ABSTRACT

Title of dissertation: Anomaly Detection in Noisy Images

Xavier Gibert Serra, Ph.D. Examination, Fall 2015

Dissertation directed by: Professor Rama Chellappa
Department of Electrical and Computer Engineering

Finding rare events in multidimensional data is an important detection problem that has applications in many fields, such as risk estimation in insurance industry, finance, flood prediction, medical diagnosis, quality assurance, security, or safety in transportation. The occurrence of such anomalies is so infrequent that there is usually not enough training data to learn an accurate statistical model of the anomaly class. In some cases, such events may have never been observed, so the only information that is available is a set of normal samples and an assumed pairwise similarity function. Such metric may only be known up to a certain number of unspecified parameters, which would either need to be learned from training data, or fixed by a domain expert. Sometimes, the anomalous condition may be formulated algebraically, such as a measure exceeding a predefined threshold, but nuisance variables may complicate the estimation of such a measure. Change detection methods used in time series analysis are not easily extendable to the multidimensional case, where discontinuities are not localized to a single point. On the other hand, in higher dimensions, data exhibits more complex interdependencies, and there is redundancy that could be exploited to adaptively model the normal data.

In the first part of this dissertation, we review the theoretical framework for anomaly detection in images and previous anomaly detection work done in the context of crack detection and detection of anomalous components in railway tracks. In the second part, we propose new anomaly detection algorithms. The fact that curvilinear discontinuities in images are sparse with respect to the frame of shearlets, allows us to pose this anomaly detection problem as basis pursuit optimization. Therefore, we pose the problem of detecting curvilinear anomalies in noisy textured images as a blind source separation problem under sparsity constraints, and propose an iterative shrinkage algorithm to solve it. Taking advantage of the parallel nature of this algorithm, we describe how this method can be accelerated using graphical processing units (GPU). Then, we propose a new method for finding defective components on railway tracks using cameras mounted on a train. We describe how to extract features and use a combination of classifiers to solve this problem. Then, we scale anomaly detection to bigger datasets with complex interdependencies. We show that the anomaly detection problem naturally fits in the multitask learning framework. The first task consists of learning a compact representation of the good samples, while the second task consists of learning the anomaly detector. Using deep convolutional neural networks, we show that it is possible to train a deep model with a limited number of anomalous examples. In sequential detection problems, the presence of time-variant nuisance parameters affect the detection performance. In the last part of this dissertation, we present a method for adaptively estimating the threshold of sequential detectors using Extreme Value Theory on a Bayesian framework. Finally, conclusions on the results obtained are provided, followed by a discussion of possible future work.

Anomaly Detection in Noisy Images

by

Xavier Gibert Serra

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2015

Advisory Committee:
Professor Rama Chellappa, Chair/Advisor
Professor Piya Pal
Professor Shuvra Bhattacharyya
Professor Vishal M. Patel
Professor Amitabh Varshney, Dean's Representative

Dedication

To my daughter Mar for patiently waiting for this work to be completed.

Acknowledgments

This dissertation would not have been possible without the help and contributions from many people. Please forgive me if I inadvertently left somebody out.

First and foremost, I would like to thank my advisor, Professor Rama Chellappa for his invaluable support and guidance. His patience and perseverance have been and continue to be a source of inspiration to me. I appreciate the trust that he has put on me while working with him as his teaching assistant as well as his research assistant. I am also thankful to Professor Vishal Patel for introducing me to the field of sparse representations and for helping me during my struggles. I am also thankful to Professor Demetrio Labate at UT Houston for helping me write chapter 3. I am also thankful to Mark Smith and Felipe Arrate for introducing me to the field of nuclear and medical imaging.

I would like to acknowledge the National Railroad Passenger Corporation (Amtrak) for making the research presented in this dissertation possible. In particular, I would like to thank Michael Trosino, Michael Craft, Joe Smack and Joe Mascara for granting permission to use the railway images that made this work possible and for providing guidance and assessing the practical feasibility of the research presented in this volume.

I would like to acknowledge the invaluable financial support from the Federal Railroad Administration, without which this work would have not been possible. I am specially grateful to Leith Al-Nazer, FRA's Technical Representative, for his guidance and valuable comments during this project, as well as Cam Stuart and Gary Carr for managing the overall research program.

I would also like to acknowledge the financial support from the University of Mary-

land seed grant program, which gave me the opportunity to do collaborative work with the University of Maryland Baltimore.

I would like thank and acknowledge ENSCO, Inc. for all the opportunities that I had while working for such a superb company, specially for providing a path to US citizenship by sponsoring my H1B visa, and by allowing me to work on cutting-edge technologies. I am specially specially thankful to ENSCO for making me part of a world-class team and introducing me to the field of railway inspection. First and foremost, I would like to acknowledge my supervisors, William Jordan, Antonio Aquino, Boris Nejikovsky, and David Ford, who have provided guidance during my tenure at ENSCO. I am specially thankful to Antonio Aquino for helping me improve my software design skills. I would like to thank Gary Carr, Boris Nejikovsky, William Jordan, Christian Diaz, Andie Berry, David McNew, Zhipeng Liu, Ashutosh Morde, Amaury Rolin, Harkirat Sahambi, Jeff Henderson, and Cindy Scott for their contributions in making vision-based railway track inspection a reality. I would like to specially thank Gary Carr for inviting me to participate in the 2005 DARPA Grand Challenge as a member of Team ENSCO, where I learned a lot about time-constrained detection in outdoor environments. I would also like to thank Paul Broome and Greg Young for providing the financial assistance that made the ENSCO participation in the Grand Challenge a reality. I would like to thank Eric Sherrock teaching me how to write government proposals.

I would like to thank and acknowledge SET/Leidos for letting me to participate in the UMD Team Internship Program during summer of 2014. I am specially thankful to my mentor Yanlin Guo for her guidance and patience, as well as to Rafael Alonso, Michael Zuniga, and S. Gulu Gambhir for making this research collaboration possible.

I would like to thank all the students, faculty and staff at the Center for Automation Research (CFAR) and the University of Maryland Institute for Advanced Computer Studies (UMIACS) for making this amazing research environment that has been producing top quality research for more than fifty years. I have special gratitude to Daniel Bogachek who helped me with some of the experiments in chapter 5. I am also specially thankful to David Doermann and Huiping Li for being my mentors at CFAR between 2001 and 2003. I still have fond memories of those early years at the Language and Media Processing Lab.

I would like to thank my family for their sacrifices that made this dissertation possible. I specially thank my lovely wife Elena for her support and patience, our daughter Mar, and my parents Dolors and Llorenç for their encouragement and support.

Finally, I thank god.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

1.1 Motivation

1.1.1 Problem Description

Anomaly detection is the problem of detecting patterns on data that do not conform to an established normal behavior [1]. What constitutes an anomaly is very subjective and a mathematically rigorous definition can only be provided under specific contexts. In this dissertation we address the problem of finding anomalies in noisy images.

The techniques described in this dissertation can be used to find flaws in a collection of images with similar content. In particular, we have a sequence of noisy images and we want to determine whether they are normal or whether they contain flaws. Usually, many of these images are normal (without flaws), but a small number of them may contain anomalies. As these two classes are highly unbalanced, and we do not have prior knowledge about their relative frequencies, standard discriminative learning machines will exhibit poor performance. For example, a standard support vector machine would produce a solution where almost all the anomalies correspond to support vectors, resulting in overfitting. A further twist would be when the class of anomalies is not homogeneous. In this case, there are several subclasses of flaws and these subclasses are also unbalanced relative to each other, and no prior knowledge about their relative frequencies is available. The extreme cases are when there is a very small number of examples in the training set (one-shot learning) [2], or no examples at all (zero-shot learning) [3]. Both scenarios are common in anomaly detection problems (even in the context of big data), so the inference techniques that will be described in this dissertation will need to handle them.

In some situations there may be additional privileged information only available during training time, but not at testing time. One example in medical imaging, is the problem of estimating an electrophysiological cardiac map from magnetic resonance imaging (MRI), positron emission tomography (PET), single photon emission computed tomography (SPECT) images, or any combination of image modalities [4]. In this problem, there may be a limited number of real electrophysiological measurements available for patients on the training set with corresponding PET or SPECT images, but only PET/SPECT images at test time. In addition, there may be instances where a decision cannot be made due to poor image quality or severe occlusion. These cases require that a "reject option" be taken into consideration.

1.1.2 Challenges

Images collected in outdoor environments by an unattended camera are subject to large variations due to illumination and weather. The presence of clutter creates a situation where the signal-to-noise ratio can be negative. Also, high-speed imaging applications demand that the images be collected with very short exposure times that result in weak signals. In the nuclear medical imaging field, due to limits on safe radiation levels that the human body can tolerate, measured signals are also weak despite using longer exposure times. In these cases, it is necessary to enforce application-specific prior knowledge to produce more accurate estimates. Priors used in image processing include smoothness (enforced through total variation regularization) or sparsity with respect to a dictionary that is known to produce compact representations of the image content.

1.2 Proposed Algorithms and their Contributions

In this section we introduce the algorithms and methods proposed in this dissertation and their key contributions.

1. **Discrete Shearlet Transform on GPU with Applications in Anomaly Detection and Denoising:**

   Shearlets have emerged in recent years as one of the most successful methods for the multiscale analysis of multidimensional signals. Unlike wavelets, shearlets form a pyramid of well-localized functions defined not only over a range of scales and locations, but also over a range of orientations and with highly anisotropic supports. As a result, shearlets are much more effective than traditional wavelets in handling the geometry of multidimensional data and this was exploited in a wide range of applications from image and signal processing. However, despite their desirable properties, the wider applicability of shearlets is limited by the computational complexity of current software implementations.

   Contributions: Our contributions have been an efficient CUDA implementation of the shearlet transform and demonstrating its applicability for the problem of detecting cracks on textured images.

2. **Robust Fastener Detection for Autonomous Visual Railway Track Inspection:**

   Fasteners are critical railway components that maintain the rails in a fixed position. Their failure can lead to train derailments due to gage widening or wheel climb, so their condition needs to be periodically monitored. We propose a new method for fastener detection by 1) carefully aligning the training data, 2) reducing intra-class variation, and 3) bootstrapping difficult samples to improve the classification margin.

   Contributions: Several computer vision methods have been proposed in the literature for track inspection applications. However, these methods are not robust to clutter and background noise present in the railroad environment. Using the histogram of oriented gradients features and a combination of linear SVM classifiers, this algorithm can inspect ties for missing or defective rail fastener problems with a probability of detection of 98% and a false alarm rate of 1.23% on a new dataset of 85 miles of concrete tie images collected in the US Northeast Corridor (NEC) between 2012 and 2013. To the best of our knowledge, detection result on this dataset of 203,287 crossties is the largest ever reported in the literature.

3. **Material Classification and Semantic Segmentation of Railway Track Images with Deep Convolutional Neural Networks:**

   The condition of railway tracks needs to be periodically monitored to ensure passenger safety. Cameras mounted on a moving vehicle such as a hi-rail vehicle or a geometry inspection car can generate large volumes of high resolution images.

Extracting accurate information from those images has been challenging due to the presence of clutter in the railroad environment.

Contributions: We describe a novel approach to visual track inspection using material classification and semantic segmentation with Deep Convolutional Neural Networks (DCNN). We show that DCNNs trained end-to-end for material classification are more accurate than shallow learning machines with hand-engineered features and are more robust to noise. Our approach results in a material classification accuracy of 93.35% using 10 classes of materials. This allows for the detection of crumbling and chipped tie conditions at detection rates of 86.06% and 92.11%, respectively, at a false positive rate of 10 FP/mile on the 85-mile Northeast Corridor (NEC) 2012-2013 concrete tie dataset.

4. **Deep Multi-task Learning for Railway Track Inspection:**

   Automated track inspection using computer vision and pattern recognition methods have the potential to improve safety by allowing for more frequent inspections while reducing human errors. Achieving full automation is still very challenging due to the number of different possible failure modes as well as the broad range of image variations than can potentially trigger false alarms. Also, the number of defective components is very small, so not many training examples are available for the machine to learn a robust anomaly detector.

   Contributions: We show that detection performance can be improved by combining multiple detectors within a multi-task learning framework. We show that this approach results in better accuracy in detecting defects on railway ties and fasteners.

5. **Sequential Anomaly Detection with Adaptive Thresholding via Extreme Value Theory:**

   Anomaly detection is usually applied to sequences of images. There are nuisance variables, such as changes in illumination as well as shifts in the noise and clutter distributions. Methods that adapt each image independently from the others do not exploit such local dependencies. Extreme value theory provides the foundation for adaptive thresholding. In this chapter we use EVT within a Bayesian framework to optimally adjust the sensitivity of anomaly detectors.

   Contributions: We show that by approximating the lower tail of the PDF of the scores with an Exponential distribution (a special case of the Generalized Pareto distribution), and using the Gamma conjugate prior learned from the training data, it is possible to reduce the variability in false alarm rate across different sequences and improve the overall performance. This method has shown to increase the defect detection rate of railway fasteners in the presence of clutter (at PFA 0.1%) from 95.40% to 99.26%.

1.3 Organization

This dissertation is organized as follows. In chapter 2, we review the literature in anomaly detection in images and put them in the context of vision-based automated railway inspection. In chapter 3, an algorithm for separating cracks from background texture using mutually incoherent dictionaries of shearlets and isotropic wavelets is presented, followed by a fast implementation using GPUs [5]. In chapter 4, we describe a detector for finding defective fasteners on railway tracks [6]. In chapter 5, we introduce deep convolutional neural network and we describe a new approach for performing material classification and semantic segmentation on track inspection images [7]. This problem is formulated within the multi-task learning framework [8] and we show that by learning a shared representation between this task and the task in chapter 4, both tasks achieve better performance. In chapter 6 we introduce an adaptive thresholding method for anomaly detection on sequences of images [9]. Chapter 7 concludes the dissertation and discusses possible future research directions that could further extend this work. In appendix A we introduce CardioViewer, a tool that we created to study the applicability of statistical anomaly detection methods in the field of medical imaging.

Chapter 2: Background

2.1 Anomaly Detection

The theory of change detection in sequential data has been well established for years. [10]. If samples in the normal class are independent and identically distributed (iid), it can be shown that the CUSUM algorithm provides the optimal detection rate for a given false alarm rate. For real-time applications, the theory of quickest detection [11], provides a framework that can be used to detect changes in the distribution of a signal within the shortest possible delay. In this dissertation, we address the problem of anomaly detection in images and videos. Due to the multidimensionality nature of the data, anomaly detection in images and videos require formulating the problem differently than classical formulations used for time series analysis. For example, causality is irrelevant in images. Moreover, invariance to affine transformations in the data may, indeed, be desirable.

Assuming that the normal samples are iid, distance-based methods, such as the one-class SVM or the k-nearest neighbor classifier, may suffice. However, in practice, the data is neither independent nor stationary, so the distribution of image features may shift over sample index. Therefore, methods that do not take context into account tend to produce bursts of false detections when the probability distribution of the data shifts away from the distribution that generated the samples used to train the statistical model. This phenomenon may be mitigated through proper normalization of the images. However, blindly normalizing the images may eliminate evidence about the anomalous event. Moreover, the anomalous data may not be linearly separable from the normal one, so in these cases it would be necessary to use a non-Euclidean metric, which is usually induced by a properly chosen kernel function. In some applications the data may be embedded in a lower dimensional manifold, so the use of a geodesic distance may produce more accurate results.

