different dictionary training methods with feature encoding methods. The examined feature encoding methods are the Linear Spatial Pyramid (ScSPM) [17] and the Pyramid Efficient Match Kernels (EMKSPM) [16]. The competing dictionary training methods are k-means [119], K-SVD [45], NMF [110], GNMF [47] and the proposed CA-GNMF. All dictionaries were calculated from SIFT image features [1] using the framework presented in Section 5.4.1.

Table 5.2 shows that the ScSPM encoding method combined with every tested dictionary method outperforms the EMKSPM. For this reason, we use ScSPM through-

Table 5.2: Comparison between our proposed CA-GNMF method with other competing dictionary training methodologies using the RGB-D database.

Dictionary meth.	ScSPM	EMKSPM
K-means	86.26±2.15	62.63±2.44
K-SVD	86.31±2.83	63.08±2.37
NMF	87.67±2.26	66.16±2.26
GNMF	88.28±2.13	66.11±2.20
CA-GNMF	**89.94±2.91**	**66.47±3.16**

out our experimental procedure. In addition, this experiment showed the capabilities of our method when describing SIFT features compared to competing methodologies. However, in order to draw solid conclusions about the effectiveness of our methodology, we conducted experiments using several features. These are presented in the ensuing section.

5.4.4 Comparisons of dictionary training methodologies implemented on multiple image features

We also conducted a comparison of our CA-GNMF against the aforementioned competing dictionary methods for a number of features. For our comparison, we used the SIFT feature and the kernel descriptors presented in [8]. In order to use the features of [8] we had to replace negative values of feature elements with zero elements, thus creating a sparse version of those feature vectors. Despite that, the 3D point cloud features of [8] do not provide stable results when non-negativity is enforced and for this reason those features are not used in the following comparison. Classification performance using the RGB-D dataset is presented in Table 5.3. Our proposed methodology outperforms regular NMF and GNMF regardless of the feature used. However, for some features our method is less efficient than the k-means and K-SVD.

As seen in Table 5.3, not all features are complemented by the proposed dictionary learning method. This is a downside of the enforced non-negativity on the examined features. However, our proposed method still manages the second best classification

performance when used with those features. This shows the discriminatory capabilities of our proposed dictionary method, even when used with non-optimal features, and the necessity of introducing new features for 3D image categorization.

Table 5.3: Comparing the proposed CA-GNMF methodology to k-means for a number of features. Results on two 3D datasets are shown.

Feature	K-means	K-SVD	NMF	GNMF	CA-GNMF
SIFT	86.26±2.15	86.31±2.83	87.67±2.26	88.28±2.13	**89.94±2.91**
RGB Gradient KDES	90.40±2.14	91.41±2.05	90.76±1.72	90.86±1.72	**91.47±1.89**
RGB LBP KDES	89.90±2.64	91.87±088	92.32±1.91	92.27±1.30	**92.42±2.21**
RGB Normalized Color descr. KDES	**93.99±1.84**	92.98±1.68	92.58±1.39	92.37±2.17	92.63±0.94
Depth Gradient KDES	84.75±3.15	**88.84±1.97**	87.07±2.48	87.07±1.33	87.79±1.28
Depth LBP KDES	76.52±2.78	**84.80±1.65**	81.87±1.99	82.82±2.30	84.04±0.79

5.5 Conclusion

We introduced a novel dictionary training method for classification of 3D images. In order to design efficient dictionaries, we assess the similarity of features originating from similar content. Each image was partitioned into regions (contexts) based on image characteristics such as depth. This procedure provides the context from which each feature was extracted. Feature context labels were subsequently used in dictionary training. The proposed methodology was experimentally shown to achieve excellent results. Summarizing the contributions of this work:

- Depth information is applied to describe the contextual relations between extracted features.
- Dictionary learning with context supervision eliminates feature discrimination problems inside a dictionary training set.
- Our proposed dictionary method yields top-performing results when combined with suitable image features.

Our future plans include the use of this dictionary training method on an optimized framework. Our proposed framework will include experiments with suitable features,

optimal dictionary size and more 3D imaging datasets. All the above will provide solid conclusions about the effectiveness of our proposed dictionary method. Furthermore, the method can be used in combination with our other proposed methodologies presented in Chapters 4,7 and 6. The combination of all our presented methodologies in this thesis are expected to yield even better classification results in database benchmarks, but also in real-life applications.

Chapter 6

Using depth information for the creation of image pyramid representation

The most common image representation method among 2D image classification methodologies is the Spatial Pyramid Matching kernel (SPM) [2]. Spatial pyramid is a multilevel hierarchical representation based on the bag of features model (BoF) [66]. This hierarchical representation is based on the partitioning of an image in rectangular regions, termed *scales*. The final image representation captures discriminatory image information by combining the representation histograms associated with each scale into a long concatenated vector. A visualization of such pyramid is shown in the upper part of Fig 6-1.

Representation vectors derived from spatial pyramid are usually classified by means of a support vector machine (SVM). The ability of the SVM to handle effectively high dimensional vectors makes it suitable for combination with pyramid image representation. For this reason, this combination is the most widely used in two and 3D image classification, achieving good classification rates.

Although 2D systems have to rely on texture and color intensities, 3D images have depth features that can additionally contribute to the classification task. In this chapter, we present a novel 3D image pyramid representation for 3D images [108].

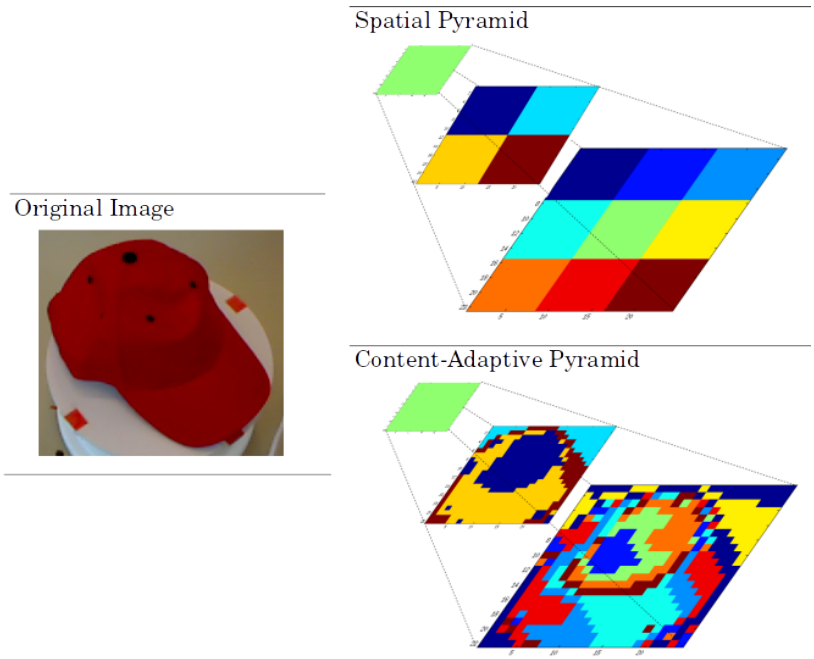


Figure 6-1: Differences between the regular spatial pyramid and the proposed content-adaptive pyramid representation. Both pyramids represent the same image, with the proposed representation being constructed using regions.

Our work focuses on an adaptive pyramid architecture that can efficiently represent the content of each image. Unlike the SPM representation [2], the proposed method builds a pyramid representation that forms a hierarchy of non-rectangular regions. Similar to scene understanding methodologies [50, 114–117], we partition the image in small blocks and we perform graph partitioning on those blocks using content attributes like color and depth. In this way, we define arbitrarily-shaped spatial scales on the pyramid representation according to the image content. Using this process, the proposed method forms a representation that is more intrinsically associated with image content. This is shown in the lower part of Fig 6-1.

The contributions of the present chapter are:

- A pyramid image representation constructed by using non-rectangular spatial regions that adapt to the image content.

- An investigation that experimentally shows that depth is the most suitable characteristic to guide the construction of the proposed pyramid representation.

- The combination of our pyramid representation with the classification method in [119], which results in a system that achieves state-of-the-art results in two 3D image classification datasets.

The rest of the chapter is organized as follows. In Section 6.1, we review the current 3D image classification methods. Section 6.2 describes the extraction of region indexes based on graph partitioning. In Section 6.3, we demonstrate the construction of our novel content-adaptive pyramid image representation. In Section 6.4, a new 3D image classification framework is presented, followed by a detailed experimental assessment. Conclusions are drawn in Section 6.5.

6.1 Review of current 3D image classification methodologies

A spatial pyramid is a multi-level hierarchy of rectangular regions. Each pyramid level consists of rectangular regions called *scales* or *spatial bins* [2]. The number of levels (and scales) is constant for every image. The most widely used architecture is a three-level pyramid [2, 10, 11, 13–30, 43]. In this construction, the first level has one scale, the second has four scales, and the last level has 16 scales. Features from each scale are encoded using a codebook and are used for calculating a separate representation histogram for each scale. The concatenation of those scale representation vectors forms the final image representation vector.

3D image classification methodologies follow [7–9, 43] the processes and techniques used in 2D image classification using the conventional SPM. Since most 3D categorization methodologies are comprised from multiple image features, SPM is used for the representation of each texture and depth feature. The most used architecture [7–9] is a three-level pyramid totaling 14 scales. The first level consists of one scale, the second has four and the last levels consists of nine scales. All 3D methodologies [7–9, 43] proposed the concatenation of each individual feature representation. This results to a long representation vector used as data for the classification algorithms.

The spatial pyramid method yields state-of-the-art results for 2D and 3D image classification. Its main disadvantage, however, lies in the fact that it does not take into account the spatial arrangement of content within an image. Implicitly, the SPM-based methodologies assume that the content of interest is located in the image center and there is little background clutter.

Solutions to this problem have been studied by methods in [22–30]. These methods use a detection or localization process in order to find the main object within an image. Some methods [24–30] focus on the object, by disregarding or underestimating encoded feature vectors extracted from the background. Other methods [22, 23] create class-specific feature encoding weighting creating more discriminative image representations. Although, they all result on using an SPM-based representation which somewhat treats the aforementioned spatial arrangement of each image.

In order to deal with the disadvantages of the spatial pyramid, arising from its rigid construction, we construct a new pyramid representation for 3D images, which adapts to the image content by forming arbitrarily-shaped hierarchical regions.

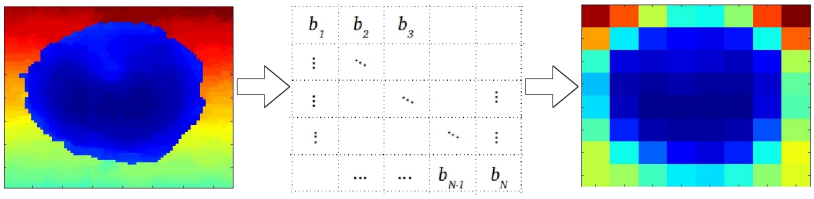


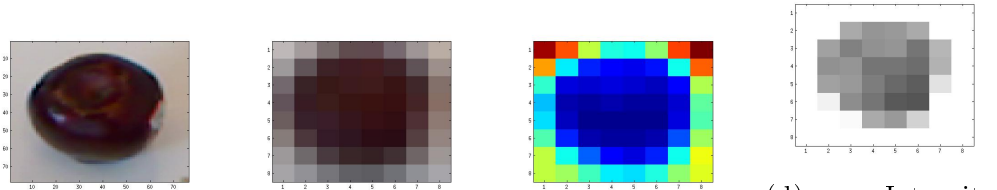
Figure 6-2: Representation of the block of characteristics

6.2 Graph Partitioning And Region Indexing

6.2.1 Image characteristics revealing image content

In SPM, each image is represented using encoded vectors, which correspond to features extracted over dense grids (also called as *spatial cells* [2]). Instead, scene understanding methods [114–117, 120], regionalize images according to local content by using algorithms such as those in [121, 122].

The proposed method uses a dense grid of rectangular blocks, studies the relation



(a) Original image (b) Color blocks (c) Depth blocks (d) Intensity blocks

Figure 6-3: (a) The original image from which *blocks of characteristics* are extracted. (b), (c), (d) characteristic blocks based on color, depth information, and intensity respectively.

between them and captures discriminative information. To achieve content aggregation, our method extracts simple characteristics from each block. Similar to the work in [117,123], *characteristic blocks* are constructed based on color, intensity, and depth information (Fig 6-2).

A characteristic block is represented by the mean value of information from the pixels inside that block. The color characteristic blocks are represented by the mean RGB values of the pixels in each block. An intensity block is represented as the mean intensity of the pixels within the block. Further, depth blocks are represented by the mean depth value extracted from depth maps. The set of these blocks are denoted as $B = \{b_1, b_2, ..., b_N\}$, where N is the number of blocks in the image (or the depth map). An example of such partitioning can be seen in Fig 6-2. In our analysis, color blocks are denoted as B_c, intensity blocks as B_g and depth blocks as B_d. An example of these characteristics can be seen in Fig 6-3.

Extracted blocks are grouped into spatial regions that are subsequently used for pyramid representation. In order to form connective regions, characteristic blocks are regarded as the vertices of an undirected graph. For the creation of graph vertices, each characteristic B_c, B_g or B_d can be used individually. Characteristics can also be combined in order to achieve better partitioning. The partitioning procedure is very crucial, mostly because it affects significantly the discriminative capacity of our image representation. An experimental assessment of characteristics and their combinations for the optimum representation will be presented in Section 6.4.2.