Also, anomaly detection is closely related to another detection problem, which is saliency detection. For instance, anomalous data is defined with respect to normal data, in the similar way salient regions in an image are defined wrt non-salient ones. For example, Xu et al. used saliency for detecting cracks on pavement images [12]. However, while in the saliency detection problem spatial locality is implied, anomaly detection can be formulated with or without local (dis)similarity constraints.

2.2 Railway Track Inspection

Monitoring the condition of railway components is essential to ensure train safety, especially on High Speed Rail (HSR) corridors. Amtrak's recent experience with concrete ties has shown that they have different kind of problems than wooden ties [13]. Although concrete ties have life expectancies of up to 50 years, they may fail prematurely for a variety of reasons:

- Alkali-silica reaction (ASR), which is a chemical reaction between cement alkalis a non-crystalline (amorphous) silica that forms alkali-silica gel at the aggregate surface [14]. These reaction rims have a very strong affinity with water and have a tendency to swell. These compounds can produce internal pressures that are strong enough to create cracks, allowing moisture to penetrate, and thus accelerating the rate of deterioration.

- Delayed Ettringite Formation (DEF) is a type of internal sulfate attack that occurs in concrete that has been cured at excessively high temperatures [15].
- In addition to ASR and DEF, ties can also develop fatigue cracks due to normal traffic or by being impacted by flying debris or track maintenance machinery. Once small cracks develop, repeated cycles of freezing and thawing will eventually lead to bigger defects.

Fasteners maintain gage by keeping both rails firmly attached to the crossties. According to the Federal Railroad Administration (FRA) safety database[1], in 2013, out of 651 derailments due to track problems, 27 of them were attributed to gage widening caused by defective spikes or rail fasteners, and another 2 to defective or missing spikes or rail fasteners.

Also, in the United States, regulations enforced by the FRA[2] prescribe visual inspection of high-speed rail tracks with a frequency of once or twice per week, depending on the class of track (which specifies maximum authorized speeds for both freight and passenger trains). These manual inspections are currently performed by railroad person-

[1] http://safetydata.fra.dot.gov

[2] 49 CFR 213 – Track Safety Standards

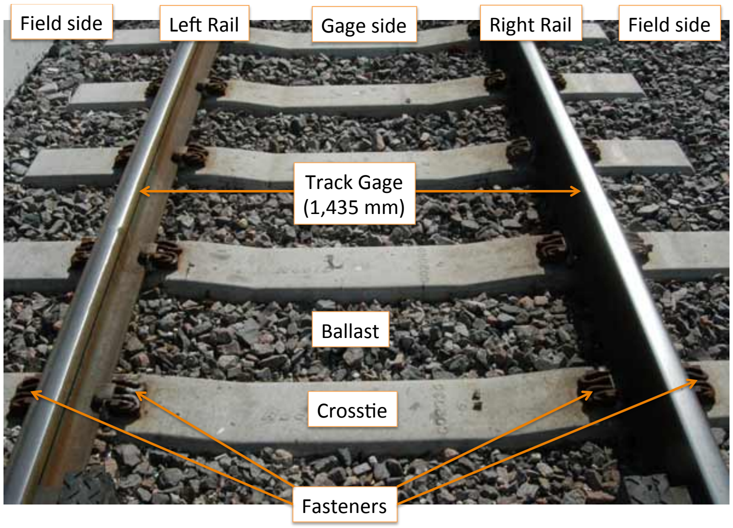


Figure 2.1: Definition of basic track elements.

nel, either by walking on the tracks or by riding a hi-rail vehicle at very low speeds. However, such conventional visual inspections of mainlines are subjective and do not produce an auditable visual record. In addition, railroads usually perform automated track inspections with specialized track geometry measurement vehicles at intervals of 30 days or less between inspections. These automated inspections can directly detect gage widening conditions. However, it is preferable to find fastening problems before they develop into gage widening conditions. The locations and names of the basic track elements mentioned in this chapter are shown in Figure 2.1.

Since the pioneering work by Trosino *et al.* [16, 17], machine vision technology has been gradually adopted by the railway industry as a track inspection technology. Those first generation systems were capable of collecting images of the railway right of way and storing them for later review, but they did not facilitate any automated detection. As faster processing hardware became available, several vendors began to introduce vision systems with increasing automation capabilities.

In [18, 19], Marino *et al.* describe their VISyR system, which detects hexagonal-headed bolts using two 3-layer neural networks (NN) running in parallel. Both networks take the 2-level discrete wavelet transform (DWT) of a 24×100 pixel sliding window (their images use non-square pixels) as an input to generate a binary output indicating the presence of a fastener. The difference is that the first NN uses Daubechies wavelets, while the second one uses Haar wavelets. This wavelet decomposition is equivalent to performing edge detection at different scales with two different filters. Both neural networks are trained with the same examples. The final decision is made using the maximum output of each neural network.

In [20, 21], Gibert *et al.* describe their VisiRail system for joint bar inspection. The system is capable of collecting images on each rail side, and finding cracks on joint bars using edge detection and a Support Vector Machine (SVM) classifier that analyzes features extracted from these edges. In [22], Babenko describes a fastener detection method based on a convolutional filter bank that is applied directly to intensity images. Each type of fastener has a single filter associated with it, whose coefficients are calculated using an illumination-normalized version of the Optimal Trade-off Maximum Average Correlation Height (OT-MACH) filter [23]. This approach allowed accurate fastener detection and localization and achieved over 90% fastener detection rate on a dataset of 2,436 images. However, the detector was not tested on longer sections of track. In [24], Resendiz *et al.* use texture classification via a bank of Gabor filters followed by an SVM to determine the location of rail components such as crossties and turnouts. They also use the MUSIC algorithm to find spectral signatures to determine expected component locations. In [25], Li *et al.* describe a system for detecting tie plates and spikes. Their method, which is described in more detail in [26], uses an AdaBoost-based object detector [27] with a model selection mechanism that assigns the object class that produces the highest number of detections within a window of 50 frames. Table 2.1 summarizes several systems reported in the literature.

Recent advances in CMOS imaging technology, have resulted in commercial-grade line-scan cameras that are capable of capturing images at resolutions of up to 4,096$\times$2 and line rates of up to 140 KHz. At the same time, high-intensity LED-based illuminators with life expectancies in the range of 50,000 hours have become available. This technology enables virtually maintenance-free operation over several months. Therefore, technology that enables autonomous visual track inspection from an unattended vehicle (such as a passenger train) may become a reality in the not-too-distant future.

Table 2.1: Evolution of automated visual railway component inspection methods.

Authors	Year	Components	Defects	Features	Decision methods
Stella *et al.* [19,28,29]	2002–09	Fasteners	Missing	DWT	3-layer NN
Singh *et al.* [30]	2006	Fasteners	Missing	Edge density	Threshold
Hsieh *et al.* [31]	2007	Fasteners	Broken	DWT	Threshold
Gibert *et al.* [20,21]	2007–08	Joint Bars	Cracks	Edges	SVM
Babenko [22]	2008	Fasteners	Missing/Defective	Intensity	OT-MACH corr.
Xia *et al.* [32]	2010	Fasteners	Broken	Haar	Adaboost
Yang *et al.* [33]	2011	Fasteners	Missing	Direction Field	Correlation
Resendiz *et al.* [24]	2013	Ties/Turnouts	–	Gabor	SVM
Li *et al.* [25]	2014	Tie plates	Missing spikes	Lines/Haar	Adaboost
Feng *et al.* [34]	2014	Fasteners	Missing/Defective	Haar	PGM
Gibert *et al.* [5]	2014	Concrete ties	Cracks	DST	Iterative shrinkage
Khan *et al.* [35]	2014	Fasteners	Defective	Harris-Stephen, Shi-Tomasi	Matching errors
Gibert *et al.* [6]	2015	Fasteners	Missing/Defective	HOG	SVM
Gibert *et al.* [7]	2015	Concrete ties	Tie Condition	Intensity	Deep CNN

Chapter 3: Discrete Shearlet Transform on GPU with Applications in Anomaly Detection and Denoising

3.1 Introduction

During the last decade, a new generation of multiscale systems has emerged which combines the power of the classical multiresolution analysis with the ability to process directional information with very high efficiency. Some of the most notable examples of such systems include the *curvelets* [36], the *contourlets* [37] and the *shearlets* [38]. Unlike classical wavelets, the elements of such systems form a pyramid of well localized waveforms ranging not only across various scales and locations, but also across various orientations and with highly anisotropic shapes. Thanks to their richer structure, these more sophisticated multiscale systems are able to overcome the poor directional sensitivity of traditional multiscale systems and have been used to derive several state-of-the-art algorithms in image and signal processing (cf. [39, 40]).

Shearlets, in particular, offer a unique combination of very remarkable features: they have a simple and well understood mathematical structure derived from the theory of affine systems [38, 41], they provide optimally sparse representations, in a precise sense, for a large class of images and other multidimensional data where wavelets are suboptimal

[42, 43] and the directionality is controlled by shear matrices rather than rotations. This last property, in particular, enables a unified framework for continuum and discrete setting since shear transformations preserve the rectangular lattice and is an advantage in deriving faithful digital implementations [44, 45].

The shearlet decomposition has been successfully employed in many problems from applied mathematics and signal processing, including decomposition of operators [46], inverse problems [47, 48], edge detection [49–51], image separation [52] and image restoration [53–55]. However, one major bottleneck to the wider applicability of the shearlet transform is that current discrete implementations tend to be very time consuming making its use impractical for large data sets and for real-time applications. For instance, the current (CPU-based) MATLAB implementation [1] of the 2D shearlet transform, run on a typical desktop PC, takes about two minutes to denoise a noisy image of size 512×512 [44, 56]. The running time of the current (CPU-based) MATLAB implementation of the 3D shearlet transform for denoising a video sequence of size 192^3 is about five minutes [55]. Running times for alternative shearlet implementations from Shearlab [45] as well as for the current implementation of the curvelet transform [57] are comparable.

In recent years, General Purpose Graphics Processing Units (GPGPUs) have become ubiquitous not only on High Performance Computing (HPC) clusters, but also on workstations. For example, Titan, which was until recently the world's fastest supercomputer, contains 18,688 NVIDIA Tesla K20X GPUs. These GPUs provide about 90% of

[1] Note that this code also includes some C routines to speed-up the computation time. This is true both for the 2D and 3D implementations.

Titan's peak computing performance, which is greater than 20 PetaFLOPS (quadrillion floating point operations per second). Due to their energy efficiency and capabilities, GPGPUs are also becoming mainstream on mobile platforms, such as iOS and Android devices. There are two main architectures for GPGPU computing: CUDA and OpenCL. CUDA was designed by NVIDIA, and has been around since 2006. OpenCL was originally designed by Apple, Inc, and was introduced in 2008. OpenCL is an open standard maintained by the Khronos Group, whose members include Intel, AMD, NVIDIA, and many others, so it has broader industry acceptance than any other architecture. In 2009, Microsoft introduced DirectCompute as an alternative architecture for GPGPU computing, which is only available in Windows Vista and later. OpenCL has been designed to provide the developer with a common framework for doing computation on heterogeneous devices. One of the advantages of OpenCL is that it can potentially support any computing device, such as CPUs, GPUs, and FPGAs, as long as there is an OpenCL compiler available for such processor. NVIDIA provides CUDA/OpenCL drivers, libraries and development tools for the three major Operating Systems (Linux, Windows and Mac OS X), while AMD/ATI$^{\text{TM}}$and Intel provide OpenCL drivers and tools for their respective GPUs.

The objective of this chapter is to introduce and demonstrate a new implementation of the 2D and 3D discrete shearlet transform which takes advantage of the computational capabilities of the Graphics Processing Unit (GPU). To demonstrate the effectiveness of the proposed implementations, we will illustrate its application on problems of image and video denoising and on a problem of feature recognition aiming at crack detection of railway components. In particular, we will show that our new implementation takes about 40 milliseconds to denoise an image of size 512×512, which is a $233\times$ speed-up compared to single core CPU, and about 3 seconds to denoise a video of size 192^3, which is a $551\times$ speed-up compared to single core CPU.

The organization of the chapter is as follows. In Section 3.2, we recall the construction of 2D and 3D shearlets. Next, in Section 3.3, we present our implementation of the discrete shearlet transform and, in Section 3.4, we benchmark our implementation using three specific applications. Finally, concluding remarks and future work are discussed in Section 3.5.

3.2 Shearlets

In this section, we recall the construction of 2D and 3D shearlets (cf. [41,42]).

3.2.1 2D Shearlets

To construct a smooth Parseval frames of shearlets for $L^2(\mathbb{R}^2)$, we start by defining appropriate multiscale function systems supported in the following cone-shaped regions of the Fourier domain $\widehat{\mathbb{R}}^2$:

$$\mathcal{P}_1 = \left\{(\xi_1, \xi_2) \in \mathbb{R}^2 : |\frac{\xi_2}{\xi_1}| \leq 1\right\}, \ \mathcal{P}_2 = \left\{(\xi_1, \xi_2) \in \mathbb{R}^2 : |\frac{\xi_2}{\xi_1}| > 1\right\}.$$

Let $\phi \in C^\infty([0,1])$ be a 'bump' function with $\text{supp}\phi \subset [-\frac{1}{8}, \frac{1}{8}]$ and $\phi = 1$ on $[-\frac{1}{16}, \frac{1}{16}]$. For $\xi = (\xi_1, \xi_2) \in \widehat{\mathbb{R}}^2$, let $\Phi(\xi) = \Phi(\xi_1, \xi_2) = \phi(\xi_1)\,\phi(\xi_2)$ and define the function

$$W(\xi) = W(\xi_1, \xi_2) = \sqrt{\Phi^2(2^{-2}\xi_1, 2^{-2}\xi_2) - \Phi^2(\xi_1, \xi_2)}.$$

Note that the functions $W_j^2 = W^2(2^{-2j}\cdot)$, $j \geq 0$, have support inside the Cartesian coronae

$$C_j = [-2^{2j-1}, 2^{2j-1}]^2 \setminus [-2^{2j-4}, 2^{2j-4}]^2$$

and that they produce a smooth tiling of the frequency plane:

$$\Phi^2(\xi_1, \xi_2) + \sum_{j \geq 0} W^2(2^{-2j}\xi_1, 2^{-2j}\xi_2) = 1 \quad \text{for } (\xi_1, \xi_2) \in \widehat{\mathbb{R}}^2.$$

Let $V \in C^\infty(\mathbb{R})$ so that $\operatorname{supp} V \subset [-1, 1]$, $V(0) = 1$, $V^{(n)}(0) = 0$, for all $n \geq 1$ and

$$|V(u-1)|^2 + |V(u)|^2 + |V(u+1)|^2 = 1 \qquad \text{for } |u| \leq 1.$$

For $F_{(1)}(\xi_1, \xi_2) = V(\frac{\xi_2}{\xi_1})$ and $F_{(2)}(\xi_1, \xi_2) = V(\frac{\xi_1}{\xi_2})$, the *shearlet systems associated with the cone-shaped regions* $\mathcal{P}_\nu$, $\nu = 1, 2$ are defined as the countable collection of functions

$$\{\psi_{j,\ell,k}^{(\nu)} : j \geq 0, -2^j \leq \ell \leq 2^j, k \in \mathbb{Z}^2\}, \tag{3.1}$$

where

$$\hat{\psi}_{j,\ell,k}^{(\nu)}(\xi) = |\det A_{(\nu)}|^{-j/2}\, W(2^{-j}\xi)\, F_{(\nu)}(\xi A_{(\nu)}^{-j} B_{(\nu)}^{-\ell})\, e^{2\pi i \xi A_{(\nu)}^{-j} B_{(\nu)}^{-\ell} k}, \tag{3.2}$$

and

$$A_{(1)} = \begin{pmatrix} 4 & 0 \\ 0 & 2 \end{pmatrix}, \quad B_{(1)} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, \quad A_{(2)} = \begin{pmatrix} 2 & 0 \\ 0 & 4 \end{pmatrix}, \quad B_{(2)} = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}.$$

Note that the dilation matrices $A_{(1)}, A_{(2)}$ produce anisotropic dilations, namely, *parabolic scaling* dilations; by contrast, the *shear matrices* $B_{(1)}, B_{(2)}$ are non-expanding and their integer powers control the directional features of the shearlet system. Hence, the systems (3.1) form collections of well-localized functions defined at various scales, orientations and locations, controlled by the indices j, ℓ, k respectively. In particular, the functions $\hat{\psi}^{(1)}_{j,\ell,k}$, given by (3.2) with $\nu = 1$, can be written explicitly as

$$\hat{\psi}^{(1)}_{j,\ell,k}(\xi) = 2^{-2j}\, W(2^{-2j}\xi)\, V\Big(2^j \frac{\xi_2}{\xi_1} - \ell\Big)\, e^{2\pi i \xi A_{(1)}^{-j} B_{(1)}^{-\ell} k},$$

showing that their supports are contained inside the trapezoidal regions

$$\Sigma_{j,\ell} := \{(\xi_1, \xi_2) : \xi_1 \in [-2^{2j-1}, -2^{2j-4}] \cup [2^{2j-4}, 2^{2j-1}], |\tfrac{\xi_2}{\xi_1} - \ell 2^{-j}| \leq 2^{-j}\}$$

in the Fourier plane (see Fig. 3.1). Similar properties hold for the functions $\hat{\psi}^{(2)}_{j,\ell,k}$.

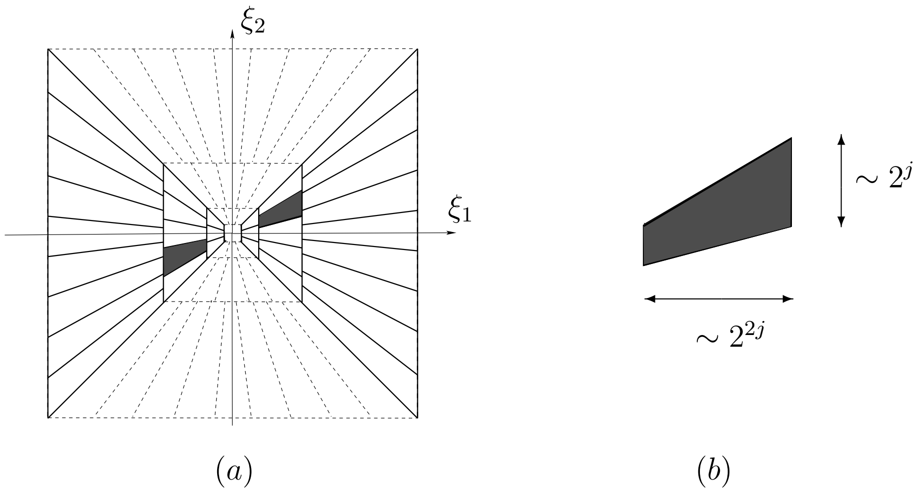


Figure 3.1: (a) The tiling of the frequency plane $\widehat{\mathbb{R}}^2$ induced by the shearlets. (b) Frequency support $\Sigma_{j,\ell}$ of a shearlet $\psi^{(1)}_{j,\ell,k}$, for $\xi_1 > 0$. The other half of the support, for $\xi_1 < 0$, is symmetrical.

A smooth Parseval frame for the whole space $L^2(\mathbb{R}^2)$ is obtained by combining the two shearlet systems associated with the cone-based regions $\mathcal{P}_1$ and $\mathcal{P}_2$ together with a coarse scale system, associated with the low frequency region. To ensure that all elements of this combined shearlet system are C_c^∞ in the Fourier domain, the elements whose supports overlap the boundaries of the cone regions in the frequency domain are slightly modified. That is, we define a *shearlet system for* $L^2(\mathbb{R}^2)$ as

$$\left\{\widetilde{\psi}_{-1,k} : k \in \mathbb{Z}^2\right\} \bigcup \left\{\widetilde{\psi}_{j,\ell,k,\nu} : j \geq 0, |\ell| < 2^j, k \in \mathbb{Z}^2, \nu = 1,2\right\}$$
$$\bigcup \left\{\widetilde{\psi}_{j,\ell,k} : j \geq 0, \ell = \pm 2^j, k \in \mathbb{Z}^2\right\}, \tag{3.3}$$

consisting of:

- the *coarse-scale shearlets* $\{\widetilde{\psi}_{-1,k} = \Phi(\cdot - k) : k \in \mathbb{Z}^2\}$;
- the *interior shearlets* $\{\widetilde{\psi}_{j,\ell,k,\nu} = \psi^{(\nu)}_{j,\ell,k} : j \geq 0, |\ell| < 2^j, k \in \mathbb{Z}^2, \nu = 1,2\}$, where the functions $\psi^{(\nu)}_{j,\ell,k}$ are given by (3.2);
- the *boundary shearlets* $\{\widetilde{\psi}_{j,\ell,k} : j \geq 0, \ell = \pm 2^j, k \in \mathbb{Z}^2\}$, obtained by joining together slightly modified versions of $\psi^{(1)}_{j,\ell,k}$ and $\psi^{(2)}_{j,\ell,k}$, for $\ell = \pm 2^j$, after that they have been restricted in the Fourier domain to the cones $\mathcal{P}_1$ and $\mathcal{P}_2$, respectively. We refer to [41] for details.

For brevity, let us denote the system (3.3) using the compact notation

$$\{\widetilde{\psi}_\mu,\ \mu \in M\},$$

where $M = M_C \cup M_I \cup M_B$ are the indices associated with *coarse scale shearlets*, *interior shearlets*, and *boundary shearlets*, respectively. We have the following result from [41]:

Theorem 3.2.1. *The system of shearlets* (3.3) *is a Parseval frame for* $L^2(\mathbb{R}^2)$. *That is, for any* $f \in L^2(\mathbb{R}^2)$,

$$\sum_{\mu \in M} |\langle f, \widetilde{\psi}_\mu \rangle|^2 = \|f\|^2.$$

All elements $\{\widetilde{\psi}_\mu,\ \mu \in M\}$ *are* C^∞ *and compactly supported in the Fourier domain.*

As mentioned above, it is proved in [42] that the 2D Parseval frame of shearlets $\{\widetilde{\psi}_\mu,\ \mu \in M\}$ provides essentially optimal approximations for functions of 2 variables which are C^2 regular away from discontinuities along C^2 curves.

The mapping from $f \in L^2(\mathbb{R}^2)$ into the elements $\langle f, \widetilde{\psi}_\mu \rangle$, $\mu \in M$, is called the *2D shearlet transform.*

3.2.2 3D Shearlets

The construction outlined above extends to higher dimensions. In 3D, a shearlet system is obtained by appropriately combining 3 systems of functions associated with the pyramidal regions

$$\mathcal{P}_1 = \left\{ (\xi_1, \xi_2, \xi_3) \in \mathbb{R}^3 : |\frac{\xi_2}{\xi_1}| \le 1, |\frac{\xi_3}{\xi_1}| \le 1 \right\},$$
$$\mathcal{P}_2 = \left\{ (\xi_1, \xi_2, \xi_3) \in \mathbb{R}^3 : |\frac{\xi_1}{\xi_2}| < 1, |\frac{\xi_3}{\xi_2}| \le 1 \right\},$$
$$\mathcal{P}_3 = \left\{ (\xi_1, \xi_2, \xi_3) \in \mathbb{R}^3 : |\frac{\xi_1}{\xi_3}| < 1, |\frac{\xi_2}{\xi_3}| < 1 \right\},$$

in which the Fourier space $\widehat{\mathbb{R}}^3$ is partitioned. With ϕ defined as above, for $\xi = (\xi_1, \xi_2, \xi_3) \in \widehat{\mathbb{R}}^3$, we now let

$$\Phi(\xi) = \Phi(\xi_1, \xi_2, \xi_3) = \phi(\xi_1)\,\phi(\xi_2)\,\phi(\xi_3)$$

and $W(\xi) = \sqrt{\Phi^2(2^{-2}\xi) - \Phi^2(\xi)}$. As in the 2-dimensional case, we have the smooth tiling condition

$$\Phi^2(\xi) + \sum_{j \ge 0} W^2(2^{-2j}\xi) = 1 \text{ for } \xi \in \widehat{\mathbb{R}}^3.$$

Hence, for $d = 1, 2, 3$, $\ell = (\ell_1, \ell_2) \in \mathbb{Z}^2$, the 3D *shearlet systems associated with the pyramidal regions* $\mathcal{P}_d$ are defined as the collections

$$\{\psi^{(d)}_{j,\ell,k} : j \geq 0, -2^j \leq \ell_1, \ell_2 \leq 2^j, k \in \mathbb{Z}^3\},$$

where

$$\hat{\psi}^{(d)}_{j,\ell,k}(\xi) = |\det A_{(d)}|^{-j/2}\, W(2^{-2j}\xi)\, F_{(d)}(\xi A^{-j}_{(d)} B^{[-\ell]}_{(d)})\, e^{2\pi i \xi A^{-j}_{(d)} B^{[-\ell]}_{(d)} k},$$

$F_{(1)}(\xi_1, \xi_2, \xi_3) = V(\frac{\xi_2}{\xi_1})V(\frac{\xi_3}{\xi_1})$, $F_{(2)}(\xi_1, \xi_2, \xi_3) = V(\frac{\xi_1}{\xi_2})V(\frac{\xi_3}{\xi_2})$, $F_{(3)}(\xi_1, \xi_2, \xi_3) = V(\frac{\xi_1}{\xi_3})V(\frac{\xi_2}{\xi_3})$,

the anisotropic dilation matrices $A_{(d)}$ are given by

$$A_{(1)} = \begin{pmatrix} 4 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{pmatrix},\ A_{(2)} = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 2 \end{pmatrix},\ A_{(3)} = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 4 \end{pmatrix},$$

and the *shear matrices* are defined by

$$B^{[\ell]}_{(1)} = \begin{pmatrix} 1 & \ell_1 & \ell_2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix},\ B^{[\ell]}_{(2)} = \begin{pmatrix} 1 & 0 & 0 \\ \ell_1 & 1 & \ell_2 \\ 0 & 0 & 1 \end{pmatrix},\ B^{[\ell]}_{(3)} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \ell_1 & \ell_2 & 1 \end{pmatrix}.$$

Similar to the 2D case, the shearlets $\hat{\psi}^{(1)}_{j,\ell,k}(\xi)$ can be written explicitly as

$$\hat{\psi}^{(1)}_{j,\ell_1,\ell_2,k}(\xi) = 2^{-2j}\, W(2^{-2j}\xi)\, V\Big(2^j \frac{\xi_2}{\xi_1} - \ell_1\Big)\, V\Big(2^j \frac{\xi_3}{\xi_1} - \ell_2\Big)\, e^{2\pi i \xi A^{-j}_{(1)} B^{[-\ell_1, -\ell_2]}_{(1)} k}, \quad (3.4)$$

showing that their supports are contained inside the trapezoidal regions

$$\{\xi : \xi_1 \in [-2^{2j-1}, -2^{2j-4}] \cup [2^{2j-4}, 2^{2j-1}], |\frac{\xi_2}{\xi_1} - \ell_1 2^{-j}| \leq 2^{-j}, |\frac{\xi_3}{\xi_1} - \ell_2 2^{-j}| \leq 2^{-j}\}.$$

Note that these support regions become increasingly more elongated at fine scales, due to the action of the anisotropic dilation matrices $A^j_{(1)}$, and the orientations of these regions are controlled by the shear parameters ℓ_1, ℓ_2. A typical support region is illustrated in Fig. 3.2. Similar properties hold for the elements associated with regions $\mathcal{P}_2$ and $\mathcal{P}_3$.

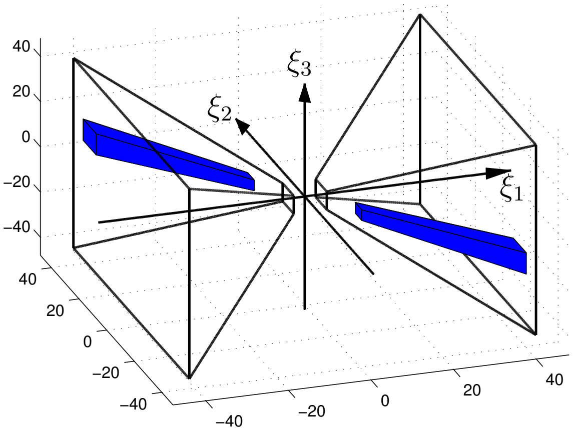


Figure 3.2: Frequency support of a representative shearlet function $\psi_{j,\ell,k}$, inside the pyramidal region $\mathcal{P}_1$. The orientation of the support region is controlled by $\ell = (\ell_1, \ell_2)$; its shape is becoming more elongated as j increases ($j = 4$ in this plot).

A Parseval frame of shearlets for $L^2(\mathbb{R}^3)$ is obtained by using an appropriate combination of the systems of shearlets associated with the 3 pyramidal regions $\mathcal{P}_d$, $d = 1, 2, 3$, together with a coarse scale system, which will take care of the low frequency region. Similar to the 2D case, in order to build such system in a way that all its elements are smooth in the Fourier domain, one has to appropriately define the elements of the shearlet systems overlapping the boundaries of the pyramidal regions $\mathcal{P}_d$ in the Fourier domain. We refer to [43, 50] for details. Hence, we define the *3D shearlet systems for* $L^2(\mathbb{R}^3)$ as the collections

$$\left\{\widetilde{\psi}_{-1,k} : k \in \mathbb{Z}^3\right\} \bigcup \left\{\widetilde{\psi}_{j,\ell,k,d} : j \geq 0, |\ell_1| < 2^j, |\ell_2| \leq 2^j, k \in \mathbb{Z}^3, d = 1, 2, 3\right\}$$
$$\bigcup \left\{\widetilde{\psi}_{j,\ell,k} : j \geq 0, \ell_1, \ell_2 = \pm 2^j, k \in \mathbb{Z}^3\right\},$$

which again can be identified as the coarse-scale, interior and boundary shearlets. It turns out that the 3D system of shearlets is a Parseval frame of $L^2(\mathbb{R}^3)$ [41] and it provides essentially optimal approximations for functions of 3 variables which are C^2 regular away from discontinuities along C^2 surfaces [43].