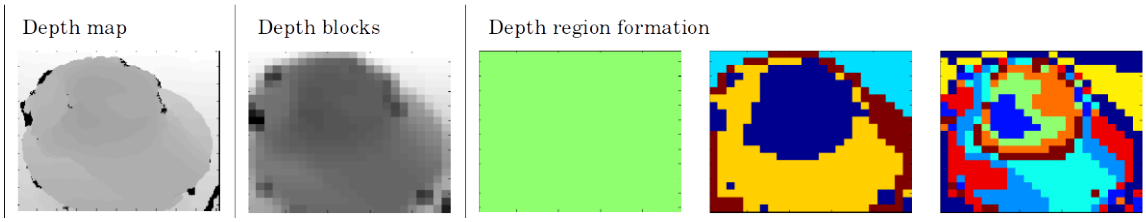


Figure 6-4: A depth map is transformed into depth characteristic blocks. Graph partiotioning of the previous characteristics lead to the creation of regions.

6.2.2 Graph of image content

As described above, the blocks are used in order to form a pyramid of regions, i.e., regions at different levels. We apply a simple classification algorithm that groups blocks into regions at different levels. Henceforth, these regions will be referred to as *scales*.

In order to form regions, we perform block clustering using graph-based partitioning, following the *ℓ-bounded graph partitioning problem* (ℓ-GP) presented in [118].

The ℓ-GP problem explains the partitioning of a graph with vertices $V = [v_1, v_2, ..., v_Q]$, where Q is the total number of vertices, into K regions. The purpose of this method is to find the k-th subset of V by minimizing the cost $C(V_k)$ with no more that ℓ vertices in a subset. The minimum value of ℓ is defined as $\ell = [Q/K]$. Each individual subset V_k corresponding to the k-th region is presented as a subset of V with $V_k = [v_1^k, v_2^k, ..., v_{Q_k}^k]$. The union of all K regions must be the original V. In order to determine the subsets the cost C of V_k must be minimized, which is defined as:

$$C(V_k) = \sum_{i \neq j} \sum_{u \in V_i, \bar{u} \in V_j} c(u, \bar{u}). \tag{6.1}$$

where u is an index indicating the u-th entry of V_k. The cost $c(u, \bar{u}) = 0$ for every edge and since the graph is undirected $c(u, \bar{u}) = c(\bar{u}, u)$, $\forall u,\ \bar{u} \in V$.

6.3 Content-Adaptive Pyramid Matching (CAPM)

6.3.1 Spectral clustering and region indexes

In order to construct the proposed content-adaptive pyramid, we label every block and form regions by grouping blocks in each pyramid level. The correspondence of each block to each individual scale is indicated through region labels, termed *region indexes*.

In conventional SPM, each pyramid has L levels. Each level l is partitioned into l^2, $l = 1, \ldots, L$, rectangularly shaped spatial scales. In our pyramid construction in order to form regions, we use the general formulation of the ℓ-GP problem, based on eq. (6.1), and we apply it on characteristic blocks. In this way, the graph vertices V contain the values of blocks B. The preferred number of block regions is $K_l = l^2$, i.e., the number of scales for each level is equal to the number of regions formed in each level. Scales are no longer rectangular but object-shaped and are determined using the preceding graph partitioning.

For an image yielding N feature vectors, we construct an $N \times N$ adjacency matrix A that consists of the pairwise distance between the values of each characteristic block of an image. We then compute at each level l, a $K_l \times K_l$ diagonal matrix D, the diagonal of which contains the K_l largest eigenvalues of A. We also compute matrix $E = [\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_N]$, the columns of which are the eigenvectors of A that correspond to the K_l largest eigenvalues. By using E we compute region indexes r, $r = [r_1^1, r_1^2, \ldots, r_L^N]$, of the n-th characteristic block in the l-th pyramid level. The assignment of region indexes is done by partitioning E into K_l sets $\Xi^l = [\xi_1^l, \xi_2^l, \ldots, \xi_{K_l}^l]$:

$$\min_{\Xi^l} \sum_{y=1}^{K_l} \sum_{\mathbf{e}_n \in \Xi_y^l} \|\mathbf{e}_n - \mu_y\|^2 \tag{6.2}$$

where μ_y are the centroids of the K_l eigenvector clusters determined using of k-means algorithm. The final step is to assign each region index r_l^n to their corresponding sets when $\forall \mathbf{e}_n \in \Xi_y^l : r_l^n = y$. Fig. 6-4 shows an example of region forming created from a depth map.

This partitioning method assigns labels to formed regions using an ordering procedure. The ordering of region labels takes place either in descending or ascending order without affecting the efficiency of the method.

6.3.2 Generating the content pyramid representation

As described above, a conventional spatial pyramid (SPM) is constructed based on rectangular spatial scales. In our proposed representation, however, scales are in the form of arbitrarily-shaped regions that adapt to image content. Henceforth, these will referred to as *adaptive scales*.

For an image represented by vector $\mathbf{p}$, which is the concatenation of scale representation vectors $\mathbf{s}$, our general implementation and simplification of the general SPM pyramid kernel $\kappa(\mathbf{p})$ is:

$$\kappa(\mathbf{p}) = \sum_{l=1}^{L} \sum_{r=1}^{l^2} \kappa(\mathbf{s}_{lr}) \tag{6.3}$$

where $\mathbf{s}_{lr}$ is the representation vector at the l-th pyramid level and r-th scale. The scale vector $\mathbf{s}_{lr}$ is the representation histogram of encoded extracted descriptors $\mathbf{z}$:

$$\mathbf{s}_{lr} = [|\mathbf{z}_{lr}^1|, |\mathbf{z}_{lr}^2|, \ldots, |\mathbf{z}_{lr}^{N_{lr}}|] \tag{6.4}$$

where N_{lr} denotes the number of encoded feature vectors within region r and [.] represents a histogram of the encoded vectors. In eq. (6.4), the region scale indexes r are determined by the preceding graph partitioning. If r is not used as a region indicator then the representation degenerates to normal SPM. This results in a representation dependent to the content of each image. Henceforth, we call this representation *content-adaptive pyramid* and the resultant method *content-adaptive pyramid matching* (CAPM).

6.3.3 Implementation on Linear Spatial Pyramid

The deployment of the proposed CAPM image representation in place of the SPM representation in [17] (ScSPM), requires that feature vector $\mathbf{x}_n$, from a set of feature vectors X, is encoded in the form of a sparse vector $\mathbf{z}_n$. The sparse feature vectors are computed by minimizing the l_1 norm using the feature-sign search algorithm [49]. The above feature encoding process is based on:

$$\min_{\mathbf{z}_n} \|\mathbf{x}_n - \mathbf{z}_n B\|_2^2 + \lambda |\mathbf{z}_n|_1 \tag{6.5}$$

where B is a $d \times W$ precomputed codebook, with d the dimensionality of the feature vector and W the total number of codewords, calculated using random feature vectors from each class. We use the pyramid kernel in eq. (6.3) and we apply the following changes on the scale vectors of eq. (6.4) by adding a *max-pooling* function [17].

$$\mathbf{s}_{lr} = \max[|\mathbf{z}_{lr}^1|, |\mathbf{z}_{lr}^2|, \ldots, |\mathbf{z}_{lr}^{N_{lr}}|] \tag{6.6}$$

6.3.4 Implementation on Pyramid Efficient Match Kernels

Efficient match kernels [8] describe the encoding of high dimensional feature vectors $\mathbf{x}_n$ through a projection to a low dimensional space. Firstly, the low dimensionality projection coefficients, i.e, the encoded feature vectors $\mathbf{z}_n$ are calculated as:

$$\mathbf{z}_n = (B^T B)^{-1}(B^T \mathbf{x}_n) \tag{6.7}$$

The encoded feature vectors provide the following local kernel:

$$k(x, y) = [B\mathbf{z}_x]^T [B\mathbf{z}_y] = \mathbf{k}_B(x)^T K_{BB}^{-1} \mathbf{k}_B(y) \tag{6.8}$$

where $\{\mathbf{k}_B\}_i = k(\mathbf{x}_n, \mathbf{b}_i)$ is a $W \times 1$ vector and $\{k_{BB}\}_{ij} = k(\mathbf{b}_i, \mathbf{b}_j)$ a $d \times d$ matrix. For $K_{BB}^{-1} = G^T G$ forming a local feature map:

$$\phi(\mathbf{x}_n) = G\mathbf{k}_B(\mathbf{x}_n) \tag{6.9}$$

the final full feature map is $\phi(X) = \frac{1}{|X|} G[\sum_{\mathbf{x}_n \in X} \mathbf{k}_B(\mathbf{x}_n)]$ Implementing the proposed pyramid representation on the above encoding, every scale vector is presented:

$$\mathbf{s}_{lr} = [|\phi(\mathbf{x}_{lr}^1)|, |\phi(\mathbf{x}_{lr}^2)|, \ldots, |\phi(\mathbf{x}_{lr}^{N_{lr}})|] \tag{6.10}$$

6.3.5 Classification

Pyramid methods presented so far in the literature are based on the concatenation of feature vectors extracted from each pyramid scale. This normally produces a long feature vector. The most widely used classification algorithm for such vectors is Support Vector Machines (SVM) [56]. This is due to the SVM's ability to efficiently classify multidimensional vectors such as those generated by pyramid image representation methods.

The SVM, when paired with conventional SPM, operates on multidimensional feature vectors without specific regard to the pyramid scales that comprise these vectors. This can have an adverse impact on the overall classification performance if discriminatory information is concentrated in a small number of scales. In order to overcome this problem, we used the proposed *Scale Based Support Vector Machine* (SBSVM) algorithm [119], described in Chapter 7, in which each scale vector was classified separately and contributed independently to the final decision.

The main difference between the scale-based and the conventional application of SVM lies in the fact that each image is no longer classified by a single decision taken based on a long concatenated representation vector but, instead, individual decisions based on each scale are combined. Specifically, each scale is processed by a linear SVM that was previously trained based on the individual scales from a random subset of images. The independent decisions reached using the scale-trained SVM are used in a voting process, in which the decision label that is in the majority is considered the image label.

The combination of the proposed representation with the SBSVM will be experimentally shown to achieve improved performance over the combination with the conventional application of SVM.

6.4 Experimental Evaluation

6.4.1 Experimental setup

Since our image representation is to be used with 3D images, we compare the proposed representation with the SPM [2] in the context of 3D image classification. Specifically, we use our representation in combination with Pyramid Efficient Match Kernels Over Kernel Descriptors [8] and Texture and Depth Sparse Representation Fusion [43], which are 3D classification methodologies that used SPM image representation. These methods were presented in detail in Section 6.1.

For the experimental evaluation of our method we adopt a framework similar to the one used in [8]. In our experiments, each examined method is assessed using the dictionary training methodology proposed in its respective work ([8, 16, 17, 43]). Dictionary training for all examined methods is used with a codebook comprising 1000 codewords. These codewords are computed from 20 feature vectors that are randomly chosen from each image from across each database.

In all our experiments we use a three-level pyramid with 14 pyramid scales in total. Specifically, the first level has one scale, the second level has four scales and the final pyramid level has nine scales.

In the classification stage, we use the training/testing protocol of Chapter4.4.1 which, unlike the framework proposed in [8], uses more testing than training images. We keep the same training setup for both SVM and SBSVM algorithms, i.e., in all experiments, we randomly choose two images per class for training and use all the remaining images for testing. Further, we randomly choose 10 different training/testing sets for the classification training/testing process. We repeat the above for 10 different trained dictionaries. As a result, we conduct 100 random experiments in total, which increases confidence in the validity of our results and conclusions.

Table 6.1: Classification efficiency for several characteristics (± standard deviation).

Feature	Pyramid	Guiding characteristic	BTDI database	RGB-D database
SIFT	ScCAPM	Color	96.29±2.70	94.82±1.46
SIFT	ScCAPM	Depth	**98.07±1.46**	**95.58±1.49**
SIFT	ScCAPM	Color + Depth	97.54±1.56	94.34±1.61
SIFT	ScCAPM	Grayscale	97.21±2.15	95.47±1.12
SIFT	ScCAPM	Grayscale + Depth	96.74±1.95	94.61±1.62
SIFT	ScCAPM	Grayscale + Depth (normalized)	96.31±2.00	95.10±1.42
SIFT	EMK CAPM	Color	79.61±4.59	69.29±2.64
SIFT	EMK CAPM	Depth	75.92±3.84	**70.01±3.01**
SIFT	EMK CAPM	Color + Depth	79.83±4.02	66.06±2.86
SIFT	EMK CAPM	Grayscale	**81.29±4.15**	69.08±2.98
SIFT	EMK CAPM	Grayscale + Depth	80.06±4.35	67.42±2.71
SIFT	EMK CAPM	Grayscale + Depth (normalized)	80.31±4.14	67.42±3.07

6.4.2 Suitability of characteristics to guide the representation.

As described in Section 6.2.2, characteristic blocks are used to capture the content of an image and guide the construction of the proposed CAPM representation. In order to determine the most suitable characteristic, we test several characteristics for guiding the region formation of the proposed pyramid representation. To this end, we use our proposed CAPM and the classification algorithm in [119] with the systems in [16, 17]. In order to reach reliable conclusions, we conduct experiments on BTDI and RGB-D 3D datasets. The characteristics that we test are listed in Table 6.1 and include color, texture, depth and their combinations. As seen in Table 6.1, depth is in most cases the best-performing characteristic to guide the construction of the pyramid.

Since depth performed best in most experiments, it will be subsequently used as the characteristic that guides the pyramid construction in all our experiments. As will be seen in the ensuing section, depth-adaptive pyramid representations yield superior results on the examined datasets.

6.4.3 Evaluation of CAPM representation

In order to assess the efficiency of our proposed pyramid representation, we compare it against the standard SPM. Our first experiment assesses the performance of our

Table 6.2: Results of BTDI dataset (± standard deviation). Depth was used for the construction of the content-adaptive pyramid.