3.3 Discrete Implementation

A faithful numerical implementation of the 2D shearlet transform was originally presented in [44]. Let us briefly recall the main steps of this implementation.

3.3.1 2D Discrete Shearlet Transform

Recall that the shearlet coefficients associated with the interior shearlets can be expressed as:

$$\langle f, \psi^{\nu}_{j,\ell,k}\rangle = 2^{3j/2} \int_{\widehat{\mathbb{R}}^2} \hat{f}(\xi)\, W(2^{-2j}\xi)\, F_{(\nu)}(\xi A^{-j}_{(\nu)} B^{-\ell_{(\nu)}})\, e^{2\pi i \xi A^{-j}_{(\nu)} B^{-\ell_{(\nu)}} k}\, d\xi.$$

First, to compute $\hat{f}(\xi_1, \xi_2)\, W(2^{-2j}\xi)$ in the discrete domain, at the resolution level j, we apply the Laplacian pyramid algorithm [58], which is implemented in space-domain. Let $\hat{f}[k_1, k_2]$ denote 2D Discrete Fourier Transform (DFT) of $f \in \ell^2(\mathbb{Z}^2_N)$, where we adopt the convention that brackets $[\cdot, \cdot]$ denote arrays of indices, parentheses $(\cdot, \cdot)$ denote function evaluations, and where we interpret the numbers $\hat{f}[k_1, k_2]$ as samples $\hat{f}[k_1, k_2] = \hat{f}(k_1, k_2)$

from the trigonometric polynomial

$$\hat{f}(\xi_1,\xi_2) = \sum_{n_1,n_2=0}^{N-1} f[n_1,n_2]\, e^{-2\pi i(\frac{n_1}{N}\xi_1+\frac{n_1}{N}\xi_2)}.$$

The Laplacian pyramid algorithm will accomplish the multiscale partition illustrated in Figure 3.3, by decomposing $f_a^{j-1}[n_1,n_2]$, $0 \le n_1,n_2 < N_{j-1}$, into a low pass filtered image $f_a^j[n_1,n_2]$, a quarter of the size of $f_a^{j-1}[n_1,n_2]$, and a high pass filtered image $f_d^j[n_1,n_2]$. Observe that the matrix $f_a^j[n_1,n_2]$ has size $N_j \times N_j$, where $N_j = 2^{-2j}N$, and $f_a^0[n_1,n_2] = f[n_1,n_2]$ has size $N \times N$. In particular, we have

$$\widehat{f_d^j}(\xi_1,\xi_2) = \hat{f}(\xi_1,\xi_2)\, W(2^{-2j}(\xi_1,\xi_2))$$

and thus, $f_d^j[n_1,n_2]$ are the discrete samples of a function $f_d^j(x_1,x_2)$, whose Fourier transform is $\widehat{f_d^j}(\xi_1,\xi_2)$. Since this operation is implemented as a convolution in space-domain, this step of the algorithm is one of the most computationally expensive.

The next step produces the directional filtering and this is achieved by computing the DFT on the pseudo-polar grid, and then applying a one-dimensional band-pass filter to the components of the signal with respect to this grid. More precisely, let us define the pseudo-polar coordinates $(u,v) \in \mathbb{R}^2$ as follows:

$$\begin{aligned} (u,w) = &\ \ (\xi_1, \tfrac{\xi_2}{\xi_1}) \quad \text{if } (\xi_1,\xi_2) \in \mathcal{P}_1, \\ (u,w) = &\ \ (\xi_2, \tfrac{\xi_1}{\xi_2}) \quad \text{if } (\xi_1,\xi_2) \in \mathcal{P}_2. \end{aligned}$$

After performing this change of coordinates, we obtain

$$\hat{f}(\xi_1,\xi_2)\, W(2^{-2j}\xi_1, 2^{-2j}\xi_2)\, F_{(\nu)}(\xi A_{(\nu)}^{-j} B^{-\ell_{(\nu)}}) = g_j(u,w)\, V(2^j w - \ell), \tag{3.5}$$

where $g_j(u,w) = \hat{f}_d^j(\xi_1,\xi_2)$. This shows that the directional components are obtained by simply translating the window function V. The discrete samples $g_j[n_1, n_2] = g_j(n_1, n_2)$ are the values of the DFT of $f_d^j[n_1, n_2]$ on a pseudo-polar grid.

Now let $\{v_{j,\ell}[n] : n \in \mathbb{Z}\}$ be the sequence whose discrete Fourier transform gives the samples of the window function $V(2^j k - \ell)$, i.e., $\hat{v}_{j,\ell}[k] = V(2^j k - \ell)$. Then, for fixed $n_1 \in \mathbb{Z}$, we have

$$\mathcal{F}_1\left(\mathcal{F}_1^{-1}\left(g_j[n_1, n_2]\right) * v_{j\ell}[n_2]\right) = g_j[n_1, n_2]\,\mathcal{F}_1\left(v_{j\ell}[n_2]\right), \tag{3.6}$$

where $*$ denotes the one-dimensional convolution along the n_2 axis and $\mathcal{F}_1$ is the one-dimensional discrete Fourier transform Thus (3.6) gives the algorithmic implementation for computing the discrete samples of $g_j(u, w)\, v(2^j w - \ell)$. At this point, to compute the shearlet coefficient in the discrete domain, it suffices to compute the inverse PDFT or directly re-assemble the Cartesian sampled values and apply the inverse two-dimensional FFT.

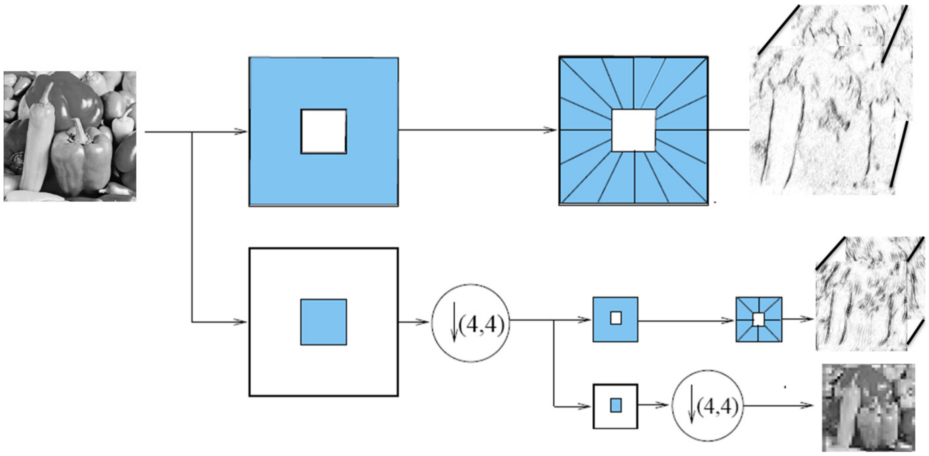


Figure 3.3: The figure illustrating the succession of Laplacian pyramid and directional filtering.

Figure 3.3 illustrates the cascade of Laplacian pyramid and directional filtering.

Recall that, once the discrete shearlet coefficients are obtained, the inverse shearlet transform is computed using the following steps: (i) convolution of discrete shearlet coefficients and synthesis directional filters; (ii) sum of all directional components; (iii) reconstruction by inverse Laplacian pyramidal transformation.

3.3.2 2D GPU-based Implementation

Before implementing the 2D Discrete Shearket Transform algorithm on the GPU, we profiled the existing implementation available as a MATLAB toolbox at `http://www.math.uh.edu/~dlabate/shearlet_toolbox.zip`. Table 3.3 contains the breakdown of the processing times showing that the FFT computations used to perform directional filtering and the convolution part of the *à trous* algorithm used for pyramidal image decomposition and reconstruction take around 75% of the computation time. Hence they were the first candidates for porting into CUDA.

Since most of the computing time for performing a discrete shearlet transform is spent in FFT function calls, it is crucial to have the best possible library to perform FFTs. The main two GPU vendors provide optimized FFT libraries: NVIDIA provides *cuFFT* as part of its CUDA Toolkit, and AMD provides *clAmdFft* as part of its Accelerated Parallel Processing Math Libraries (APPML). We decided to use CUDA as our development architecture both because there is better documentation and because of the availability of more mature development tools. We implemented the device code in CUDA C++, while the host code is pure C++. Since both CUDA C/C++ and OpenCL are based on the C programming language, porting the code from CUDA to OpenCL should not be difficult. However, for code compactness, we made extensive use of templates and operator overloading, which are supported in CUDA C++, but not in OpenCL, which is based on C99.

To facilitate the development, we used GPUmat from the GP-you Group, a free (GPLv3) GPU engine for MATLAB® (source code is available from `http://sourceforge.net/projects/gpumat/`). This framework provides two new classes, *GPUsingle* and *GPUdouble*, which encapsulate vectors of numerical data allocated on GPU memory, and allow mathematical operations on objects of such classes via function and operator overloading. Transfers between CPU and GPU memory are as simple as doing typecasting, and memory allocation and deallocation is done automatically. The idea is that existing MATLAB functions could be reused without any code changes. In practice, however, in order to get acceptable performance it is necessary to hand-tune the code or even use lower level languages such as C/C++.

Fortunately, the GPUmat framework provides an interface for manipulating these objects from MEX files, and a mechanism for loading custom kernels. Although there are commercial alternatives to GPUmat such as Jacket from AccelerEyes, or the Parallel Computing Toolbox from Mathworks, we found that GPUmat is pretty robust and adds very little overhead to the execution time as long as we follow good programming practices such as in-place operations and reuse of preallocated buffers.

Our implementation supports both single precision (32-bit) and double precision (64-bit) IEEE 754 floating point numbers. We generated the filter bank of directional filters using the Fourier-domain approach from [44], where directional filters are designed as Meyer-type window functions in the Fourier domain. Since this step only needs to be run once and does not depend on the image dimensions, we precompute these directional filters using the original MATLAB implementation.

For the Laplacian pyramidal decomposition, we ported the *à trous* algorithm using symmetric extension [37] into CUDA. This algorithm requires performing non-separable convolutions with decimated signals. For efficiency reasons, the kernel that performs *à trous* convolutions preloads blocks of data into shared memory, so that the memory is only accessed once from each GPU thread.

With the above GPU-based Laplacian pyramid and directional filter implementation, it is just a matter of applying convolutions in the GPU to find the forward and inverse shearlet transform.

The main steps of our GPU-based shearlet transform are shown in table 3.1.

Table 3.1: Main steps of the shearlet transform

Forward transform	**Inverse transform**
1. *Laplacian decomposition*	1. *Forward FFT of directional components*
2. *Forward FFT of Laplacian components*	2. *Modulation with complex conjugate directional filter bank*
3. *Modulation of Laplacian components with directional filter bank*	3. *Inverse FFT of directional components*
4. *Inverse FFT of directional components*	4. *Laplacian reconstruction*

3.3.3 3D discrete shearlet transform

The algorithm for the discretization of the 3D shearlet transform is very similar to the 2D shearlet transform and our implementation of the 3D discrete shearlet transform adapts the code available from `http://www.math.uh.edu/~dlabate/3Dshearlet_toolbox.zip` and described in [55]. The main practical difference is that storing the 3D shearlet coefficients is much more memory-intensive. Since the memory requirement can be easily exceed the available GPU memory, in our algorithm we compute one convolution at a time in CUDA and add the result to the output.

3.4 Applications

In the following, we illustrate the advantages of our new implementation of the discrete shearlet transform by considering three applications: denoising of natural images corrupted with white Gaussian noise, detection of cracks in textured images and denoising of videos. The source code, sample data as well as the MATLAB scripts used to generate all the figures in this chapter are publicly available at `http://www.umiacs.umd.edu/~gibert/ShearCuda.zip`.

For benchmark, we have evaluated the performance of the new discrete shearlet transform both on multicore CPUs and GPU. All CPU tests have been performed on a Dell PowerEdge C6145 with four-socket AMD Opteron$^{\text{TM}}$6274 processors at 2.2GHz (64 cores total) and 256GB RAM, running Red Hat Enterprise Linux (REHL) 6. This machine is one of 16 identical nodes in the High Performance Computing (HPC) cluster Euclid at the University of Maryland. During these benchmarks, we had exclusive access to this node, and no other processes were running, except for regular system services. To better understand the performance of this code when running on systems with different number of cores, we limited the number of available cores on some of the experiments. We found that neither MATLAB's *maxNumCompThreads* nor *–singleCompThread* work reliably on non-Intel processors, so we used the *taskset* Linux command to set the processor affinity to the desired number of cores. GPU tests were performed on different machines running RHEL 5 or 6, and CUDA 4.2 or 5.0. The tests include devices with CUDA Compute Capabilities (CC) between 1.3 and 3.5. Table 3.2 summarizes the configurations used in our experiments.

Table 3.2: Specifications and computing environments for each of the graphics processors used on our benchmarks

GPU Model	Memory	#Cores	CC	OS	CUDA
Tesla C1060	4GB	240	1.3	RHEL 5	5.0.35
GeForce GTX 480	1.5GB	448	2.0	RHEL 6	4.2.9
Tesla C2050	3GB	448	2.0	RHEL 6	4.2.9
GeForce GTX 690[1]	2GB	1536	3.0	RHEL 6	5.0.35
Tesla K20c	4.8GB	2496	3.5	RHEL 6	5.0.35

[1] Although the GeForce GTX 690 is a dual-GPU with a total of 4GB and 3072 cores, we have only used one of the 2 devices in the GPU for our experiments.

3.4.1 Image denoising

As a first test, we evaluated the performance of our implementation of the discrete shearlet transform on a problem of image denoising, using a standard denoising algorithm based on hard threshold of the shearlet coefficients. The setup is similar to the one described in [44]. That is, given an image $f \in \mathbb{R}^{N^2}$, we observe a noisy version of it given by $u = f + \epsilon$, where $\epsilon \in \mathbb{R}^{N^2}$ is an additive white Gaussian noise process which is independent of f, i.e., $\epsilon \sim N(0, \sigma^2 \mathbf{I}_{N^2 \times N^2})$. Our goal is to compute an estimate $\tilde{f}$ of f from the noisy data u by applying a classical hard thresholding scheme [59] on the shearlet coefficients of u. The threshold levels are given by $\tau_{i,j,n} = \sigma^2_{\epsilon_{i,j}} / \sigma^2_{i,j,n}$, as in [37, 44, 60], where $\sigma^2_{i,j,n}$ denotes the variance of the n-th coefficient at the ith directional subband in the jth scale, and $\sigma^2_{\epsilon_{i,j}}$ is the noise variance at scale j and directional band i. The variances $\sigma^2_{\epsilon_{i,j}}$ are estimated by using a Monte-Carlo technique in which the variances are computed for several normalized noise images and then the estimates are averaged.