	SVM			SBSVM		
Features	**SPM**		**CAPM**	**SPM**		**CAPM**
SIFT + ShC (sl.method 1)	48.67±6.17	[43]	38.17±5.93	86.06±3.11		**91.42±1.78**
SIFT + ShC(sl.method 2)	46.64±5.39	[43]	40.11±6.61	85.71±2.96		**91.80±1.21**
SIFT + ShC (sl.method 3)	50.68±3.83	[43]	41.95±6.42	85.74±3.41		**91.01±1.84**
SIFT + ShC (sl.method 4)	51.72±4.30	[43]	40.57±6.94	85.85±3.26		**91.44±1.57**
SIFT + ShCid (sl.method 1)	52.49±4.41	[43]	40.87±5.40	86.29±3.12	[119]	**91.45±1.50**
SIFT + ShCid (sl.method 2)	47.93±4.98	[43]	38.33±5.43	86.37±3.05	[119]	**91.00±2.33**
SIFT + ShCid (sl.method 3)	57.17±5.30	[43]	41.60±6.23	86.24±3.37	[119]	**91.00±1.99**
SIFT + ShCid (sl.method 4)	51.92±6.41	[43]	42.03±8.40	85.92±3.14	[119]	**91.39±1.82**
RGB Gradient KDES	57.01±5.15	[8]	43.56±4.38	91.79±2.93	[119]	**95.15±2.47**
RGB LBP KDES	54.88±4.88	[8]	44.15±5.03	93.61±2.92	[119]	**97.71±1.69**
RGB Norm. Color descr. KDES	47.04±4.97	[8]	37.97±5.06	93.25±2.58	[119]	**96.53±2.09**
Depth Gradient KDES	43.88±4.91	[8]	30.21±4.49	61.15±4.59	[119]	**72.76±4.64**
Depth LBP KDES	41.72±6.01	[8]	29.49±4.62	73.08±4.33	[119]	**91.37±3.02**
Point cloud Size KDES	-		-	-		-
Point cloud Normal KDES	-		-	-		-
Feature combination	62.26±5.07	[8]	52.06±4.79	97.54±1.81	[119]	**99.24±0.93**

CAPM representation in conjunction with SVM classification. When our CAPM representation is used in place of the SPM representation, the grouping of the encoded features takes place in a content-adaptive rather than a predefined spatial manner. As seen in Table 6.2 and Table 6.3, every feature used with CAPM/SVM performs poorly against the conventional SPM/SVM combination. This is because the CAPM scales adapt to the content and, therefore, discriminative information tends to concentrate on a smaller number of scale vectors. In this way, the concatenation of CAPM scale vectors in a long representation vector and has an adverse impact on performance.

To address the aforementioned problem we use the SBSVM classification algorithm in [119]. The SBSVM exploits the discriminatory capabilities of the proposed CAPM representation by means of its scale-based decision making. As seen in Tables 6.2 and 6.3, the CAPM representation when combined with SBSVM classification outperforms the conventional SPM representation, which demonstrates the advantage of our representation over SPM.

The results in Tables 6.2 and 6.3 show that the application of our methodology in the texture representation of [43] results in increased performance on both datasets. Also, the CAPM representation using the features in [8] improves the over-

Table 6.3: Results of RGB-D dataset (± standard deviation). Depth was used for the construction of the content-adaptive pyramid.

Features	SVM		SBSVM	
	SPM	**CAPM**	**SPM**	**CAPM**
SIFT + ShC (sl.method 1)	35.59±2.84 [43]	36.53±5.90	86.11±1.85	**88.23±1.73**
SIFT + ShC(sl.method 2)	38.03±3.09 [43]	36.61±5.56	86.17±1.85	**88.79±1.38**
SIFT + ShC (sl.method 3)	34.19±2.86 [43]	36.48±5.22	85.92±2.17	**88.59±1.44**
SIFT + ShC (sl.method 4)	37.00±3.15 [43]	35.45±4.62	85.96±1.98	**87.96±1.82**
SIFT + ShCid (sl.method 1)	34.45±2.75 [43]	37.12±4.71	85.78±2.14 [119]	**88.04±1.53**
SIFT + ShCid (sl.method 2)	33.02±2.82 [43]	35.35±5.78	86.11±1.85 [119]	**88.64±1.62**
SIFT + ShCid (sl.method 3)	33.98±2.81 [43]	35.83±4.94	85.33±2.13 [119]	**87.79±1.57**
SIFT + ShCid (sl.method 4)	33.96±2.99 [43]	34.58±5.38	86.06±1.91 [119]	**88.85±1.59**
RGB Gradient KDES	34.69±2.86 [8]	32.43±1.96	88.35±2.02 [119]	**92.48±1.47**
RGB LBP KDES	32.44±2.60 [8]	29.25±2.90	92.77±1.86 [119]	**96.49±1.36**
RGB Norm. Color descr. KDES	28.11±2.89 [8]	26.84±2.64	91.70±1.54 [119]	**93.86±1.53**
Depth Gradient KDES	25.64±2.81 [8]	22.80±2.29	78.79±2.55 [119]	**87.10±1.98**
Depth LBP KDES	23.67±2.64 [8]	19.44±2.56	85.91±2.03 [119]	**92.85±1.59**
Point cloud Size KDES	29.94±2.88 [8]	29.18±2.77	78.96±2.05 [119]	**80.08±2.31**
Point cloud Normal KDES	27.18±2.29 [8]	22.55±4.49	79.76±2.32 [119]	**83.39±2.42**
Feature combination	47.80±2.77 [8]	45.56±3.13	98.62±0.80 [119]	**99.18±0.60**

all performance compared to the SPM representation using the same features. The best performance, however, is achieved by the combination of CAPM representations based on multiple features in a joint feature vector, which provides state-of-the-art classification results on both datasets. Overall, the pairing of our proposed CAPM representation with SBSVM classification outperforms the conventional SPM/SVM and SPM/SBVM image classification approach.

Clearly, the application of CAPM constitutes a content-adaptive extension of the methods that are included in our comparison. The adaptation capacity of our proposed representation, forms coherent regions and separates content of interest from less relevant background content. In this way, the CAPM scales also separate regions of interest within particular classes.

6.4.4 Multi-view vs. Single viewpoint

In order to assess the categorization capabilities of multi-view methods when only one view is available we use the multi-view method in [8] as reference. That method was compared with the method in [119] in a scenario in which only one viewpoint was

Table 6.4: Multi-view vs. single-viewpoint methodologies on the RGB-D database. (± standard deviation).

	SPM / SVM				**CAPM / SBSVM**	
Features	Pyramid		Multiview	Single-view	Pyramid	Single-view
RGB Gradient KDES	EMK SPM	[8]	-	40.58±5.45	EMK CAPM	99.71±0.70
RGB LBP KDES	EMK SPM	[8]	-	39.47±5.32	EMK CAPM	99.98±0.20
RGB Norm. Color descr. KDES	EMK SPM	[8]	-	38.94±6.16	EMK CAPM	99.96±0.27
Depth Gradient KDES	EMK SPM	[8]	69.0±2.3	28.36±4.93	EMK CAPM	98.63±1.61
Depth LBP KDES	EMK SPM	[8]	66.3±1.3	26.28±4.38	EMK CAPM	99.92±0.38
Point cloud Size KDES	EMK SPM	[8]	60.0±3.3	37.94±5.42	EMK CAPM	94.00±2.58
Point cloud Normal KDES	EMK SPM	[8]	-	35.06±5.15	EMK CAPM	98.55±1.79
Feature combination	EMK SPM	[8]	86.5±2.1	59.02±5.54	EMK CAPM	**100.00±0.00**

available. Dictionary learning in [8] took place by using random features, extracted from multiple views. In the present work, we randomly chose *one view* for every object in every class from the RGB-D dataset, creating a single-viewpoint subset containing 300 RGB+D images. Additionally in [8], tests were conducted using one randomly chosen viewpoint image for testing while all remaining viewpoint images from the remaining objects in each class were used for training. In this way, the multi-view method essentially operates in a single-viewpoint framework. We applied this experimental procedure to the method in [8] and we also implemented our proposed CAPM representation in the same method using SBSVM.

The results from all experiments are shown in Table 6.4 along with their standard deviation. As seen, when only one viewpoint is available, the performance of the multi-view method in [8] drops significantly. As can be seen in the last column of Table 6.4, when using our proposed CAPM representation, the single-viewpoint method outperforms the multi-view method by a significant margin.

This experiment shows that multi-view methods are very relaint on the availability of several views and therefore, are not robust to a reduction in the number of available viewpoints. Our proposed representation, despite being used on a single-viewpoint framework, outperforms multi-view method approaches.

6.5 Conclusions

In this chapter, we introduced a novel image representation that is suitable for 3D image classification. Unlike most current approaches, our representation was not constructed by means of a fixed pyramid but, instead, relies on image content and uses content-adaptive scales rather than fixed rectangular scales. In addition to the new representation, we experimentally showed that depth information benefits the formation of regions within the pyramid structure. When coupled with suitable classification, our proposed methodology achieved state-of-the-art results on all examined 3D image datasets. The main contributions of this work are:

- This introduction of a novel pyramid representation that contributes to better recognition rates.
- Our novel pyramid design can provide better representation by taking into consideration the content of each image.
- Depth information is experimentally shown to be the most efficient content index for an image.

The experiments presented in this chapter are based on the experimental framework used from other methods (Section 6.4.1). These experimental settings provide an easy comparison between our proposed method and the SPM used in 3D methodologies [7–9,43]. However, these methods were optimized and presented in their proposed framework, including important parameters, such as pyramid architecture and scale vector dimensionality. We believe that the recognition rates achieved by our method can be improved by optimizing the representation using our proposed framework. Some indications about the efficiency of our proposed representation methodology can be seen in Chapter 8.3.5. The flexible architecture provided from our novel representation outperforms SPM when used with a deep convolutional neural network.

Furthermore, an interesting future work could include the implementation of our method in real-time processes with realistic data. This will allow us to draw conclusions about the robustness of our novel method in realistic applications.

Chapter 7

Classification of 2D and 3D Images Using Pyramid Scale Decision Voting

One of the most popular image representations for image classification is by means of Spatial Pyramid Matching (SPM) [2]. A spatial pyramid is a multi-level hierarchy of rectangular regions (called *scales*) that describe each image locally by using the Bag-of-features model (BoF) [66]. Each region is represented by a histogram formed by the encoded vectors of features extracted from that region. The concatenation of those scale vectors forms a long vector that represents the image.

The majority of 2D image classification methods [2, 10, 11, 13–30] and 3D methods [7–10, 43, 44] use the SPM as their representation vector. The most suitable classification algorithm for the SPM representation, which is also used in all aforementioned methodologies, is the Support Vector Machine (SVM) [56]. As described in Chapter 2.5, the SVM algorithm can effectively categorize data represented by high-dimensional vectors. That is also the case for 2D and 3D image categorization where images are described using representation vectors. However, SVM fails to correctly classify a large percentage of images, which implies that training may not be efficient. It also exhibits increasingly better performance for increasing amounts of training data while, in some cases, when limited training data is available, performance is significantly lower. Therefore, an interesting challenge is to try to improve classification performance by using a suitable training procedure in cases where available training

data does not lead to good classification performance or in case limited training data is available.

In this chapter, we present a novel method using individual representation scales for 2D and 3D image classification presented in [119]. This part of our research, studies the discrimination capabilities of individual scales coming from pyramid representation. We propose that *individual* pyramid scale vectors train an SVM in order to take into account the fact that some scale vectors of low discrimination power may have a negative effect on classification performance. Although our method performs best in the context of 3D image classification systems, it is also applicable to 2D image classification, which benefits from requiring fewer training samples to achieve similar performance.

The rest of the chapter is organized as follows. We present our scale-based classification method in Section 7.1. Experimental evaluation with detailed description of the experimental results for two and 3D datasets is presented in Section 7.2. Conclusions are drawn in Section 7.3

Table 7.1: Results on ten random classes of Caltech101 dataset. The second column shows the percentage of pyramid scales that can independently lead to correct recognition of each class. The third column shows the classification rate achieved when the decision taken is based on the majority voting over all scales of an image from that class.

Class	Percentage of scales	Classification rate
Platypus	12.10%	0.00%
Menorah	13.79%	1.30%
Gramophone	11.85%	2.44%
Rooster	13.43%	30.77%
Buddha	13.52%	32.00%
Camera	18.81%	72.50%
Stapler	29.66%	74.29%
Inline skate	22.22%	80.95%
Airplanes	51.71%	99.37%
Faces easy	47.32%	100.00%

7.1 Scale based classification

The efficiency of image classification depends on image representation as well as classification algorithms. Recent research, shows that pyramid image representations yields the best performance with the most popular being the spatial pyramid matching (SPM) [2]. That method represents an image as a long vector that includes information from all pyramid scales. However, the independent discriminatory capability of each individual pyramid scale has not been studied.

In order to demonstrate the need for scale-based decisions, we conduct an experiment using the Caltech 101 dataset. The method used is the ScSPM [17] with SIFT feature extraction [1] and SPM architecture [2]. The second column of Table 7.1, shows the percentage of scales that can *independently* correctly describe the images in each class. The third column reports classification performance achieved by our method.

As seen, even in classes that are easy to classify, the percentage of SPM scales that could independently represent the class accurately is less than 50%. This percentage decreases even more for classes that are more difficult to recognize. In practice, this means that much of the information included in conventional pyramid image representations may not be discriminatory and may hinder rather than contribute to correct classification.

We also observe that smaller training sets will proportionally provide a smaller number of less discriminant scales, leading to a well trained SVM. Advantages of proportionally smaller training sets are also evident on larger datasets with SPM vectors [10, 13, 14, 16, 17, 21, 124], where the number SVM training images are proportionally much smaller to the total number of images. So large training set are more prone to corruption that may lead to misclassification for all classification methods.

7.1.1 Conventional Support Vector Machine

In two and 3D methodologies [8, 13, 16, 17, 43], the classification of images, based on their pyramid representation $\mathbf{p}$, takes place using binary SVM decision functions of

the form:

$$f^j(\mathbf{p}) = (\sum_{i=1}^{M} a_i^j \mathbf{p}_i)^T \mathbf{p} + b^j = \mathbf{w}_j^T \mathbf{p} + b^j \tag{7.1}$$

which separate the jth class from the rest. In eq. (7.1) $\mathbf{w}$ is the normal vector to the separating hyperplane, a the Lagrange multiplier, and b the y-intercept of the border line created by the support vectors.

If Λ is the number of classes and M the number of available training images, the SVM learns Λ linear functions using the representation vectors $\mathbf{p}_i$ as training data, $\{(\mathbf{p}_i, y_i)\}_{i=1}^{M}, y_i \in \Upsilon = \{1, ..., \Lambda\}$. Each linear function $f^j(\cdot)$ is defined by $\{\mathbf{w}_j^\top \mathbf{p} | j \in \Upsilon\}$. For a test vector $\mathbf{p}$, its class label y is determined by

$$y = \max_{j \in \Upsilon} \mathbf{w}_j^\top \mathbf{p} \tag{7.2}$$

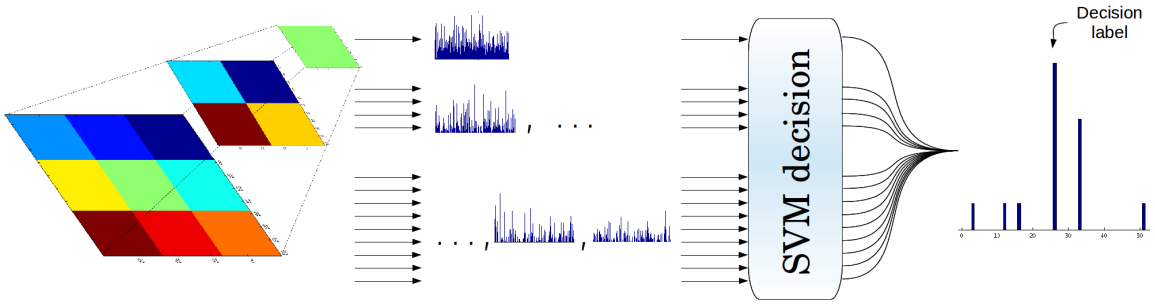


Figure 7-1: Classification using the novel SBSVM method. The SPM representation is decomposed in its pyramid scales. The scale vectors are input to the scale-trained SVM which provides a decision for each scale. The final classification label is the one that received the majority of votes.

7.1.2 Scale-based Support Vector Machine

In order to take into account the different discriminatory capabilities of different pyramid scales, we deployed scale-by-scale classification, in which each scale vector is classified separately and contributes to the final decision. The testing procedure can be seen in Fig. 7-1. Using this approach, each image is no longer represented using a single vector but, instead, the representation comprises a number of shorter

independent pyramid scale vectors. Training takes place using a random subset of the available images and we train a SVM using the independent pyramid scale vectors extracted from the training images. This will be experimentally shown to give an advantage over the conventional application of SVM in terms of classification performance.

We call this novel classification algorithm *Scale Based Support Vector Machine (SBSVM)*. The class label of an image, is no longer determined by its long SPM vector $\mathbf{p}$ but by its constituent individual scale vectors $\mathbf{s}_l$, $l = 1, 2, \ldots, L$, where L is the number of scales in the chosen pyramid representation architecture. In this way, instead of using the SVM decision functions of eq. (7.1), we use decision functions calculated using $M \times L$ scale vectors as training data. For $M \times L$ training scale vectors $\{(\mathbf{s}_k, y_k)\}_{k=1}^{M \times L}, y_k \in \Upsilon = \{1, ..., \Lambda\}$, the SVM learns Λ linear functions each one defined by $\{\mathbf{w}_j^\top \mathbf{s} | j \in \Upsilon\}$. Subsequently each scale $\mathbf{s}$ is classified independently:

$$f^j(\mathbf{s}) = (\sum_{k=1}^{M \times L} a_k^j \mathbf{s}_k)^T \mathbf{s} + b^j = \mathbf{w}_j^T \mathbf{s} + b^j \tag{7.3}$$

Each test image is represented by using the set of scale vectors the labels of which are predicted independently by the SVM. The final classification decision for the test image, is reached by a voting procedure based on the collection of the scale classification labels. The label that appears in the majority is the final classification label assigned to the test image. As will be seen in the ensuing section, the SBSVM improves classification performance.

7.2 Experimental Evaluation

7.2.1 Experimental setup

We compared our algorithm to a number of competing methodologies. In order to reach conclusive results, we repeat the dictionary training process ten times for the tested methodologies and randomly choose ten different training/testing sets for SVM training/testing.

For 2D classification, we use Linear Spatial Pyramid (ScSPM) [17] and Locality-constrained Linear Coding (LLC) [13]. We used the Caltech101 database from which we excluded the background class. For dictionary training we choose, as in [17], the size of all codebooks to be 1024 codeword vectors computed from randomly chosen feature vectors from across each database. We use a three-level pyramid with 21 pyramid scales, proposed in [17], kept stable through our experiments.

For 3D image classification we used methods such as Texture and Depth Sparse Representation Fusion [43] and Pyramid Efficient Match Kernels Over Kernel Descriptors [8]. We follow in part the framework presented in [8]. For dictionary training we choose the size of all codebooks to be 1000 codeword vectors from 20 random samples per image across all experiments. We use an identical dictionary training process for all the examined methods [8,17,43]. As for pyramid architecture, we use a three-level pyramid with 14 pyramid scales, as proposed in [8]. In the aforementioned 3D experiments, we randomly choose two images per class for training and use all the remaining images for testing. For BTDI dataset, the training set is roughly one fourth of the available data and in the RGB-D database we use about one third of the images for training.

7.2.2 Evaluation of SBSVM classification

On 2D datasets

We completed a series of experiments on the 2D Caltech 101 database. Our experiments follow the setup outlined earlier and are based on small training datasets. Results are reported in Table 7.2 and show the improvements that our method achieves under those conditions. As seen, the performance of SBSVM is better or equal than that of the conventional application of SVM, where larger training sets are assumed.

Further, we used training sets of up to 30 images per class. In those experiments the proposed method performs roughly as well as the SVM classification based on long representation vectors. The main reason for that is the large training dataset caring more scale vectors of low representational value which affect the classification.

For instance, a set of 30 images per class provides 30 training vectors per class for the conventional application of SVM while the same set provides to the SBSVM 630 training vectors per class.

Although, the SBSVM does not achieve state-of-the-art performance for this dataset when large training sets are used it still achieves impressive results with limited training sets.

Table 7.2: Results of Caltech101 dataset (± standard deviation).

Method / Classification	Number of training images				
	2	5	10	15	30
ScSPM / SVM [17]	38.93±1.01	55.24±0.85	62.96±0.40	**67.38±0.70**	**73.79±1.26**
ScSPM / SBSVM	60.27±1.43	**67.60±1.07**	63.12±2.27	63.32±1.81	70.08±1.23
LLC / SVM [13]	33.83±1.00	48.34±0.75	59.21±0.88	63.98±0.57	71.39±0.56
LLC / SBSVM	**61.71±0.54**	67.32±1.68	**64.52±1.47**	65.21±2.19	66.35±2.70

On 3D datasets

In order to evaluate our method for 3D classification, we compared our novel SBSVM method to the conventional SVM approach, in conjunction with a number of feature extraction and representation options that are used in 3D methodologies. To this end, we used the methodology in [43], where depth information is used along with the scale vectors (coming from texture) for the training and classification using SBSVM.

We applied the experimental procedure detailed in Section 7.2.1 for SBSVM and SVM classification. The results for all 3D image classification methodologies are shown in Table 7.3. We observe that our SBSVM classification algorithm yields a major improvement over the regular SVM in every method. The reason for the improvement is the suitability of the SBSVM classification even when only few training images are available.

Both the individual and combined features in [8] benefit from SBSVM classification. Further, the combined features (last row of Table. 7.3) yield superior performance by using a mix of scale vectors corresponding to different features. This yields major improvements over the conventional SVM approach for the classification of SPM vectors.

Table 7.3: Experimental results on two three- dimensional datasets (± standard deviation). As seen, when limited training samples are available, SBSVM is significantly better than the conventional SVM approach.

Features	Pyramid		BTDI dataset SVM	BTDI dataset SBSVM	RGB-D dataset SVM	RGB-D dataset SBSVM
SIFT + ShCid (sl.method 1)	ScSPM+BoF	[43]	52.49±4.41	86.29±3.12	34.45±2.75	85.78±2.14
SIFT + ShCid (sl.method 2)	ScSPM+BoF	[43]	47.93±4.98	86.37±3.05	33.02±2.82	86.11±1.85
SIFT + ShCid (sl.method 3)	ScSPM+BoF	[43]	57.17±5.30	86.24±3.37	33.98±2.81	85.33±2.13
SIFT + ShCid (sl.method 4)	ScSPM+BoF	[43]	51.92±6.41	85.92±3.14	33.96±2.99	86.06±1.91
RGB Gradient KDES	EMK SPM	[8]	57.01±5.15	91.79±2.93	34.69±2.86	88.35±2.02
RGB LBP KDES	EMK SPM	[8]	54.88±4.88	93.61±2.92	32.44±2.60	92.77±1.86
RGB Normalized Color descr. KDES	EMK SPM	[8]	47.04±4.97	93.25±2.58	28.11±2.89	91.70±1.54
Depth Gradient KDES	EMK SPM	[8]	43.88±4.91	61.15±4.59	25.64±2.81	78.79±2.55
Depth LBP KDES	EMK SPM	[8]	41.72±6.01	73.08±4.33	23.67±2.64	85.91±2.03
Point cloud Size KDES	EMK SPM	[8]	-	-	29.94±2.88	78.96±2.05
Point cloud Normal KDES	EMK SPM	[8]	-	-	27.18±2.29	79.76±2.32
Feature combination	EMK SPM	[8]	62.26±5.07	**97.54±1.81**	47.80±2.77	**98.62±0.80**

7.3 Conclusion

In this chapter, we introduced a novel classification method for pyramid image representations. We experimentally showed that the scales used in a conventional SPM unequally contribute to the correct classification of an image. Our method takes this fact into account in two and 3D image classification and achieves excellent results, outperforming the conventional SVM-based classification that uses concatenated SPM vectors. Finally, our SBSVM method achieves state-of-the-art results in 3D image classification. The main contributions of this work are:

- Our research showed that often scales do not provide sufficient representation of image content, thus not contributing favorably to classification rates.
- When using individual per scale classifications we achieve a solution for scale discrimination issues. However, big training sets do not seem to benefit from this approach.

For future improvements on classification, we plan a further improvement for this method by using multiple SVMs. Each SVM classifier will correspond to individual scales rather than one for all scales. This could solve issues, such as reduced performance when using big training data sets, where the magnitude of non discriminative

scales is overwhelming for one classifier. This application can be optimized with modern parallel and multi-processing techniques. We also consider the combination of our classification technique with features extracted by deep learning algorithms.

Chapter 8

Deep Learning methodologies for 3D image data

3D image classification methodologies use depth in order to enhance classification efficiency. Advances have been made in feature extraction depth maps [7–9, 43] and their combination with texture features. These methods are closely related to 2D image classification in their overall system design. The design of image classification systems consists of processes including feature extraction, dictionary learning, feature encoding, image representation and classification. These methods are based on architectures that that have been seen to perform well. However, core processes in image classification systems, such as feature extraction and image representations, are not optimized in a way with regard to the nature of used image data.

More recently, deep learning was proposed as an alternative design and system optimization methodology. Deep learning is a machine learning technology where the proposed system captures learning representations, i.e, features and vector representations, of input data. This particular attribute of deep learning algorithms is used to deal with the aforementioned optimization problem. The most popular deep learning algorithms are the Deep Convolutional Neural Networks (CNNs) [12] and the Deep Belief Networks (DBNs) [64]. These machine learning algorithms were recently implemented in a plethora of computer vision problems [4, 6, 12, 59, 61, 62, 64], including image classification [5, 60], yielding state-of-the-art results. All the aforementioned

categorization systems are implemented on 2D image data.

This chapter presents our research on deep learning methodologies and implementations on 3D image classification. We present our novel research implementing 3D image data on deep learning algorithms. We study the effects of input data on a CNN model using color, depth and their combination in order to enhance the overall recognition rates. We also propose a novel method that formulates image representation methods [2, 108] as input channels. Our goal is to enhance CNN input data with spatial and content-adaptive information provided by these representations. Finally, we present experiments regarding the performance of CNN architectures.

The rest of this chapter is organized as follows. In Section 8.1, we present related work of CNN implementation for 2D and 3D image data. Our research and proposed methodologies are presented in Section 8.2. The experimental setup and results are shown in Section 8.3. Finally, conclusions and future work are discussed in Section 8.4.

8.1 Related work

The CNN algorithm was firstly proposed by Fukushima in [65], which introduced a naturally inspired model for pattern recognition. However, that early model is computational intensive and prone to overfitting.