For our experiments, we used 5 levels of the Laplacian pyramid decomposition, and we applied a directional decomposition on 4 of the 5 scales. We used 8 shear filters of sizes 32×32 for the first two scales (coarser scales), and 16 shear filters of sizes 16×16 for the third and fourth levels (fine scales). The shear filters are Meyer-type windows [44]. We used the 512×512 Barbara image to test our algorithm and, to assess its performance, we used the peak signal-to-noise ratio (PSNR), measured in decibels (dB), defined by

$$PSNR = 20 \log_{10} \frac{255N}{\|f - \tilde{f}\|_F},$$

where $\|\cdot\|_F$ is the Frobenius norm, the given image f is of size $N \times N$ and $\tilde{f}$ denotes the estimated image.

In order to minimize latency as well as bandwidth usage on the PCIe bus, we first transferred the input image to GPU memory, then we let all the computation happen on the GPU and we finally transferred the results back to CPU memory. We have verified that both CPU and GPU implementations provide an output PSNR of 29.9dB when the input PSNR is 22.1dB. At these noise levels, there is no difference in PSNR between single and double precision implementations.

To verify the numerical accuracy, we ran the shearlet decomposition and reconstruction on a noise free image (without thresholding), and we obtained a reconstruction MSE (Mean Squared Error) of 9.197×10^{-09} for single precision and 2.503×10^{-12} for double precision on a GeForce GTX 690. On the CPU implementation, we get reconstruction errors of 9.1711×10^{-09} and 1.6643×10^{-26}, respectively. This verifies that our implementation does provide the exact reconstruction.

The running times vary significantly depending on the number of CPU cores available and the GPU model. Figure 3.4 shows a comparison of running times (wall times) of the image denoising algorithm on different hardware configurations. We can clearly see that the CPU implementation does not scale well as we increase the number of CPU cores due to parts of the algorithm running sequentially. For a fair comparison of multicore vs GPU, we would have to compare the performance to a fully optimized CPU implementation. It should be noted that there is enough coarse level parallelism on this algorithm to accomplish full CPU utilization without incurring any inter CPU communication issues. However, the trend reveals that for this application, GPU is more efficient than CPU. In summary, the denoising algorithm takes 8.89 seconds on 4 CPU cores vs. 0.038 seconds on the GeForce GTX 690 (a 233× speed-up) when using single precision. For double precision, it takes 10.7 seconds on 4 CPU cores vs. 0.127 seconds on the GeForce GTX 690 (an 84× speed-up).

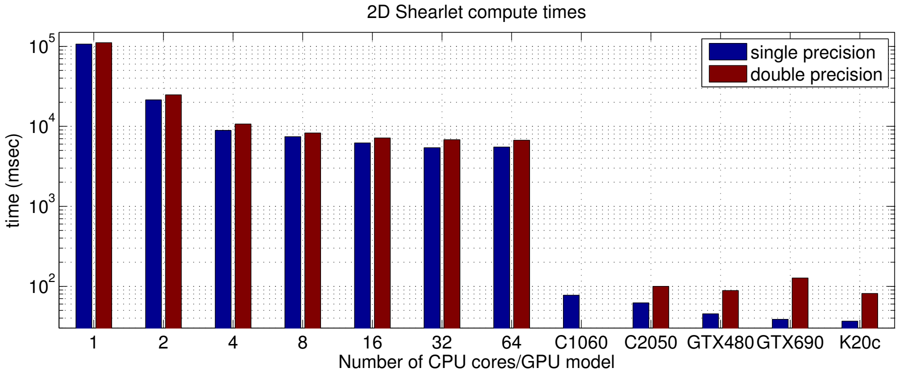


Figure 3.4: Comparison of CPU vs GPU run times for denoising a 512×512 image using shearlets.

Table 3.3 shows the breakdown of different parts of the image denoising algorithm both on CPU and GPU.

Table 3.3: Comparison of processing times for denoising a single precision 512×512 image on a multicore CPU using 4 CPU cores vs. a GeForce GTX 690 GPU.

Step	4-core CPU		GTX 690 GPU	
	time (s)	% time	time (ms)	% time
Laplacian pyramid	2.787	31.6%	18.282	47.3%
Directional filters	4.386	49.7%	18.350	47.5%
Hard threshold	0.375	4.2%	1.967	5.1%
Other	1.281	14.5 %	0.063	0.2%
TOTAL TIME	8.829 seconds		38.662 msec	

3.4.2 Crack detection

Detection of cracks on concrete structures is a difficult problem due to the changes in width and direction of the cracks, as well as the variability in the surface texture. This problem has received considerable attention recently. Redundant representations, such as undecimated wavelets, have been extensively used for crack detection [61, 62]. However, wavelets have poor directional sensitivity and have difficulties in detecting weak diagonal cracks. To overcome this limitation, Ma *et al.* [63] proposed the use of the *nonsubsampled contourlet transform* [37] for crack detection. However, all these methods rely on the assumption that the background surface can be modeled as additive white Gaussian noise and his assumption leads to matched filter solutions. As a matter of fact, on real images textures are highly correlated and applying linear filters leads to poor performance.

To address this problem, we propose a completely new approach to crack detection based on separating the image into morphologically distinct components using sparse representations, adaptive thresholding and variational regularization. This technique was pioneered by Stark *et al.* [64] and later extended and generalized by many authors (e.g., [52, 53, 65]). In particular, we will use the Iterative Shrinkage Algorithm with a combined dictionary of shearlets and wavelets to separate cracks from background texture.

To demonstrate the performance of the GPU-accelerated Iterative Shrinkage Algorithm, we processed three 512×512 images. The images correspond to cracks on concrete railroad crossties collected by ENSCO Inc. during summer 2012 using four 2048×1 line-scan cameras, which were assembled into 8192×3072 frames. The cameras were triggered using a calibrated encoder, producing images with square pixels with a constant size of 0.43mm. We manually cropped these images so that we can decouple crack detection from crosstie boundary tracking. As one can see from Figure 3.6, these cracks propagate in different directions and the background texture has a lot of variation. However, due to the fact that the information in these images is highly redundant, it is possible to separate the image into two components, that is, cracks and texture, by solving an ℓ_1 optimization problem [52].

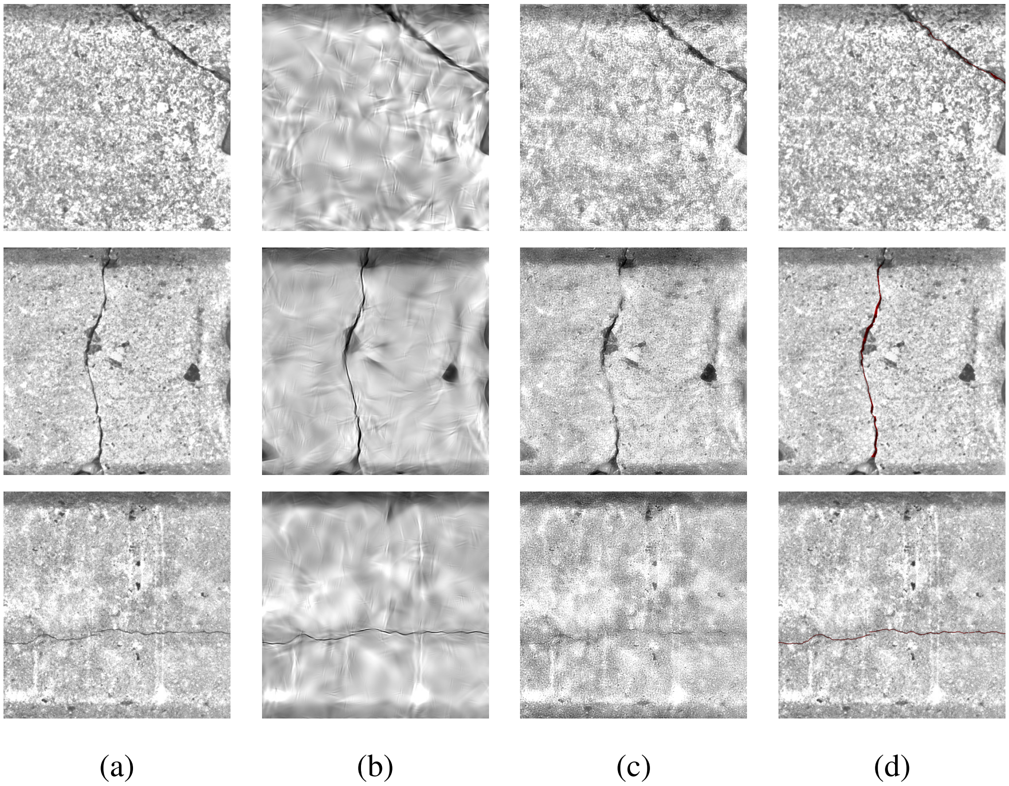


Figure 3.5: **Image separation.** (a) Original images separated into (b) Cracks and (c) Textural background components (d) Crack ground truth

More precisely, we model an image x containing cracks on textural background as a superposition of a crack component x_c with a textural component x_t:

$$x = x_c + x_t.$$

Let Φ_1 and Φ_2 be the dictionaries corresponding to wavelets and shearlets, respectively.

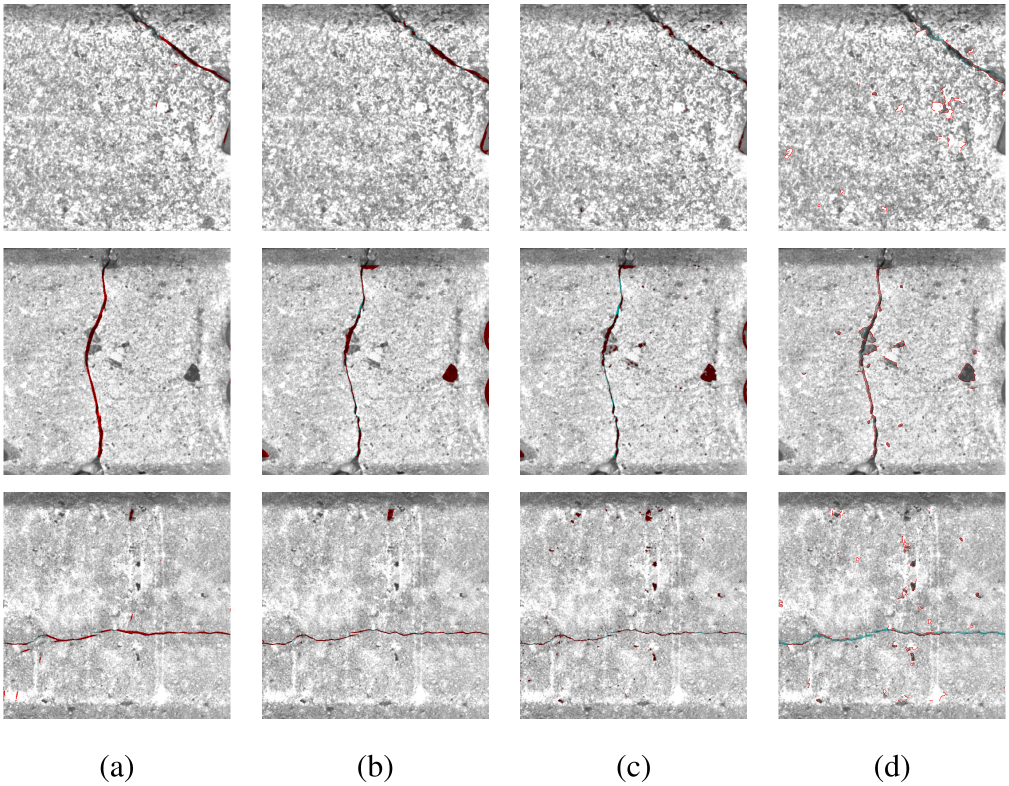


Figure 3.6: **Crack detection results.** (e) using shearlet coefficients (Shearlet-C) (f) using thresholding in the image reconstruction using shearlets (Shearlet-I) (g) using intensity thresholding in the original image (h) using Canny edge detection. All results are generated at peak F_2 score

We assume that x_c is sparse in a shearlet dictionary Φ_1 and similarly x_t is sparse in a wavelet dictionary Φ_2. That is, we assume that there are sparse coefficients a_c and a_t so that $x_c = \Phi_1 a_c$ and $x_t = \Phi_2 a_t$. Then, one can separate these components from an x via the coefficients a_c and a_t by solving the following optimization problem:

$$(\hat{a}_c, \hat{a}_t) = \arg\min{}_{a_c, a_t} \lambda \|a_c\|_1 + \lambda \|a_t\|_1 + \frac{1}{2} \|x - \Phi_1 a_c - \Phi_2 a_t\|_2^2, \tag{3.7}$$

where for an n-dimensional vector b the ℓ_1 norm is defined as $\|b\|_1 = \sum_i |b_i|$. This image separation problem can be solved efficiently using an *iterative shrinkage algorithm* proposed in [52].

In our numerical experiments, we used symlet wavelets with 4 decomposition levels to generate Φ_2 and a 4-level shearlet decomposition with Meyer filters of sizes 80×80 on all 4 scales, 8 directional filters on the first three scales, and 16 directional filters on the forth scale, to generate Φ_1. To assess the performance of the separation algorithm, we calculated the ROC curves for each image using the following 2 detection methods.

a) *Shearlet-C*: This method takes advantage of the Parseval property of the shearlet transform and performs crack detection directly in the transform domain. We first decompose the image into cracks and texture components using Iterative Shrinkage with a shearlet dictionary and a wavelet one. Instead of using the reconstructed image, we analyze the values of the shearlet transform coefficients. For each scale in the shearlet transform domain, we analyze the directional components corresponding to each displacement and collect the maximum magnitude across all directions. If the sign of the shearlet coefficient corresponding to the maximum magnitude is positive, we classify the corresponding pixel as background, otherwise we assign the norm of the vector containing the maximum responses at each scale to each pixel and apply a threshold.

b) *Shearlet-I*: We first decompose the image into cracks and texture components as described for the previous method. Then, we apply an intensity threshold on the reconstructed cracks image.

We compare our results to the following 2 basic methods not based on shearlets:

c) *Intensity*: This is the most basic approach, which only uses image intensity. After compensating for slow variations of intensity in the image, we apply a global threshold.

d) *Canny*: We use the Canny [66] edge detector as implemented in MATLAB using the default $\sigma = \sqrt{2}$ and the default high to low threshold ratio of 40%.

After using a low-level detector, it may be necessary to remove small isolated regions corresponding to false detections due to random noise. This postprocessing step may reduce the false detection rate on intensity-based methods. However, to provide an objective comparison, we have generated the experimental results without running any postprocessing. We leave the performance analysis of a complete crack detector for future work.