The most influential work of modern deep neural network techniques was introduced by Lecun in [59]. The proposed model improved the previous model of Fukushima, proposing backpropagation in CNN. That proposed CNN architecture was proposed for document recognition. The model, known as LeNet-5, introduced a multi-level network architecture that includes convolutional and pooling layers. That model was implemented for recognizing numbers, but was not capable of solving more complex problems due to overfitting.

The work in [12], introduced improvements over LeNet-5 architecture including rectifier linear unit layers (ReLu) and a "dropout" regularization. The ReLu layers improved non-linearity and computational efficiency without corrupting the computed features. Moreover, the proposed "dropout" function solved the overfitting problem.

The proposed model in [12] consisted of 7 layers and 60 million different parameters. The computational cost for training and validation made authors turn to an optimized Graphical Processing Unit (GPU) algorithm which achieved state-of-the-art results on ImageNet dataset [70]. Methodologies proposed in [12] were implemented by all modern convolutional neural networks.

More recent methodologies are based on novel architectures, that enhance the performance of conventional CNNs. The proposed CNN methods focus on the pre-processing of input data and custom CNN layers in order to provide solutions to CNN disadvantages.

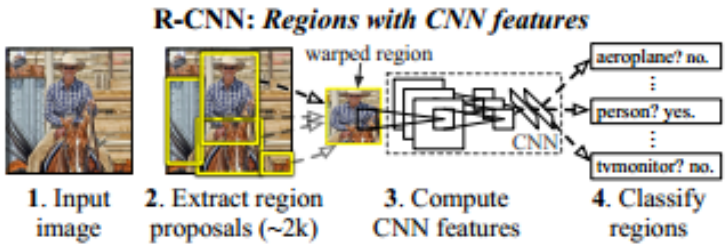


Figure 8-1: The architecture of region-based CNN model in [4].

The work in [4] proposed a region-trained CNN for object detection and categorization. The proposed model extracts approximately 2000 bottom-up regions from each examined image. All regions are resized in a specific resolution and then passed as inputs to a CNN model. Region representations are then classified using multiple class-specific linear SVMs instead of the conventional CNN "decision" layers. As presented in Chapter , decision layers are denoted those CNN layers degrading extracted features and representations to a vector form thus describing each examined image. That method yields state-of-the-art results in VOC 2012 [125] and the 200-class ILSVRC2013 detection dataset [60].

In [5], authors proposed a custom CNN layer inspired from the pyramid image representation method of [2]. That image representation method, known as spatial pyramid matching (SPM), is a hierarchical representation of predefined rectangular image regions termed scales. SPM can represent images of various sizes, a general problem for CNN architectures that requires that all images are of predefined reso-

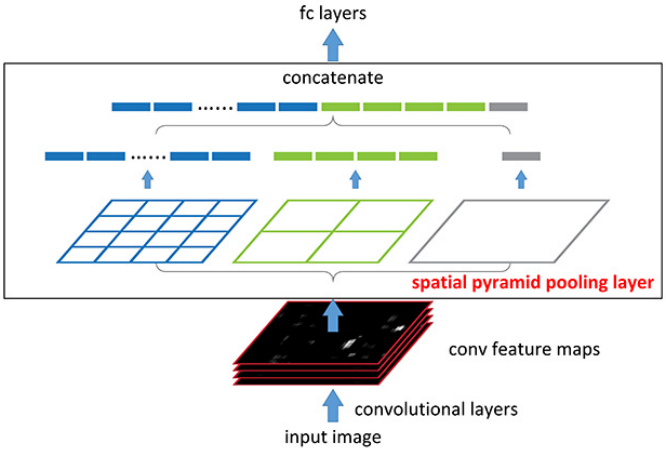


Figure 8-2: The Spatial Pyramid Pooling layer (SPP) as presented in [5].

lution. That method extracted image features using a series of convolutional layers with no regard to image resolution. Their proposed Spatial Pyramid Pooling (SPP) layer combines all image features like the SPM method before the "decision" layers. That method yields state-of-the-art results for the VOC 2007 [69] and Caltech101 [68] image datasets.

An CNN implementation for 3D image data was proposed in [6]. That method proposes an object detection system using CNN-extracted features from texture and depth. The most important contributions of that work is the examination of CNN features represented by a three-channel depth representation. The proposed features represent horizontal disparity, the height above ground and angle in relation to gravity. All proposed features result to representations classified by a SVM model. It is experimentally shown that the aforementioned features are more discriminative than single channel depth information that describes horizontal disparity. That method yields state-of-the-art results in object detection using the NYU depth dataset V2 [82].

Classification for 3D images can benefit from the development of CNN algorithms. The aforementioned methods have similarities with our proposed methodologies [108, 119] for 3D image categorization.

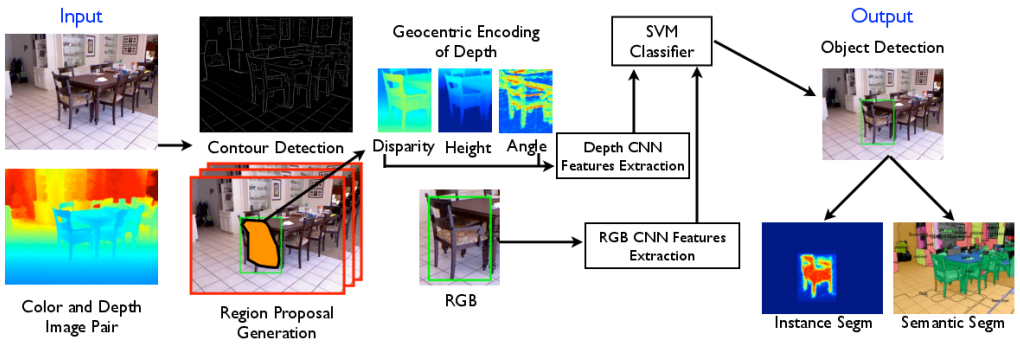


Figure 8-3: The CNN model in [6], performing feature extraction from depth information .

8.2 Convolutional neural networks for 3D image data

CNNs are becoming more popular for applications such as image/object categorization [5,60] and scene understanding [62]. These implementations have been presented primarily for 2D image data. 3D image data, i.e, images represented by texture and depth, were used in many image classification and object recognition applications [7–9, 43]. Advancements in these areas are mainly in the processes of feature extraction and image representations.

Feature extraction is the process of representing images using vectors with characteristics from the contents of each examined image. Research in depth feature extraction methods rely in different approaches due to the nature of depth information. Methods in [7–9] proposed feature extraction algorithms related to other known 2D methodologies [10, 11]. As a result, the extracted features are not fully suitable for depth information in less detailed depth maps. In [43], a shape features extraction method on depth maps is proposed in order to address the disadvantages of previous methods.

Advances in image representations for use in image categorization systems are limited, due to the design complexity of such methods. The most commonly used image representations in 2D and 3D image classification are methods in [2,16]. These methods captured spatial information and represent each examined image with no regard to the depicted content. More recently the method in [108] uses depth to represent content information, thus creating a content-adaptive image representation.

All aforementioned features and image representation methodologies are inspired from researcher's concepts about each imaging problem. Those methods are eventually optimized using specific benchmarks. Method efficiency is the outcome of innovative design and exhaustive experimental trials. However, according to deep learning theory, methods are subject to design limitations, as a result, proposed methodologies may not be suited to the nature of 3D image images.

Contrary to the conventional approaches, CNN is a complete self-adaptive categorization model that adapts to input data. Input data are degraded by multiple convolutional layers that lead to a final decision. Each convolutional layer is forming its own features, consisting of convolutions of input data multiplied with a layer kernel. In addition, other processes such as pooling are used to create more efficient representations used as input to other convolutional layers. CNN training is an iterative process including a backpropagation phase rectifying individual layer kernels, resulting to reduced training errors and better recognition rates. Except of the initial image data and CNN architecture, nothing else is user defined. As a result, features are not subject to the aforementioned limitations of research-defined features. Overcoming this problem, CNNs yield state-of-the-art results in many applications [4–6, 12, 59–62, 64].

8.2.1 CNN-based features extracted from 3D image data

As described earlier, a 3D image is the combination of texture and depth image information depicting the same content. In most image categorization methods, texture is described by features using image gradients, i.e., the variations of intensities inside that image. However, texture can also be represented from three channels (red, green, and blue) describing each color intensities. The conventional CNN feature extraction uses all three color channels after subtracting the mean value of each channel computed from images inside the dataset.

For 3D image categorization, depth disparity values can be used in similar fashion, i.e., as a channel from which the mean depth has been subtracted. This setup will provide a CNN model trained using depth information. To this end, we will be able to reach a conclusion about the discriminatory capabilities of depth features.

The findings of current 3D image categorization [7–9,43,108] experimentally showed that the best-performing systems achieve their results using combinations of texture and depth. In Section 8.3.4, we present a series of experiments on 3D image data. We use texture, depth and their combinations in order to reach conclusions about the discriminatory capabilities of 3D image features. Moreover, we will be able to assess feature extraction capabilities of CNN on multi-channel input data.

8.2.2 Image representations enhancing CNN features

In general, the CNN algorithm does not support conventional image representations, i.e., vectors that represent an image as histogram of its comprising feature encodings. The advantage of image representation lies in the fact that each image is described with respect to predefined spatial information [2, 16] or individual content-adaptive information [108].

Little work has been done regarding the application of spatial or content information in a CNN model [5]. This is mostly because the aforementioned methods were designed to represent images with no regard to their initial size. Input CNN data currently require fixed resolution images that create the same number of convolutions for every image. The problem is found in the final convolutional layers, i.e., the decision layers, where CNN features are degraded to vector form. These layers require a "strict" architecture in order to provide the final decision.

The authors in [5] try to overcome the problem of fixed-sized input images by introducing a spatial pyramid pooling strategy. The spatial pooling layer is a pooling layer inspired from the work in [2] which is implemented before the first decision layer. Essentially, that pooling layer implements a spatial pyramid combining CNN features. As a result, that architecture allows an interdependent number of convolution features to be combined from the pooling layer and provides a more discriminative representation that captures spatial information.

In this chapter, we propose a different implementation of image representations used in CNN models. CNNs have the attribute of learning patterns from input data very efficiently. So, we implement image representations as input channels to a

CNN model. These channels will provide enhanced information to input data, thus achieving improved categorization performance.

Spatial Pyramid Matching (SPM) [2] image representation regionalizes an image in predefined rectangular regions termed scales. Each scale is a representation histogram of encoded image features found inside that region. The final representation is the concatenation of representation histograms from all scales. A similar implementation is presented in [108] but, instead of rectangular regions, we used content-adaptive regions, achieving more discriminative representations. More information about image representation methods can be found in Chapter 2.4.2 and 6.

Our implementation uses the aforementioned representations implementing a spatial or content partitioning on pixels consisting the examined image. For spatial partitioning, we use the SPM representation [2] which creates a grid of $l \times l$ rectangular regions where $l = 1, 2, ...L$ and L denoting the pyramid level. When $l = 1$ the representation presents all pixels comprising an image, which is equal to the BoW model [54] and the common CNN implementation. The CAMP representation [108] uses individual regions describing parts of objects and scenery derived from content partitioning. Content region extraction is achieved by using spectral clustering on image information such as color and depth. For our experiments, we used the k-means algorithm to cluster pixels values and achieve similar content regions. We partition an image to l content regions using depth information for content indexing, as proposed in [108]. Scale labels, for both representations, are further normalized with values ranging from zero to one. As a result, we are able to describe spatial or content-adaptive information as a data channel which can be combined with other input data. Experiments of our proposed methodologies are shown in Section 8.3.5.

8.3 Experimental results

8.3.1 Dataset

Some of the most used datasets to evaluate the performance of proposed CNN architectures are the ImageNet [70], CIFAR 10 [71], CIFAR 100 [71] and the Caltech 101 [68] dataset. ImageNet is a massive collection of 2D image data, categorized as a WordNet hierarchy of 21841 synsets with an average of 650 images per class, totalling 14,197,122 images. CIFAR 10 is a collection of 60000 32×32 resolution images which are organized in 10 classes. An alternative version of the aforementioned dataset is CIFAR 100, that consists of 100 classes containing 600 images per class. The CIFAR 100 can also be used as a dataset of 20 superclasses and 100 subclasses. Finally, the Caltech101 is a known dataset consisted of 102 classes, used for the evaluation of many 2D image classification methodologies. The common feature between all the aforementioned datasets is their data magnitude, a necessary attribute for CNN training.

For our experimental setup, we used the RGB-D dataset [7]. This 3D imaging dataset is compiled by 300 household objects depicted in multiple views in two axes, i.e., multiple angles horizontally and in different height angles. The resulting 51 class dataset contains approximately 41877 object frames. The advantage provided from this dataset is the magnitude of available images for training and validating the proposed CNN models.

8.3.2 CNN architecture

For each CNN model we used a predefined architecture proposed by [126] optimized for the CIFAR 10 dataset [71]. This dataset is comprised by 60000 32×32 resolution color images ordered in 10 classes. The proposed CNN model has three convolution levels for feature extraction and other two forming decision layers. The first feature extraction layer creates 32 kernels of $5 \times 5 \times 3$ pixels including all three color channels. The layer has 32 outputs of 32×32, that are degraded by a 3×3 max pooling layer

concluding to 32 outputs of 16 $\times$ 16. The second layer has 32 kernels of 5 $\times$ 5 $\times$ 32 convoluting the output of previous layer resulting to 32 outputs of 16 $\times$ 16. The max pooling layer of 3 $\times$ 3 provides the final 32 output of 8 $\times$ 8. The final feature layer uses 64 kernels of 5 $\times$ 5 $\times$ 32 outputting 64 of 8 $\times$ 8 features also degraded by a 3 $\times$ 3 max pooling layer concluding to 64, 4 $\times$ 4 outputs. The first decision layer is a convolution layer that creates 64 kernels of 4 $\times$ 4 $\times$ 64, producing 64 outputs of 1 $\times$ 1. The final convolution layer takes the aforementioned features and creates 10 kernels of 1 $\times$ 1 that output a class identity vector.

We chose this architecture because it is easily implemented on 3D data and requires relatively less computational power than other CNN architectures. Of course, each dataset requires its dedicated architecture in order to provide optimal results. However, the solution of a dedicated architecture is time consuming and unfortunately not presented in this thesis. Our initial research regarding CNN model architectures is presented in the ensuing Section 8.3.6.

Our experiments require changes in the aforementioned CNN model architecture. The default first feature extraction convolution layer uses kernels of 5 $\times$ 5 $\times$ 3 that present the three levels of color. To this end, we change kernel dimensionality in a way suiting each proposed experiment, as for instance, CNN model that uses depth maps as input data requires kernels of 5 $\times$ 5 $\times$ 1 because depth is a single channel image information. The input layer of each experiment will be presented in individual experiment sections. Finally, the final decision layer is changed from 10 to 51, 1 $\times$ 1 kernels that represent the number of classes in the final class identity vector.

8.3.3 Experimental framework

RGB-D dataset was presented in two experimental frameworks [7, 119]. The experimental setup in [119], RGB-D dataset was used in a single-view configuration. Training images were far less than testing images, thus creating a challenging validation setup. Authors reject the multi-view framework of [7] using one random view per object, degrading the initial dataset to just 300 images each depicting one object. Two images per class were used for training the classification algorithm and the rest

were used for validation. However, the use of this particular training/testing framework cannot be applied to a CNN model, because deep learning algorithms require large training sets.

In [7] authors proposed a multi-view experimental framework for the RGB-D dataset. That framework uses one object and its comprising views per class for validation while using the remaining objects for training the classification algorithm. The results of several methods using this framework are presented in Table 8.1.

Table 8.1: Results of the RGB-D Object Dataset using the multiple view framework.

Methods		RGB	Depth	RGB-D
SIFT + Texton + Color Histogram + Spin Images + 3D Bounding Boxes	[7]	74.5	64.7	83.8
Sparse Distance Learning	[127]	78.6	70.2	85.4
RGB-D Kernel Descriptors	[8]	80.7	80.3	86.5
Hierarchical Matcing Pursuit	[9]	**82.4**	**81.2**	**87.5**

We also use that framework because it provides the necessary number of training and testing images. In order to provide solid conclusions, we keep the number of training iteration to 300 for each examined CNN model. We use this framework throughout our experiments and compare our results with the state-of-the-art multi-view results.

8.3.4 Research on CNN features

The default CNN model of CIFAR-10 uses texture images, i.e, images with three color channels. Input images are normalized by subtracting the mean color values calculated by images comprising the dataset.

It is proven by common practice that data ranging from 0 to 1 perform better in various mathematic complications and classification algorithms. We applied this empirical method to all individual texture and depth input channels, and we then conducted experiments with or without mean normalization. Table 8.2, shows that the proposed mean normalization has an adverse effect on CNN performance in com-

parison with experiments that did not use this normalization.

Table 8.2: Results showing the training/validation error of the RGBD dataset when using diffent input data (less is better).

Input data	Training err.	Validation err.
Mean Color	53.09%	73.26%
Color	41.84%	65.26%
Mean Depth	77.81%	90.09%
Depth	63.69%	77.24%
Color + Depth comb. Mean	**39.78%**	**59.51%**
Color + Depth idv. Mean	45.57%	65.29%
Color + Depth	**38.44%**	**60.93%**

We also used the available depth maps corresponding to each image as input data, in order to assess the discriminative capabilities of depth CNN-computed features. The use of depth as a CNN input required the change of the first convolutional layer, where we use 32 kernels of $5 \times 5 \times 1$. The rest of the CNN model is kept as described in the previous Section 8.3.2. This setup allowed us to perform experiments with or without mean depth normalization. As shown in Table 8.2, the conventional mean normalization is not favoring the recognition results. We also observe the difference in performance between depth and texture, where texture is outperforming depth in every experiment.

However, as shown in [7–9, 43, 108, 119], the contributions of depth is shown in combination with texture. To this end, we changed the first feature extraction convolutional layer to 32 kernels of of $5 \times 5 \times 4$, where four is the number of three channels plus the depth information. For this setup, we conducted three experiments including a conventional mean normalization in all data with no separation, a mean normalization applied individually to texture and depth before combination in a common input data matrix. We also conducted an experiment with no mean subtraction. The results in Table 8.2, show that the combination of texture and depth provides improves performance compared to CNN models using individual features. The experiment where all input channels are used subtracted by their mean value, is

performing equally well with those not using mean normalization.

In conclusion, when image input data, such as texture and depth are used separately then optimal performance is achieved with no mean normalization. However, when combining depth and texture, the CNN model yields top-performing results. As for the use of mean normalization in data combination, no solid conclusion can be drawn. The combination of input data generally requires further research. We consider that more iterations will eventually provide the differences between the aforementioned models.

8.3.5 Image representations as CNN data

As described in Section 8.2.2, we implement known image representation methods [2, 108], as separate channels, in order to enhance the performance of CNN models.

Our proposed method creates extra channels including spatial or content-adaptive representation in order to enhance feature extraction capabilities of CNN models. Our proposed method is much simpler than the creation of a custom pooling layer, which requires changes in the CNN architecture. We cannot claim that our method is more efficient than the one in [5], because there is no comparison under a common experimental framework, i.e, datasets or CNN model architectures. However, Table 8.3 presents a series of experiments regarding the efficiency of our novel methodology compared to CNN models not using our method. We can also compare the representation capabilities of each image representation methods [2, 108].

For the SPM representation, we use 32×32 resolution of images to create a spatial grid of $l \times l$ scales, $l = 1, 2, ...L$ where L denotes the number of levels. Labels are assigned to each scale, which are normalized in the range of zero to one. Pixels corresponding to the first scale (top left) are represented with zero value and with one those found in the $l \times l$ scale(bottom right). The rest of the pixels are presented by intermediate values between zero and one that are computed with regard to their initial scale label and the overall number of scales $l \times l$.

A similar procedure is followed in the construction of channels corresponding to the CAPM representation, which captures image content information. As described

Table 8.3: Results showing the training/validation error of the RGBD dataset when enhancing input data with formulated image representation methods (less is better).

			Training error		Validation error	
Im. repres.	L1	L2	50 iter.	300 iter.	50 iter.	300 iter.
none	-	-	-	41.84%	-	65.26%
SPM	1	-	52.61%	40.62%	70.82%	62.42%
	4	-	53.55%	40.68%	73.38%	62.92%
	9	-	54.91%	42.78%	73.23%	64.25%
	16	-	54.94%	42.8%	73.23%	64.35%
	1	4	51.83%	41.82%	71.29%	63.88%
	4	9	50.51%	41.19%	69.59%	62.53%
	4	16	49.82%	40.35%	69.14%	62.47%
CAPM	2	-	53.00%	42.52%	68.89%	60.54%
	3	-	50.36%	39.77%	73.52%	65.25%
	4	-	56.62%	45.35%	73.33%	61.39%
	6	-	55.39%	42.76%	69.49%	61.64%
	8	-	49.28%	36.79%	66.49%	60.35%
	10	-	60.15%	46.67%	75.42%	63.84%
	1	4	58.51%	49.03%	67.92%	60.02%
	2	4	54.18%	42.15%	66.05%	59.18%
	2	6	56.11%	44.49%	70.77%	63.39%
	2	8	51.35%	40.17%	69.44%	61.20%
	4	8	**48.26%**	**38.17%**	**60.50%**	**53.04%**
	6	8	51.19%	40.68%	71.43%	65.22%

in Chapter 6, the most suitable content indexing information is depth. In order to simulate the computation of content regions used by the CAPM representation, we use k-means over depth pixels values. k-means algorithm is able to efficiently regionalize the content of an image into k number of content regions. The content labels provided for each pixel are normalized in the range of zero to one, similar to the process presented for the SPM representation.

In our experiments, we show the impact of image representations on CNN models by enhancing texture image information, i.e, three color channels. As described in Section 8.3.4, the best performing single feature normalization is with no mean subtraction. So, we use this setup in the following experiments. In order to represent one pyramid representation level, we use the first feature extraction convolutional layer of

32 kernels of $5 \times 5 \times 4$, including the color channels and the representation channel. The representation of two pyramid levels is achieved using a convolutional layer of 32 kernels of $5 \times 5 \times 5$, where color and two representation channels are defined. This layer architecture is used for all experiments of both examined representations.

Table 8.3, shows the classification error achieved from each proposed model, with the smallest percentage showing the best results. As shown in Table 8.3, the use of image representation such as CNN input data has a positive effect upon recognition rates. The use of a single level pyramid representation shows that all experiments using representation channels, outperform the model trained only with color channels. In a comparison between image representations, the results show that CAPM models yield better recognition than SPM models thus achieving smaller validation error. This is also the case when using two-level pyramid representations, i.e, dual channel representation input data. CAPM representation channels representing four regions in the first layer and eight in the second yield 12.12% less performance error than the color-only CNN model and 10.84% than the best-performing SPM architecture.

8.3.6 Experiments on CNN architectures

Throughout our experiments we used the CNN model proposed in [126] which is optimized for the categorization of CIFAR 10 dataset. That architecture was designed in order to efficiently categorize the 32×32 resolution images consisting CIFAR10 dataset. That is a suboptimal setup for the RGBD dataset, which consists of multiple resolution images. By resizing each image to 32×32 we were able to use this architecture. However, when resizing an image a part of depicted information is lost. As a result, image resizing is adversely affecting the overall recognition performance.

In this section, we perform small changes on the default CNN model of [126] studying their effects on performance. We also implement a similar input method such the one proposed in [4], where images are partitioned in multiple image patches describing smaller regions inside an image. This process may provide a solution to the aforementioned image resizing problem.

As shown in Table 8.4, when resizing an input image to a bigger one, changing

Table 8.4: Results showing the training/validation error of the RGBD dataset when applying changes on the used CNN model (less is better).

Image size	Layer 1 arch.	Input data	Training err.	Validation err.
32 × 32	default	Mean Color	62.42%	76.58%
32 × 32	default	Color	55.67%	72.04%
32 × 32 im. Patches	default	Color	43.04%	76.77%
64 × 64	default	Mean Color	83.16%	91.07%
64 × 64	default	Color	84.26%	93.07%
64 × 64	10×10×32	Color	68.43%	83.34%
64 × 64	5×5×64	Color	81.31%	90.16%
64 × 64	10×10×64	Color	66.72%	80.45%

CNN model architecture then the performance is adversely affected. When resizing an image without changing the input layer, then only a small part of the examined image will contribute to feature extraction. Even when the input layer is resized with more smaller kernels, i.e, increasing the convolutions from 32 to 64 kernels and keeping patch size to 5×5, recognition performance is not improved. Because of the smaller patch size extracted feature contain the same information. As shown in the experiment, where we resize each kernel size, CNN architectures prefer increased convolution patch size.

Finally, we conduct an experiment inspired by the method in [4] where instead of resizing an image, individual image patches are used. To this end, we segment each image in multiple patches of 32×32. The experimental results show this configuration to favor feature learning, i.e, low training error. However, increased feature learning does not translate to lower validation errors as well. The method in [4] used multiple class-specific SVMs for classification. In the proposed experiment the classification is performed by the CNN model. So, the results of this method should be further studied when using different classification methods, such as the one proposed in [119].

We have experimentally shown that using this architecture for our examined dataset may not be the optimal solution. However, in future work, we plan to propose a dedicated architecture for this dataset. This will lead to improved feature extraction and representation and possibly to improved object recognition rates. We also

consider conducting experiments, using individual image patches as described above.

8.4 Conclusion

In this chapter, we presented a research regarding the use of 3D image data on CNNs. We used the model of [126] in order to examine the efficiency of CNN models when texture, depth and their combination are used as input data. We proposed the integration of known image representations as input data channels to enhance the CNN performance. Finally, we experimentally showed the effects of classification performance achieved by changes in model architecture. Our research contributed to the following:

- We experimentally show that the combination of depth and texture as CNN-extracted features can achieve improved recognition performance.
- We introduce a novel implementation of formulating image representations as input channels leading to improved results.

This chapter is a prelude to our research in deep learning algorithms and their implementation on 3D data. Our future plans include an extended research on dedicated CNN model architectures. These models will either use depth and texture as combined input data or a model of individual CNN models contributing to a common decision. Regardless the proposed model, deep learning algorithms on 3D imaging data are a promising research topic and field of applications.

Chapter 9

Conclusions

In this thesis we presented the development of our novel 3D image classification methodologies. We conducted a complete research focusing on depth information, presenting its advantages, limitations and contributions to image classification. Each of the proposed methods highlighted the disadvantages of competing methods, and proposed novel techniques that either rectify or redefine the examined problem. As a result we achieved improved efficiency and state-of-the-art results in many processes of 3D image classification. In this chapter, we conclude this thesis by summarizing our work, presenting our contributions and future plans.

9.1 Thesis summary

Literature review of Chapter 2 describes the structure of modern image classification systems. We described all processes comprising such systems, including feature extraction, dictionary learning, feature encoding, image representation and classification algorithms. We also included a review on deep learning methodologies and analyzed the convolutional neural networks (CNN) model.

In Chapter 3, we introduced a novel collection of stereoscopic images which we organized in a 3D dataset. This chapter describes the design, acquisition process, and 3D imaging problems where our novel dataset can be used to improve current research.