To evaluate the performance of each crack detector, we manually annotated the crack pixels in each image. To mitigate the effect of ambiguous segmentation boundaries, we annotated the boundaries around the cracks as tightly as possible (making sure that only pixels completely contained inside the crack boundaries are annotated as such) and defined an envelope region around each crack whose labels are treated as "do not care". Formally, let Ω denote the set of pixels in the image, and F (foreground) denote the set of pixels labeled as cracks. We define the set B (background) as

$$B = \{x \in \Omega : \min_{f \in F} \|x - f\| > \delta\}.$$

where $\|x - f\|$ denotes the Euclidean distance between sites x and f. In our experiments we used $\delta = 3$. To account for possible small inaccuracies in the ground truth, we performed a bipartite graph matching between the detected crack pixels and the crack pixels in the ground truth. For our experiments, we allow matching within a maximum distance of 2 pixels. This choice of matching metric does not penalize crack overestimation errors as long as these errors are contained in such envelope. This allows us to decouple errors in estimating the position of the crack centerline from errors in estimating the crack width, which is more sensitive to lighting variations. Let D be the set of pixels detected as cracks by a given detector and

$$tp = |D \cap F| \qquad fn = |\bar{D} \cap F| \qquad p = tp + fn = |F|$$

$$tn = |\bar{D} \cap B| \qquad fp = |D \cap B| \qquad n = tn + fp = |B|$$

The probability of detection (PD) and false alarm (PF) are defined as

$$PD = \frac{tp}{p} \qquad PF = \frac{fp}{n}$$

A sequence of admissible detectors $D|_{PF \leq \epsilon}$, for a given false detection rate ϵ, $0 \leq \epsilon \leq 1$ would produce monotonically increasing detection rates, $PD|_{PF \leq \epsilon}$. The Receiving Operating Characteristic function (ROC curve) is defined as PD as a function of PF

$$ROC(x) = \max_{\epsilon \leq x} PD|_{PF = \epsilon}$$

One commonly used metric is the Area Under the ROC Curve (AUC), defined by

$$AUC = \int_0^1 ROC(x)\, dx,$$

which corresponds to the probability that a sample randomly drawn from F will receive a score higher than a sample randomly drawn from B. AUC provides a measure of the average performance of the detection across all possible sensitivity settings. Although it is an important measure, in practice we are interested in knowing how well the detector will work when we choose a particular sensitivity setting. For this reason, we have selected Constant False Alarm Rate (CFAR) detectors with $PF = 10^{-3}$ and $PF = 10^{-4}$ and we report the corresponding PD. For completeness, we also report the F_1 score (also know as the Dice similarity index), which is defined as

$$F_1 = \frac{2\,tp}{2\,tp + fn + fp}$$

The F_1 score is also known as the balanced $F-$score, since it is equivalent to the harmonic mean of the *precision* and *recall*:

$$F_1 = 2\,\frac{precision \cdot recall}{precision + recall}$$

where

$$precision = \frac{tp}{p} \qquad recall = \frac{tp}{tp + fn}.$$

In this chapter, we report the peak F_1 score for all methods. The Canny edge detection method estimates the location of the crack boundary, while the other three methods estimate the location of the crack itself. To have a meaningful comparison, we have generated a separate ground truth masks for the crack outline, so we can use the same matching metric on the Canny method. For each method, we have used the same algorithm parameters on all the images.

Table 3.4 summarizes our results. We observe that our shearlet-based detectors perform consistently well on all evaluation metrics. Note that, on Image 3, the Shearlet-I method, which is based on intensity in the reconstructed image, produces better results than all other methods. Due to its simplicity, the intensity-based methods is still being used by the industry. For example, the system recently proposed in [67] uses pixel intensities to detect the cracks on the road pavement. Based on the results from Table 3.4, we can conclude that, with the proper image preprocessing, intensity can still be a powerful feature for crack detection. However, the detection performance provided by shearlet-based features is more consistent across images. In future work, we will further explore the potential of combining both intensity and shearlet-based features.

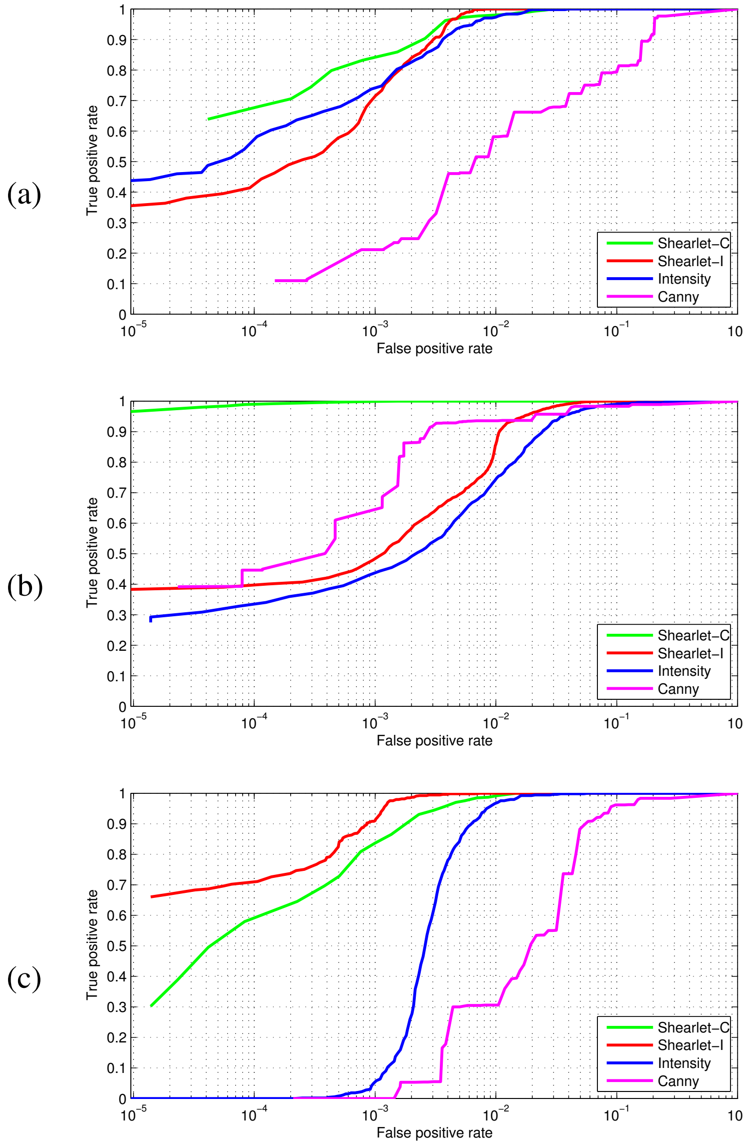


Figure 3.7: **ROC curves for crack detection.** (a) Image 1 (b) Image 2 (c) Image 3

Table 3.4: Comparison of detection performance for different crack detection algorithms.

Image	Method	AUC	F_1 score	$PD\|_{PF=10^{-3}}$	$PD\|_{PF=10^{-4}}$
1	Shearlet-C	**0.99915**	**0.79916**	**0.8398**	**0.6746**
	Shearlet-I	0.99908	0.65810	0.7140	0.4247
	Intensity	0.99874	0.73188	0.7411	0.5722
	Canny	0.94457	0.27752	0.2114	0.1099
2	Shearlet-C	**0.99999**	**0.98841**	**0.9989**	**0.9895**
	Shearlet-I	0.99557	0.62705	0.4837	0.3964
	Intensity	0.99037	0.55404	0.4371	0.3342
	Canny	0.99043	0.81787	0.6425	0.4462
3	Shearlet-C	0.99934	0.76418	0.8368	0.5874
	Shearlet-I	**0.99977**	**0.82353**	**0.9101**	**0.7098**
	Intensity	0.99650	0.45992	0.0543	0.0000
	Canny	0.96248	0.19436	0.0000	0.0000

3.4.3 Video denoising

Video denoising can be performed using the same type of algorithm described above for image denoising and consisting, essentially, in computing the shearlet coefficients of the noisy data, followed by hard thresholding and reconstruction from the thresholded coefficients. Similar to the previous section, a noisy video is obtained by adding white Gaussian noise to a video sequence.

We have tested our GPU-based implementation of the 3D shearlet video denoising algorithm using the $192 \times 192 \times 192$ waterfall video sequence. Figure 3.8 shows frame 96 before and after denoising. Figure 3.9 compares the running times of the video denoising algorithm based on CPU vs. GPU. One can notice that, when we go from single core to dual core, the run time drops from 27.5 minutes to 14.4 minutes on single precision (a 1.91× speed-up). However, when going from dual-core to quad core we only get 1.62× speed-up, and the rate of improvement as we keep doubling the number of cores keeps diminishing, to the point where the improvement from single core to 64 cores is just a 9.45× speed-up. On the other hand, a GeForce 480 produces the same result in just 3 seconds, a remarkable 551× speed-up compared to single core CPU, and 58× speed-up over 64 CPU cores.

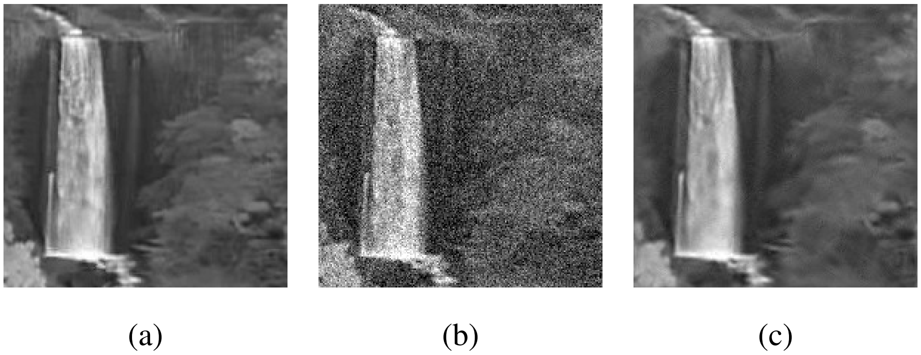


Figure 3.8: **Video denoising.** (a) Original video frame (b) Noise added (c) Denoised frame

3.5 Discussion and Conclusion

The shearlet transform is an advanced multiscale method which has emerged in recent years as a refinement of the traditional wavelet transform and was shown to perform very competitively over a wide range of image and data processing problems. However,

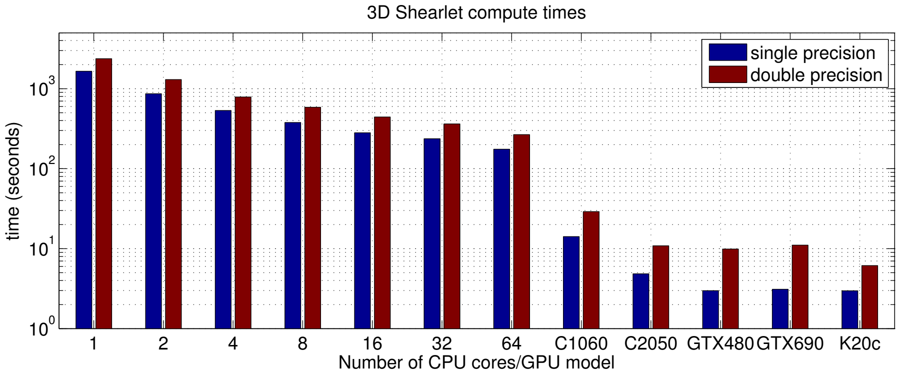


Figure 3.9: Comparison of CPU vs GPU run times for denoising a 192^3 video using 3D shearlets. Time includes all transfers between CPU and GPU.

standard CPU-based numerical implementations are very time-consuming and make the application of this method to large data sets and real-time problems very impractical.

In this chapter, we described how to speed-up the computation of the 2D/3D discrete shearlet transform by using GPU-based implementations. The development of algorithms on GPU used to be tedious and require a very specialized knowledge of the hardware. Using CUDA this is no longer the case and scientists with C/C++ programming skills can quickly develop efficient GPU implementations of data-intensive algorithms. In this chapter, we have taken advantage of the GPU-based implementation of the Fast Fourier Transform and used the capabilities of MATLAB for quick prototyping. The results presented in this chapter illustrate the practical benefits of this approach. For example, a GeForce 480 GTX, a $200 graphics card, can perform video denoising 58 times faster than an expensive 64-core machine while consuming much less power.

Our new implementation enables the efficient application of the shearlet decomposition to a variety of image and data processing tasks for which the required CPU resources would be prohibitive. There are further improvements and extensions that can be achieved such as pre-calculating the filter coefficients and porting the code to OpenCL so it can also run on AMD and Intel GPUs, but this would go beyond the scope of this chapter.

Chapter 4: Image Dictionaries for Anomaly Detection

4.1 Introduction

Monitoring the condition of railway fasteners is essential to ensure train safety. As we explained in section 2.2 in Ch 2, fasteners maintain gage by keeping both rails firmly attached to the crossties. Fasteners need to be inspected periodically and this inspections are currently performed manually by railroad personnel. However, such inspections are subjective and do not produce an auditable visual record. In addition, railroads usually perform automated track inspections with specialized track geometry measurement vehicles at intervals of 30 days or less between inspections. These automated inspections can directly detect gage widening conditions. However, it is preferable to find fastening problems before they develop into gage widening conditions. This chapter shows that, by applying computer vision techniques, it is possible to inspect tracks for missing and broken components using only grayscale images with no additional sensors. Figure 4.1 shows the types of defects that our algorithm can detect. The detectors have been tested on concrete ties, but the framework can easily accommodate other types of fasteners and ties.

This chapter is organized as follows. In Section 4.2, we review some related works on this topic. Details of our approach are given in Section 4.3. Experimental results on

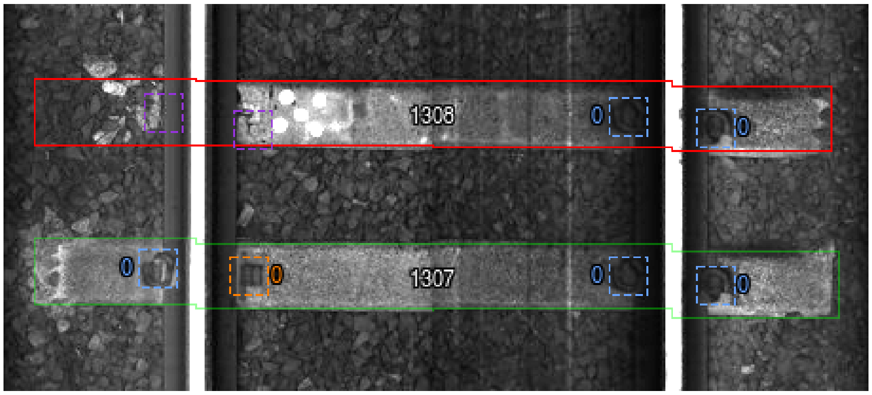


Figure 4.1: **Example of defects that our algorithm can detect.** Blue boxes indicate good fastener, orange boxes indicate broken fasteners, and purple boxes indicate missing fasteners. White numbers indicate tie index from last mile post. Other numbers indicate type of fastener (for example, 0 is for e-clip fastener).

85 miles of concrete tie images are presented in Section 4.4. Section 4.7 concludes the chapter with a brief summary and discussion.

4.2 Prior Work

Since the pioneering work by Cunningham et al. [16,17] in the mid 1990's, machine vision has been gradually adopted by the railway industry as a track inspection technology. Those first generation systems were capable of collecting images of the railway right of way and storing them for later review, but they did not facilitate any automated detection. As faster processing hardware became available, several vendors began to introduce vision systems with increasing automation capabilities.