Our proposed methodology in Chapter 4 includes a feature extraction method, extracting discriminatory information over depth maps.In order to assess our proposed feature extraction method, we also proposed an image classification system able to combine depth and texture information. We presented the individual steps of shape feature extraction, and the process of combining texture and depth representations yielding top-performing results in two 3D image datasets.

In Chapter 5, we proposed a novel dictionary learning algorithm that is constrained by context similarities between dictionary training features. Our method formulates relations between dictionary training features coming from similar context yielding top-performing results when combined with compatible image features.

Our work on image representations was presented in Chapter 6. The proposed method describes an efficient way of creating discriminative image representation in respect to depicted image content. The resulting image representation outperforms the competing SPM methodology [2], competing methodologies in multiple implementations on known 3D image categorization systems [7–9, 43] for two 3D imaging datasets achieving state-of-the-art results.

Chapter 7 presents our novel classification algorithm which uses individual scales of modern pyramid representation to enhance the classification performance. The design of our proposed algorithm provides a per-scale decision combined with majority voting. Our novel methodology yields state-of-the-art performance in two 3D image datasets.

Finally, in Chapter 8 we presented our research regarding the implementation of deep convolutional neural networks on 3D image data. Our proposed methodologies uses depth and its combination with texture as input data for CNN models and propose a novel formulation of known image representations with great results.

9.2 Contributions

In this section, we present the contributions of our proposed methodologies arranged as individual processes of 3D image classification systems. We also include the con-

tributions of our research on deep learning algorithms.

9.2.1 3D data collection

For our research purposes, we compiled a **collection of stereoscopic images** denoted as Brunel Texture and Depth Image database (BTDI). The stereoscopic acquisition method used for the collection of our dataset, is the most widely used type of portable 3D image camera in commercial use. Stereoscopic images solve issues of depth acquisition presented under various illumination conditions.

Stereoscopic images have the disadvantage of providing less detailed depth information than those acquired from other 3D cameras types. However, less detailed depth maps must be taken into consideration when designing robust and discriminative 3D image methodologies. The aforementioned attributes ensure a **challenging benchmark** which can be used for a variety of 3D imaging problems other than image classification, such as segmentation and stereo correspondence.

Due to public availability of stereoscopic cameras, data acquisition is simpler and can be achieved by less experienced researchers. However, the magnitude of our current dataset version is smaller than other competing 3D datasets. We currently plan an expansion for our proposed database by adding 430 images organized in 12 new classes. Also, the popularity of our dataset is still very low, but we plan to have this resolved by public availability.

9.2.2 Depth feature extraction

The special properties of depth emerged new challenges in feature extraction, because common feature extraction methods similar to texture method cannot be applied. Other known methods [7, 10, 11] do not fully utilize properties of depth, as described in Chapter 2.1.3.

Our novel **shape feature extraction methods over depth maps** are presented in Chapter 4. Our methods propose a "slicing" process where image regions are captured in multiple depth levels. We proposed four depth map slicing methods that

describe individual or combined regions. The extracted regions describe shapes of depicted objects described by suitable shape features. The extracted depth features are constructed with respect to depth properties, and are compatible with depth maps captured by most 3D sensors. Our novel descriptors are experimentally shown to achieve top-performing results in Chapter 4.

Depth features, as standalone descriptors are not performing as well as features extracted from texture. Of course texture information is more "enriched" than depth map information. However, research for depth features has not quite reached its maximum potential. Therefore, we continue our research in the development of such methods.

9.2.3 Dictionary learning

Dictionary learning plays an important role in the discrimination of image representation and by extension to the overall recognition performance. This process is often neglected, settling for dictionary learning methods that do not provide optimal representations.

In Chapter 5, we introduce a novel **dictionary learning method constrained by context similarities**. Dictionary methods are either computed (k-means, K-SVD) or constrained (GNMF) by similarities, i.e, distances, between dictionary training features. However, the discriminative representation of features sometimes fail thus producing same representations for unsimilar context. To solve this issue, we formulate each feature's context, i.e, image regions from which each feature was extracted. The proposed context similarity constraint is applied to GNMF dictionary training thus achieving top-performing results.

Common practice shows that nonnegative features are more discriminative than those including negative values. Because our proposed method in based on NMF, features with negative values cannot be processed. Even when we enforce non-negativity on this features, recognition performance drops due to feature design. In general, dictionary learning methods are chosen to complement the performance of preferred features. So, we have to apply this strategy to present the full potential of our pro-

posed dictionary learning method.

9.2.4 Image representation

In Chapter 4, a depth feature extraction process is described where the sparse feature encodings are combined by a dedicated depth image representation. For depth image representation we used a BoW model of encoded features, combined with a max pooling function thus creating a discriminative representation. The final image representation is formed by the **concatenation between texture and depth representations**. Our novel method achieves enhanced classification results over conventional texture representations, proving the representational value of depth. The experimental results of Chapter 4 show the top-performing capabilities of our proposed methodologies in comparison with other known methods. However, depth representation could be improved by using more efficient image representation methods, such as SPM or our novel CAPM method.

The aforementioned representation method inspired our further **research for designing efficient image representations** used in 3D image classification. We conducted experiments, shown in Chapter 7, which demonstrate that individual scale vectors comprising pyramid representations, such as SPM, are more representative than others. These discriminant scale vectors favor recognition and contribute to improved categorization results. But when concatenated with less representative scales from the same image, classification performance is adversely affected. The findings of our research led us to the development of our novel scale-based classification algorithm [119].

This research also revealed margins for improvement on image representations design. Image representations, like SPM, describe image content with multiple regions with respect to spatial information. However, rectangular regions do not take the depicted image content into account. As a result, this representation is not so efficient.

We proposed a novel **pyramid image representation creating regions based on image content**. Each pyramid level of our novel pyramid representation, partitions the image into a number of regions. These regions are no longer rectangular

but are arbitrarily shaped according to image content. So, our scale vectors represent depicted objects/scenery and their consisting parts. Our method provides state-of-the-art results in two 3D image datasets. During the experimental process, we also prove that **depth is the best image information to describe depicted content** inside an image. Further improvement for this method includes the combination of spatial and content information, and the use of deep learning algorithms.

9.2.5 Classification

Our research in designing image representation in Section 9.2.4, showed that representational value of individual pyramid scales is lost when concatenated in the final representation vector. In order to solve this problem, we introduced a novel **scale-based classification algorithm**.

We train and test an SVM model using individual scales provided from each image, thus showing their representation efficiency. The final class label assigned by our algorithm, is computed from a majority voting on per-scale decisions of each image. By doing so, non-discriminant scales are no longer adversely affecting classification. The proposed methodology provides major improvements, yielding state-of-the-art results in one 2D image classification dataset for small training sets and two 3D imaging datasets. Our proposed model can be expanded with the use of multiple per-scale SVM models. This architecture should produce improved recognition results and solve the problem of reduced performance in bigger datasets.

9.2.6 Deep learning

Our research on the implementation of 3D image data to deep learning algorithms was presented in Chapter 8.

We used a predefined CNN model for 2D image classification and modified its architecture to perform our experiments. The implementation of **depth and its combination with texture as CNN input data** showed impressive results. Proving, once again, that combination of both image information is beneficial for image

categorization. An improvement on recognition was possible by implementing our novel **data formulation of image representations** which was used to enhance CNN input data. This shows that CNN models, and deep learning algorithms in general, can be improved by the implementation of representations either as input data or as dedicated layers. We also made experiments proposing small CNN architecture changes, and their influence on overall performance. This chapter presented a study on CNN models. However, research must continue by considering dedicated CNN architectures for 3D image classification.

9.3 Future work

Concluding this thesis, we present our future plan regarding our research in 3D image categorization.

We proposed many methodologies regarding the individual processes of 3D image classification. In most cases, we presented our methods individually using the framework proposed from other methodologies in order to make a comprehensible comparison. However, competing methods were optimized for these particular frameworks, which may not favor our novel methods. For future methods, we should propose our framework, and combine each presented methodology to an optimized system.

In previous sections, we mentioned some "weak points" of our current methodologies. Our future plans include the development of improved discriminative depth features and image representations. We believe that by studying features created by CNN models, we could propose improved features for 3D image classification. Also, our scale-based classification algorithm can be improved using multiple dedicated per-scale SVM models. We also plan to incorporate this algorithm in the classification of CNN-crafted image features and representations.

However, our first priority is the continuation of our research in deep learning algorithms for 3D image classification. The design of a dedicated CNN model will contribute to 3D imaging research, either by achieving state-of-the-art results or by improving our proposed methods.

There is still more work to be done, and we will continue improving this research topic. We hope that methods and techniques presented in this thesis, will contribute to research and inspire researchers get involved in the topic of 3D imaging.

Bibliography

[1] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004.

[2] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bag of features: Spatial pyramid matching for recognizing natural scene categories. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2006.

[3] H. Larochell, D. Erhan, A. Courville, J. Bergstra, and Y. Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. In *International Conference on Machine Learning*, pages 473–480, 2007.

[4] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Region-based convolutional networks for accurate object detection and semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015.

[5] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015.

[6] S. Gupta, R. Girshick, P. Arbelaez, and J. Malik. Learning rich features from RGB-D images for object detection and segmentation. In *European Conference on Computer Vision*, 2014.

[7] K. Lai, L. Bo, X. Ren, and D. Fox. A large-scale hierarchical multi-view RGB-D object dataset. In *IEEE International Conference on Robotics and Automation*, 2011.

[8] L. Bo, X. Ren, and D. Fox. Depth kernel descriptors for object recognition. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011.

[9] L. Bo, X. Ren, and D. Fox. Unsupervised feature learning for RGB-D based object recognition. In *International Symposium on Experimental Robotics*, 2012.

[10] L. Bo, X. Ren, and D. Fox. Kernel descriptors for visual recognition. *Advances in Neural Information Processing Systems*, 2010.

[11] L. Bo, X. Ren, and D. Fox. Hierarchical matching pursuit for image classification: Architecture and fast algorithms. In *Advances in Neural Information Processing Systems*, 2011.

[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[13] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Guo. Locality-constrained linear coding for image classification. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2010.

[14] Y. Huang, K. Huang, Y. Yu, and T. Tan. Salient coding for image classification. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2011.

[15] Y. Huang, K. Huang, Y. Yu, and T. Tan. Exploring relations of visual codes for image classification. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2011.

[16] L. Bo and C. Sminchisescu. Efficient match kernel between set of features for visual recognition. *Advances in Neural Information Processing Systems*, 2009.

[17] J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2009.

[18] S. Gao, L-T. Chia, T. I.W.-H., and Z. Ren. Concurrent single-label image classification and annotation via efficient multi-layer group sparse coding. *IEEE Transactions on Multimedia*, 16(3):762–771, 2014.

[19] S. Guo and W. Z.J. An unsupervised hierarchical feature learning framework for one-shot image recognition. *IEEE Transactions on Multimedia*, 15(3):621–632, 2013.

[20] M. Jian, C. Jung, and Y. Zheng. Discriminative structure learning for semantic concept detection with graph embedding. *IEEE Transactions on Multimedia*, 16(2):413–426, 2014.

[21] Y. Zhang, Z. Jiang, and Davis L.S. Learning structured low-rank representations for image classification. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2013.

[22] J. Feng, B. Ni, Q. Tian, and S. Yan. Geometric ℓp-norm feature pooling for image classification. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2011.

[23] C. Weng, H. Wang, and J. Yuan. Learning weighted geometric pooling for image classification. In *IEEE International Conference on Image Processing*, 2013.

[24] K. Murphy, A. Torrabla, D. Eaton, and W. Freeman. Object detection and localization using local and global features. In *Lecture notes in Computer Science*, 2006.

[25] D. Crandall and D. Huttenlocher. Weakly supervised learning of part-based spatial models for visual object recognition. In *European Conference on Computer Vision*, 2006.

[26] M. H. Nguyen, L. Torresani, F. de la Torre, and C. Rother. Weakly supervised discriminative localization and classification: a joint learning process. In *IEEE International Conference on Computer Vision*, 2009.

[27] H. Billen, V. P. Namboodiri, and L. V. Gool. Object and action classification with latent variables. In *British Machine Vision Conference*, 2010.

[28] Y. Zhang and T. Chen. Weakly supervised object recognition and localization with invariant high order features. In *British Machine Vision Conference*, 2010.

[29] Y. Chai, V. Lempitsky, and A. Zimmerman. Bicos: A bi-level co-segmentation method for image classification. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2011.

[30] O. Russakovsky, Y. Lin, K. Yu, and L. Fei-Fei. Object-centric spatial pooling for image classification. In *European Conference on Computer Vision*, 2012.

[31] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded up robust features. In *European Conference on Computer Vision*, 2006.

[32] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2005.

[33] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape context. *IEEE Transactions Pattern Analysis Machine Intelligence*, 24:509–522, 2002.

[34] H. Ling and D. W. Jacobs. Shape classification using the inner-distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(2):286–299, 2007.

[35] J. Radon. The Radon transform and some of its applications ("translation of radons 1917 paper"). 1983.

[36] O.K. Al-Shaykh and J.F. Doherty. Invariant image analysis based on Radon transform and SVD. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 43(2):123–133, 1996.

[37] H. Al-Yousefi and S.S. Udpa. Recognition of arabic characters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(8):853–857, 1992.

[38] A.K. Jain. Fundamentals of digital image processing. *Upper Saddle River, NJ: Prentice-Hall*, 1989.

[39] J.L.C. Sanz, E.B. Hinkle, and A.K. Jain. Radon and projection transform-based computer vision: Algorithms, a pipeline architecture, and industrial applications. *New York: Springer-Verlag*, 1988.