In [19, 29], Marino et al., describe their VISyR system, which detects hexagonal-headed bolts using two 3-layer neural networks (NN) running in parallel. Both networks take the 2-level discrete wavelet transform (DWT) of a 24×100 pixel sliding window (their images use non-square pixels) as an input to generate a binary output indicating the presence of a fastener. The difference is that the first NN uses Daubechies wavelets, while the second one uses Haar wavelets. This wavelet decomposition is equivalent to performing edge detection at different scales with two different filters. Both neural networks are trained with the same examples. The final decision is made using the maximum output of each neural network. In [20, 21], Gibert et al., describe their VisiRail system for joint bar inspection. The system is capable of collecting images on each rail side, and finding cracks on joint bars using edge detection and a Support Vector Machine (SVM) classifier that analyzes features extracted from these edges. In [22], Babenko describes a fastener detection method based on a convolutional filter bank that is applied directly to intensity images. Each type of fastener has a single filter associated with it, whose coefficients are calculated using an illumination-normalized version of the Optimal Tradeoff Maximum Average Correlation Height (OT-MACH) filter [23]. This approach allows accurate fastener detection and localization and it achieved over 90% fastener detection rate on a dataset of 2,436 images. However, the detector was not tested on longer sections of track. In [24], Resendiz et al.use texture classification via a bank of Gabor filters followed by an SVM to determine the location of rail components such as crossties and turnouts. They also use the MUSIC algorithm to find spectral signatures to determine expected component locations. In [25], Li et al.describe a system for detecting tie plates and spikes. Their method, which is described in more detail in [26], uses an AdaBoost-based object detector [27] with a model selection mechanism that assigns the object class that produces the highest number of detections within a window of 50 frames.

Table 2.1 in chapter 2 summarizes several methods for inspecting track components

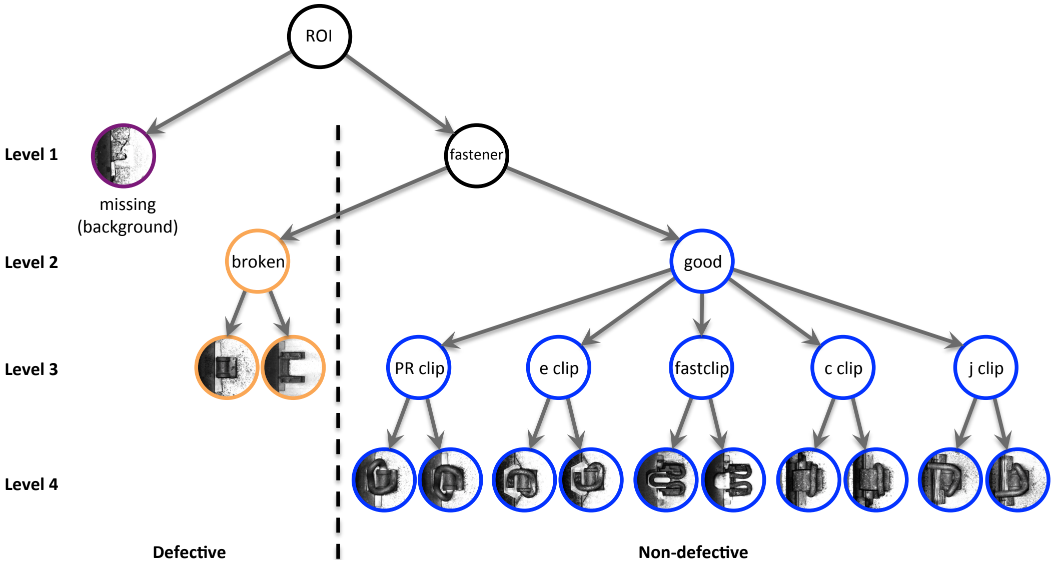


Figure 4.2: Object categories used for detection and classification (from coarsest to finest levels).

described in the literature. In addition to the works described in this section, there are other commercial vendors that offer automated visual track inspection systems, but they have not disclosed the techniques that they use nor their detection performance. More details about these and other methods can be found in the surveys by Molina and Edwards [68], and Podder [69].

4.3 Proposed Approach

In this section, we describe the details of our proposed approach to automatic fastener detection.

4.3.1 Overview

Due to surface variations that result from grease, rust and other elements in the outdoor environment, segmentation of railway components is a very difficult task. Therefore, we avoid it by using a detector based on a sliding window that we run over the inspectable area of the tie. The detector uses the well-know descriptor based on the Histograms of Oriented Gradients [70] (HOG), which was originally designed for pedestrian detection, but it has been proven effective for a variety of object detection tasks in unconstrained environments. Although, most of the time, fasteners are located very close to the rail, we need to search over a much broader area because on turnouts (switches and frogs) fasteners are positioned farther away from the rail, with more varied configurations.

4.3.2 Classification

Our goal is to simultaneously detect, within each predefined Region of Interest (ROI), the most likely fastener location and then classify such detections into one of three basic conditions: background (or missing fastener), broken fastener, and good fastener. Then, for good and broken fastener conditions, we want to assign class labels for each fastener type (PR clip, e-clip, fastclip, c-clip, and j-clip). Figure 4.2 shows the complete categorization that we use, from coarsest to finest. At the coarsest level, we want to classify fastener vs. unstructured background clutter. The background class also includes images of ties where fasteners are completely missing. We have done this for these reasons: 1) it is very difficult to train a detector to find the small hole left on the tie after the whole fastener has been ripped off, 2) we do not have enough training examples of missing fasteners, and 3) most missing fasteners are on crumbled ties for which the hole is no longer visible. Once we detect the most likely fastener location, we want to classify the detected fastener between broken vs. good, and then classify it into the most likely fastener type. Although this top-down reasoning works for a human inspector, it does not work accurately in a computer vision system because both the background class and the fastener class have too much intra-class variations. As a result, we have resorted to a bottom-up approach.

Since we use inner products, our detector may resemble the correlation-based approach used in [22], but there are three key differences that sets us apart: 1) our input is a HOG feature vector rather than raw pixel intensities, 2) we use a linear SVM to learn the coefficients of the detection filter, 3) we use a second classifier to reject misclassified fastener types.

To achieve the best possible generalization at test time, we have based our detector on the maximum margin principle of the SVM. Although SVM is a binary classifier, it is straightforward to build a multi-class SVM, for example, by combining several one-vs-rest or one-vs-one SVM classifiers, either by a voting scheme or by using the DAG-SVM framework. Our approach uses the one-vs-rest strategy, but instead of treating the background class as just another object class, we treat it as a special case and use a pair of SVMs per object class. For instance, if we had used a single learning machine, we would be forcing the classifier to perform two different unrelated tasks: a) reject the hypothesis that the image patch that does not contain random texture and b) reject the hypothesis that the object does not belong to the given category. Therefore, given a set of object classes $\mathcal{C}$, we train two detectors for each object category. The first one, with weights b_c, classifies each object class $c \in \mathcal{C}$ vs. the background/missing class $m \notin \mathcal{C}$, and the second one, with weights f_c classifies object class c vs. other object classes $\mathcal{C} \backslash c$. As illustrated in Figure 4.3, asking our linear classifier to perform both tasks at the same time would result in a narrower margin than training separate classifiers for each individual task. Moreover, to avoid rejecting cases where all f_c classifiers produce negative responses, but one or more b_c classifiers produce strong positive responses that would otherwise indicate the presence of a fastener, we use the negative output of f_c as a soft penalty. Then the likelihood that sample x belongs to class c becomes

$$L_c(x) = b_c \cdot x + \min(0, f_c \cdot x), \tag{4.1}$$

where $x = HOG(I)$ is the feature vector extracted from a given image patch I. The likelihood that our search region contains at least one object of class c is the score of the union, so

$$L_c = \max_{x \in \mathcal{X}} L_c(x), \tag{4.2}$$

where $\mathcal{X}$ is the set of all feature vectors extracted within the search region, and our classification rule becomes

$$\hat{c} = \begin{cases} \arg\max\limits_{c \in \mathcal{C}} L_c & \max\limits_{c \in \mathcal{C}} L_c > 0 \\ m & \text{otherwise.} \end{cases} \tag{4.3}$$

4.3.3 Score Calculation

For the practical applicability of our detector, it needs to output a scalar value that can be compared to a user-selectable threshold τ. Since there are several ways for a

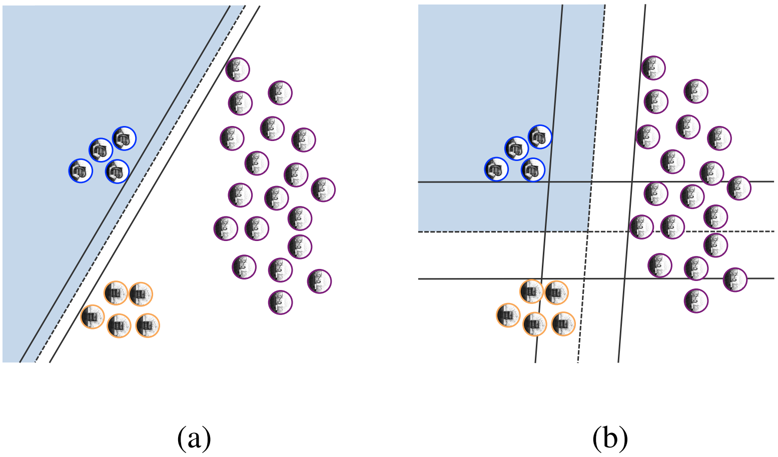


Figure 4.3: **Justification for using two classifiers for each object category.** (a) Classification region of fastener vs. rest (b) Classification region of intersection of fastener vs. background and fastener vs. rest-minus-background. The margin is much wider than using single classifier.

fastener to be defective (either missing or broken), we need to generate a single score that informs the user how confident the system is that the image contains a fastener in good condition. For the score calculation, we divide the set of object classes $\mathcal{C}$ into two mutually-exclusive subsets, one for good fasteners $\mathcal{G}$ and the other for broken fasteners $\mathcal{B}$, so $\mathcal{C} = \mathcal{G} \cup \mathcal{B}$ and $\mathcal{G} \cap \mathcal{B} = \emptyset$. We define the score for the missing fastener hypothesis as

$$S_m = \max_{c \in \mathcal{G}} L_c \tag{4.4}$$

and the score for the broken fastener hypothesis

$$S_b = - \max_{c \in \mathcal{B}} f_c \cdot x, \tag{4.5}$$

where we invert the sign of the score S_b to reflect the convention that a fastener in good condition should have a large positive score. The final score becomes the intersection of these two scores

$$S = \min(S_m, S_b). \tag{4.6}$$

The final decision is done by reporting the fastener as good if $S > \tau$, and defective otherwise.

4.3.4 Training Procedure

The advantage of using a maximum-margin classifier is that once we have enough support vectors for a particular class, it is not necessary to add more inliers to improve classification performance. Therefore, we can potentially achieve relatively good performance with only having to annotate a very small fraction of the data. To generate our training set, we initially selected $\sim$30 good quality (with no occlusion and clean edges) samples from each object category at random from the whole repository and annotated the bounding box location and object class for each of them. Our training software also automatically picks, using a randomly generated offset, a background patch adjacent to each of the selected samples. Once we had enough samples from each class, we trained binary classifiers for each of the classes against the background and tested on the whole dataset. Then, we randomly selected misclassified samples and added those that had good or acceptable quality (no occlusion) to the training set. To maintain the balance of the training set, we also added, for each difficult sample, 2 or 3 neighboring samples. Since there are special types of fasteners that do not occur very frequently (such as the c-clips or j-clips used around joint bars), in order to keep the number of samples of each type in the training set as balanced as possible, we added as many of these infrequent types as we could find.

4.3.5 Alignment Procedure

For learning the most effective object detection models, the importance of properly aligning the training samples cannot be emphasized enough. Misalignment between different training samples would cause unnecessary intra-class variation that would degrade detection performance. Therefore, all the training bounding boxes were manually annotated, as tightly as possible to the object contour by the same person to avoid inducing any annotation bias. For training the fastener vs. background detectors, our software cropped the training samples using a detection window centered around these boxes. For training the fastener vs. rest detectors, our software centered the positive samples using the user annotation, but the negative samples were re-centered to the position where the fastener vs. background detector generated the highest response. This was done to force the learning machine to learn to differentiate object categories by taking into account parts that are not common across object categories.

4.4 Experimental Results

To evaluate the accuracy of our fastener detector, we have tested it on 85 miles of continuous trackbed images. These images were collected on the US Northeast Corridor (NEC) by ENSCO Rail's Comprehensive Track Inspection Vehicle (CTIV) (see Figure 4.4). The CTIV is a hi-rail vehicle (a road vehicle that can also travel on railway tracks) equipped with several track inspection technologies, including a Track Component Imaging System (TCIS). The TCIS collects images of the trackbed using 4 Basler sprint (spL2048-70km) linescan cameras and a custom line scan lighting solution [71].

The sprint cameras are based on CMOS technology and use a CameraLink interface to stream the data to a rack-mounted computer. Each camera contains a sensor with 2 rows of 2,048 pixels that can sample at line rates of up to 70KHz. The cameras can be set to run in dual-line mode (high-resolution) or in "binned" mode, where the values of each pair of pixels are averaged inside the sensor. During this survey, the cameras were set up in binned mode so, each camera generated a combined row of 2,048 pixels at a line rate of 1 line/0.43mm. The sampling rate was controlled by the signal generated from a BEI distance encoder mounted on one of the wheels. The camera positions and optics were selected to cover the whole track with minimal perspective distortion and their fields of view had some overlap to facilitate stitching.