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

[41] B. Russell, A. Torralba, K. Murphy, and W. T. Freeman. LabelMe: a database and web-based tool for image annotation. In *IEEE International Conference on Computer Vision*, 2007.

[42] A. Johnson and M. Herbert. Using spin images for efficient object recognition in cluttered 3D scenes. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 21(5), 1999.

[43] T. Kounalakis and N. V. Boulgouris. Sparsity-based classification using texture and depth. In *IEEE International Conference on Digital Signal Processing*, 2013.

[44] M. Dimitriou, T. Kounalakis, N. Vidakis, and G. Triantafyllidis. Detection and classification of multiple objects using an RGB-D sensor and linear spatial pyramid matching. *ELCVIA*, 12(2):78–87, 2013.

[45] M. Aharon, M. Elad, and A. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, Nov 2006.

[46] S. Sra and I. S. Dhillon. Generalized nonnegative matrix approximations with bergman divergences. *Advances in Neural Information Processing Systems*, 18, 2005.

[47] D. Cai, X. He, J. Han, and T. Huang. Graph regularized nonnegative matrix factorization for data representation. *Pattern Analysis and Machine Intelligence*, 33(8):1548–1560, 2010.

[48] G. Weiwei, H. Weidong, N.V. Boulgouris, and I. Patras. Semi-supervised visual recognition with constrained graph regularized non negative matrix factorization. In *IEEE International Conference on Image Processing*, pages 2743–2747, 2013.

[49] H. Lee, A. Battle, R. Raina, and A. Y. Ng. Efficient sparse coding algorithms. In *Advances in Neural Information Processing Systems*, 2006.

[50] L. J. Li and L. Fei-Fei. What, where and who? classifying event by scene and object recognition. In *IEEE International Conference on Computer Vision*, 2007.

[51] W. Xu, X. Liu, and Y. Gong. Document clustering based on non-negative matrix factorization. In *Research and Development in Information Retrieval*, pages 267–273, 2003.

[52] D.D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems*, 2001.

[53] D. Donoho. For most large underdetermined systems of linear equations the minimal l_1-norm solution is also the sparsest solution. *Comm. Pure and Applied Math.*, 59(6):797–829, 2006.

[54] J. Sivic and A. Zisserman. Video google: a text retrieval approach to object matching in videos. In *IEEE International Conference on Computer Vision*, pages 1470–1477, 2003.

[55] K. Grauman and T. Darrell. The pyramid match kernel: discriminative classification with sets of image features. In *IEEE International Conference on Computer Vision*, volume 2, pages 1458–1465, 2005.

[56] C. Cortes and V.N. Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 1995.

[57] L. Bottou, C. Cortes, J.S. Denker, H. Drucker, I. Guyon, L.D. Jackel, Y. LeCun, U.A. Muller, E. Sackinger, P. Simard, and V. Vapnik. Comparison of classifier methods: a case study in handwritten digit recognition. In *Proceedings of the IAPR International Conference on Pattern Recognition*, volume 2, pages 77–82, 1994.

[58] S. Knerr, L. Personnaz, and G. Dreyfus. Single-layer learning revisited: a stepwise procedure for building and training a neural network. In *Neurocomputing*, volume 68 of *NATO ASI Series*, pages 41–50. Springer Berlin Heidelberg, 1990.

[59] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[60] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and Fei-Fei Li. Imagenet large scale visual recognition challenge. *CoRR*, 2014.

[61] Y. Taigman, Y. Ming, M. Ranzato, and L. Wolf. Deepface: Closing the gap to human-level performance in face verification. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2014.

[62] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *Pattern Analysis and Machine Intelligence*, 35(8):1915–1929, 2013.

[63] G. E. Hinton and S. Osindero. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:2006, 2006.

[64] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *International Conference on Machine Learning*, pages 609–616, 2009.

[65] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, 1980.

[66] L. Fei-Fei and P. Perona. A bayesian hierarchical model for learning natural scene categories. pages 524–531, 2005.

[67] G. Griffin, AD. Holub, and P. Perona. The Caltech-256. Caltech technical report, 2004.

[68] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories. *IEEE International Conference on Computer Vision and Pattern Recognition Workshop on Generative-Model Based Vision*, 2004.

[69] M. Everingham, L. Gool, C. Williams, J. Winn, and A. Zisserman. The PASCAL visual object classes challenge 2007 (VOC2007) results. Technical report, 2007.

[70] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database.

[71] A. Krizhevsky. *Learning Multiple Layers of Features from Tiny Images*. PhD thesis, Toronto University, 2009.

[72] P. Moreels and P. Perona. Evaluation of features detectors and descriptors based on 3D objects. In *International Journal of Computer Vision*, 2006.

[73] M. Goesele, N. Snavely, B. Curless, H. Hoppe, and S. M. Seitz. Multi-view stereo for community photo collections. In *IEEE International Conference on Computer Vision*, 2007.

[74] S. Savarese and L. Fei-Fei. 3D generic object categorization, localization and pose estimation. In *IEEE International Conference in Computer Vision*, 2007.

[75] A. Singh, J. Sha, K.S. Narayan, T. Achim, and P. Abbeel. A large-scale 3D database of object instances. In *IEEE International Conference on Robotics and Automation*, 2014.

[76] Willow Garage Object Recognition Challenge.

[77] A. Richtsfeld, T. Morwald, J. Prankl, M. Zillich, and M. Vincze. Segmentation of unknown objects in indoor environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012.

[78] A. Aldoma, F. Tombari, L. Di Stefano, and M. Vincze. A global hypothesis verification method for 3D object recognition. In *European Conference on Computer Vision*, 2012.

[79] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab. Model-based training, detection and pose estimation of texture-less 3D objects in heavily cluttered scenes. In *Asian Conference on Computer Vision*, 2012.

[80] A. Janoch, S. Karayev, Y. Jia, J.T. Barron ana M. Fritz, K. Saenko, and T. Darrell. A category-level 3-D object dataset: Putting the kinect to work. In *IEEE International Conference on Computer Vision Workshop on Consumer Depth Cameras in Computer Vision*, 2011.

[81] H. S. Koppula, A. Anand, T. Joachims, and A. Saxena. Semantic labeling of 3D point clouds for indoor scenes. In *Neural Information Processing Systems*, 2011.

[82] N. Silberman and R. Fergus. Indoor scene segmentation using a structured light sensor. In *IEEE International Conference on Computer Vision Workshop*, 2011.

[83] N. Silberman, P. Kohli, D. Hoiem, and R. Fergus. Indoor segmentation and support inference from RGBD images. In *European Conference on Computer Vision*, 2012.

[84] J. Mason, B. Marthi, and R. Parr. Object disappearance for object discovery. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012.

[85] J. Xiao, A. Owens, and A. Torralba. SUN3D: A database of big spaces reconstructed using SfM and object labels. In *IEEE International Conference on Computer Vision*, 2013.

[86] A. Karpathy, S. Miller, and L. Fei-Fei. Object discovery in 3D scenes via shape analysis. In *IEEE International Conference on Robotics and Automation*, 2013.

[87] K. Lai, L. Bo, and D. Fox. Unsupervised feature learning for 3D scene labeling. In *IEEE International Conference on Robotics and Automation*, 2014.

[88] F. Pomerleau, S. Magnenat, F. Colas, M. Liu, and R. Siegwart. Tracking a depth camera: Parameter exploration for fast icp. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011.

[89] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of RGB-D SLAM systems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012.

[90] D. Gossow, D. Weikersdorfer, and M. Beetz. Distinctive texture features from perspective-invariant keypoints. In *IEEE International Conference on Pattern Recognition*, 2012.

[91] B. Zeisl, K. KÃűser, and M. Pollefeys. Automatic registration of RGB-D scans via salient directions. In *IEEE International Conference in Computer Vision*, 2013.

[92] Q. Y. Zhou and V. Koltun. Dense scene reconstruction with points of interest. In *Special Interest Group on GRAPHics and Interactive Techniques*, 2013.

[93] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon. Scene coordinate regression forests for camera relocalization in RGB-D images. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2013.

[94] A. Handa, T. Whelan, J.B. McDonald, and A.J. Davison. A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM. In *IEEE International Conference on Robotics and Automation*.

[95] S. Song and J. Xiao. Tracking revisited using RGBD camera: Unified benchmark and baselines. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2013.

[96] J. Sung, C. Ponce, B. Selman, and A. Saxena. Unstructured human activity detection from RGBD images. In *IEEE International Conference on Robotics and Automation*, 2012.

[97] M. Luber, L. Spinello, and K. O. Arras. People tracking in RGB-D data with on-line boosted target models. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011.

[98] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu. Human3.6M: Large scale datasets and predictive methods for 3D human sensing in natural environments. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 36(7), 2014.

[99] G. Fanelli, M. Dantone, J. Gall, A. Fossati, and L. Van Gool. Random Forests for real time 3D face analysis. *International Journal of Computer Vision*, 101(3):437–458, 2013.

[100] B. I. Barbosa, M. Cristani, A. Del Bue, L. Bazzani, and V. Murino. Re-identification with RGB-D sensors. In *First International Workshop on Re-Identification*, 2012.

[101] S. Fothergill, H. M. Mentis, P. Kohli, and S. Nowozin. Instructing people for training gestural interactive systems. In J. A. Konstan, E. H. Chi, and K. Höök, editors, *CHI*, pages 1737–1746. ACM, 2012.

[102] L. Liu and L. Shao. Learning discriminative representations from RGB-D video data. In *International Joint Conference on Artificial Intelligence*, 2013.

[103] E. Nesli and M. SÃľbastien. Spoofing in 2D face recognition with 3D masks and anti-spoofing with kinect. 2013.

[104] R. Min, N. Kose, and J. L. Dugelay. KinectFaceDB: A kinect database for face recognition. *Systems, Man, and Cybernetics: Systems, IEEE Transactions on*, 44(11):1534–1548, 2014.

[105] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*, chapter 6.3.1, pages 303–304. Second edition.

[106] H. Samet and M. Tamminen. Efficient component labeling of images of arbitrary dimension represented by linear bintrees. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 1988.

[107] F. Korn, B. U. Pagel, and C. Faloutsos. On the "dimensionality curse" and the "self-similarity blessing". *IEEE Transactions on Knowledge and Data Engineering*, 13:96–111, 2001.

[108] T. Kounalakis and N.V. Boulgouris. Content-adaptive pyramid representation for three dimensional image classification. IEEE Transactions in Multimedia (UNPUBLISHED: under reviewing process).

[109] Z. Jiang, Z. Lin, and Davis L.S. Learning a discriminative dictionary for sparse coding via label consistent K-SVD. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2011.

[110] A. Shrivastava, H. V. Nguyen, V. .M. Patel, and R. Chellappa. Design of non-linear discriminative dictionaries for image classification. In *ACCV*, 2012.

[111] G.Th. Papadopoulos, C. Saathoff, H.J. Escalante, V. Mezaris, I. Kompatsiaris, and M.G. Strintzis. A comparative study of object-level spatial context techniques for semantic image analysis. *Computer Vision and Image Understanding*, 115(9):1288–1307, 2011.

[112] S. Nikolopoulos, G.Th. Papadopoulos, I. Kompatsiaris, and I Patras. An evidence-driven probabilistic inference framework for semantic image understanding. In *Machine Learning and Data Mining in Pattern Recognition*, volume 5632, pages 525–539. 2009.

[113] C. Carson, S. Belongie, H.Greenspan, and J.Malik. Blobworld: image segmentation using expectation-maximization and its application to image querying. *Pattern Analysis and Machine Intelligence*, 24(8):1026–1038, 2002.

[114] S. Gould, R. Fulton, and D. Koller. Decomposing a scene into geometric and semantically consistent regions. In *IEEE International Conference on Computer Vision*, 2009.

[115] M. P. Kumar and D. Koller. Efficiently selecting regions for scene understanding. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 3317–3224, 2010.

[116] R. Socher, C. C. Y. Lin, A. Y. Ng, and C. D. Manning. Parsing natural scenes and natural language with recursive neural networks. In *International Conference on Machine Learning*, 2011.

[117] D. Hoiem, A. A. Efros, and M. Hebert. Recovering surface layout from an image. In *International Journal of Computer Vision*, volume 75, pages 151–172, 2007.

[118] J. Hespanha. An efficient matlab algorithm for graph partitioning. Technical report, University of California, 2004.

[119] T. Kounalakis and N.V. Boulgouris. Classification of 2D and 3D images using pyramid scale decision voting. In *IEEE International Conference on Image Processing*, 2014.

[120] L. J. Li, R. Socher, and F. F. Li. Towards total scene understanding: Classification, annotation and segmentation in an automatic framework. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 2036–2043, 2009.

[121] A. Levinshtein, A. Stere, K. N. Kutulakos, D. J. Fleet, S. J. Dickinson, and K. Siddiqi. Turbopixels: Fast superpixels using geometric flows. *IEEE Transactions on Pattern Analysis ans Machine Intelligence*, 31(12):2290–2297, 2009.

[122] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. In *International Journal of Computer Vision*, volume 59, pages 167–181, 2004.

[123] J. Stücker and S. Behnke. Combining depth and color cues for scale- and viewpoint-invariant object segmentation and recognition using random forests. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010.

[124] X. C. Lian, Z. Li, C. Wang, B. L. Lu, and L. Zhang. Probabilistic models for supervised dictionary learning. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2010.

[125] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[126] A. Vedaldi and K. Lenc. Matconvnet – convolutional neural networks for matlab. *CoRR*, 2014.

[127] K. Lai, L. Bo, X. Ren, and D. Fox. Sparse distance learning for object recognition combining RGB and depth information. In *IEEE International Conference on Robotics and Automation*, 2011.