The collected images were automatically stitched together and saved into several files, each containing a 1-mile image. These files were preprocessed by ENSCO Rail using their proprietary tie detection software to extract the boundary of all the ties in each image. We verified that the tie boundaries were accurate after visually correcting invalid tie detections. We downsampled the images by a factor of 2, for a pixel size of 0.86 mm. To assess the detection performance under different operating conditions, we flagged the special track sections where the fastener visible area was less than 50% due to a variety of occluding conditions, such as protecting covers for track-mounted equipment or ballast accumulated on the top of the tie. We also flagged turnouts so we could report separate ROC curves for both including and excluding them. All the ties in this dataset are made of reinforced concrete, were manufactured by either San-Vel or Rocla, and were installed

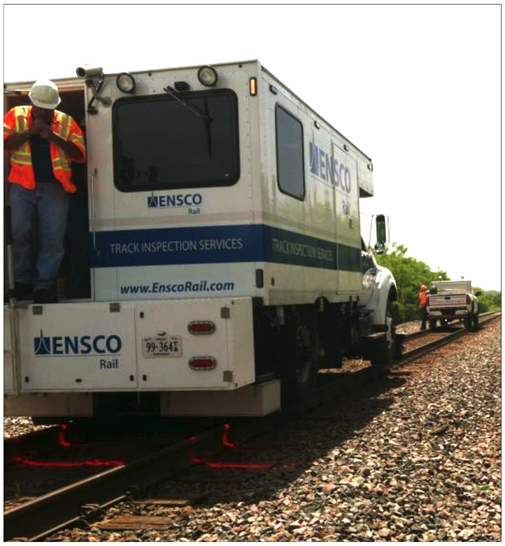


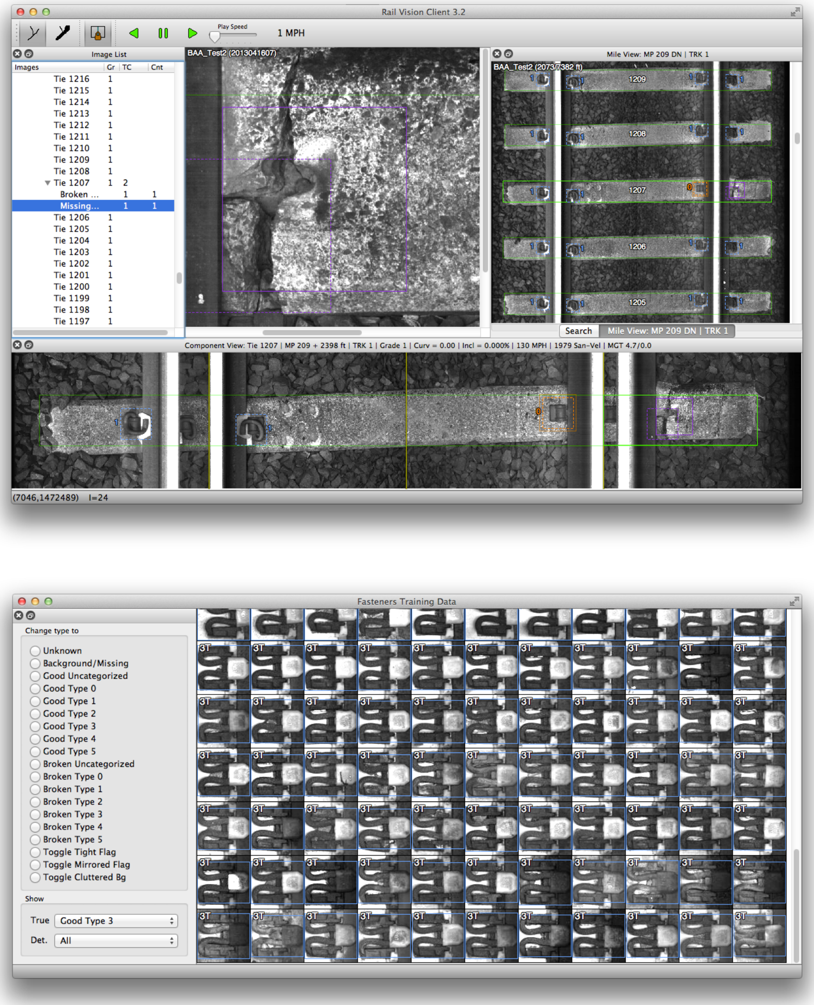
Figure 4.4: CTIV platform used to collect the images.

between 1978 and 2010.

Due to the large size of this dataset, we have implemented a customized software tool that allows the user to efficiently visualize and annotate the data (see Figure 4.5 for a screenshot). This tool has been implemented in C++ using the Qt framework and communicates with the data repository through the secure HTTPS protocol, so it can be used from any computer with an Internet connection without having to set up tunnel or VPN connections. The tool allows the user to change the threshold of the defect detector and select a subset of the data for display and review. It also has the capability of exporting lists of detected defects as well as summaries of fastener inventories by mile.

4.4.1 Fastener Categorization

On our dataset, we have a total of 8 object categories (2 for broken clips, 1 for PR clips, 1 for e-clips, 2 for fast clips, 1 for c-clips, and 1 for j-clips) plus a special



Images	Gr	TC	Cnt
Tie 1216	1		
Tie 1215	1		
Tie 1214	1		
Tie 1213	1		
Tie 1212	1		
Tie 1211	1		
Tie 1210	1		
Tie 1209	1		
Tie 1208	1		
▼ Tie 1207	1	2	
Broken …		1	1
Missing…		1	1
Tie 1206	1		
Tie 1205	1		
Tie 1204	1		
Tie 1203	1		
Tie 1202	1		
Tie 1201	1		
Tie 1200	1		
Tie 1199	1		
Tie 1198	1		
Tie 1197	1		

Component View: Tie 1207 | MP 209 + 2398 ft | TRK 1 | Grade 1 | Curv = 0.00 | Incl = 0.000% | 130 MPH | 1979 San-Vel | MGT 4.7/0.0

(7046,1472489) I=24

Change type to

○ Unknown
○ Background/Missing
○ Good Uncategorized
○ Good Type 0
○ Good Type 1
○ Good Type 2
○ Good Type 3
○ Good Type 4
○ Good Type 5
○ Broken Uncategorized
○ Broken Type 0
○ Broken Type 1
○ Broken Type 2
○ Broken Type 3
○ Broken Type 4
○ Broken Type 5
○ Toggle Tight Flag
○ Toggle Mirrored Flag
○ Toggle Cluttered Bg

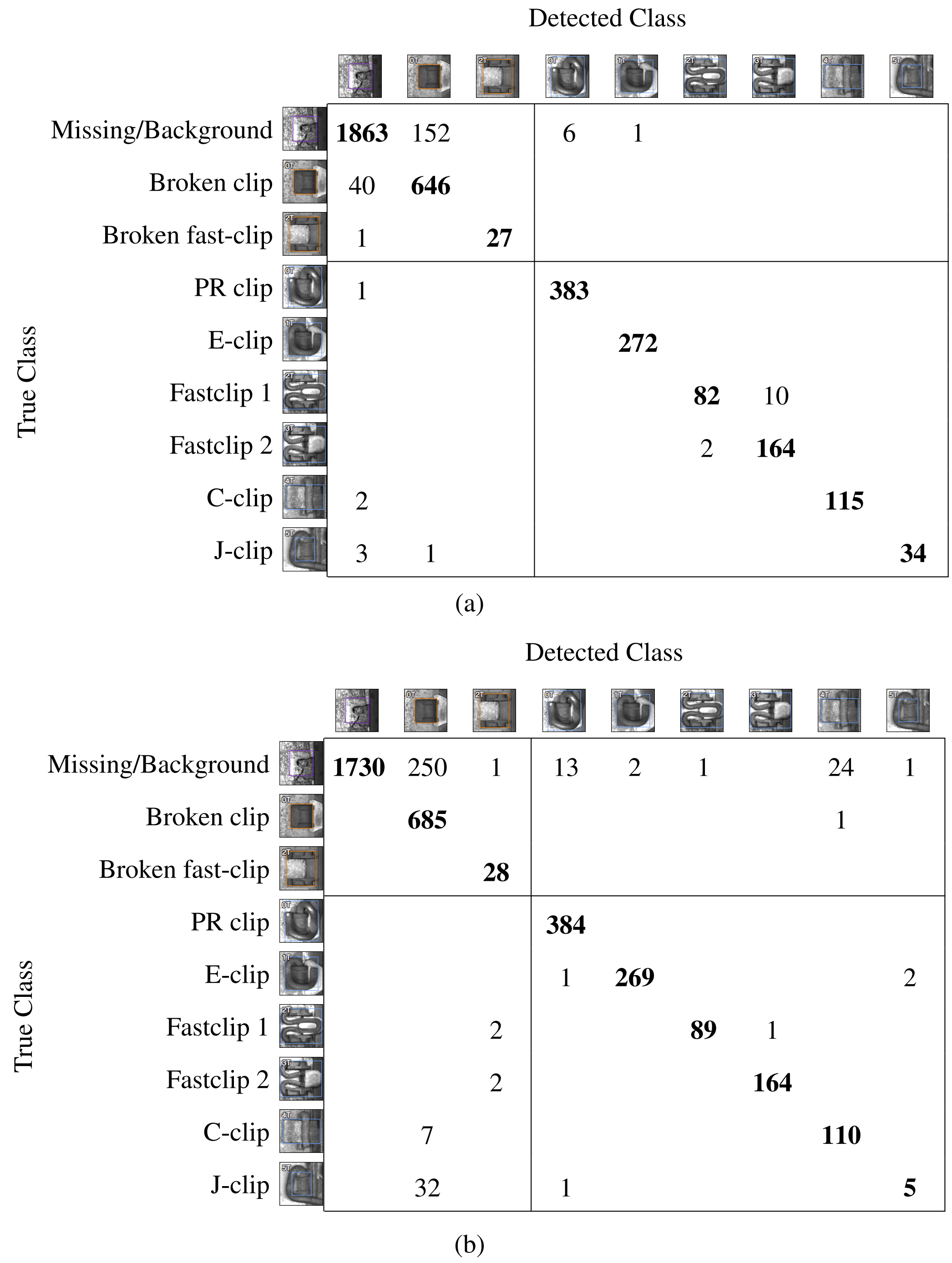
Show

True: Good Type 3

Det.: All

Figure 4.5: The GUI tool used to generate the training set and to review the detection results.

category for background (which includes missing fasteners). We also have 4 synthetically generated categories by mirroring non-symmetric object classes, so we use a total of 12 object object categories at test time. The HOG features are extracted using a 160×160 pixel sliding window with a strap of 8×8. We use the HOG implementation in the object detection module of OpenCV using default parameters. For classification, we use the linear SVM implementation in the machine learning module of OpenCV (which is derived from *LIBSVM* [72]) with a soft margin ($C = 0.01$).



True Class	Detected Class								
Missing/Background	**1863**	152		6	1				
Broken clip	40	**646**							
Broken fast-clip	1		**27**						
PR clip	1			**383**					
E-clip					**272**				
Fastclip 1						**82**	10		
Fastclip 2						2	**164**		
C-clip	2							**115**	
J-clip	3	1							**34**

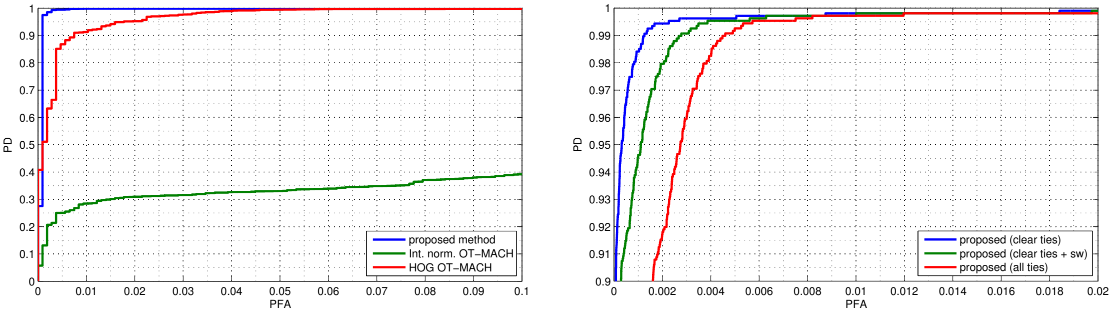
(a)

True Class	Detected Class								
Missing/Background	**1730**	250	1	13	2	1		24	1
Broken clip		**685**						1	
Broken fast-clip			**28**						
PR clip				**384**					
E-clip				1	**269**				2
Fastclip 1			2			**89**	1		
Fastclip 2			2				**164**		
C-clip		7						**110**	
J-clip		32		1					**5**

(b)

Figure 4.6: **Confusion matrix on 5-fold cross-validation of the training set using** (a) the proposed method (b) the method described in [22] with HOG features.

For training our detectors, we used a total of 3,805 image patches, including 1,069



(a)

(b)

Figure 4.7: **ROC curves for the task of detecting defective (missing or broken) fasteners** (a) using 5-fold cross-validation on the training set (b) on the 85-mile testing set.

good fasteners, 714 broken fasteners, 33 missing fasteners, and 1,989 patches of background texture. During training, we also included the mirrored versions of the missing/background patches and all symmetric object classes. To evaluate the feasibility of the algorithm, we performed 5-fold cross-validation on the training set, where we classified each patch into one of the 9 basic object categories (we excluded the 4 artificially generated mirrored categories). Figure 4.6 (a) shows the resulting confusion matrix. We only had 14 misclassified samples (0.37% error rate). If we consider the binary decision problem of finding defective fasteners (either missing or broken), we have a detection rate of 99.74% with a false alarm rate of 0.65%. This is an encouraging result, since as explained in section 4.3.4, our training set has been bootstrapped to contain many difficult samples.

To compare our method with previous work, we implemented the correlation-based approach described in [22]. However, since the OT-MACH approach on normalized image intensity did not produce very good results (see Figure 4.7 (a)), we modified the algorithm to work on HOG features. Figure 4.6 (b) shows the resulting confusion matrix.

This method had an error rate of 2.23% (6 times greater than our proposed method). The detection rate was 98.86% with a false alarm rate of 4.02%. We can see that j-clips and c-clips are the most difficult types of fasteners. These 2 types of fasteners contain more intra-class variation than other types because they are placed next to joint bars, so some of them are slightly rotated to accommodate the presence of joint bar bolts.

4.4.2 Defect Detection

To evaluate the performance of our defect detector, we divided each tie into 4 regions of interest (left field, left gage, right gage, right field) and calculated the score defined by ((4.6)) for each of them. Figure 4.7 shows the ROC curve for crossvalidation on the training set as well as for the testing set of 813,148 ROIs (203,287 ties). The testing set contains 1,051 ties images with at least one defective fastener per tie. The total number of defective fasteners in the testing set was 1,086 (0.13% of all the fasteners), including 22 completely missing fasteners and 1,064 broken fasteners. The number of ties that we flagged as "uninspectable" is 2,524 (1,093 on switches, 350 on lubricators, 795 covered in ballast, and 286 with other issues).

We used the ROC on clear ties (blue curve) in Figure 4.7 (b) to determine the optimal threshold to achieve a design false alarm rate of 0.1% ($\tau = 0.1614$). Using this sensitivity level, we ran our defective fastener detector at the tie level (by taking the minimum score across all 4 regions). Results are shown in table 4.1.

Our protocol has been to mark the whole tie as uninspectable if at least one of the fasteners is not visible in the image. This is not ideal as there are situations where parts

Table 4.1: Results for detection of ties with at least one defective fastener.

Subset	Total ties	Defective	PD	PFA
clear ties	200,763	1,037	98.36%	0.38%
clear + switch	201,856	1,045	97.99%	0.71%
all ties	203,287	1,051	98.00%	1.23%

of the tie are still inspectable, for example when the field side of the rail is covered with ballast, but the gage side is inspectable (this explains the 6 additional defective ties when including uninspectable ties).

4.5 Summary

In order to advance the state-of-the-art in automated railway fastener inspection, our design has been driven by the fundamental principle of projecting the samples into a representation that minimizes intra-class variation while maximizing inter-class separation. To achieve minimum intra-class variation, we use the HOG features, which have built-in intensity normalization, while preserving the distinctive distribution of edges. We have also implemented a sophisticated graphical user interface that facilitates accurate alignment of the fastener locations to avoid intraclass variations due to misalignment. To achieve maximum inter-class separation while maintaining the principle of parsimony, we resort to the maximum margin formulation and simplicity offered by linear SVMs. We further enforce intra-class separation during the sampling of the training data. For the fastener-vs-background classifiers we bootstrapped difficult samples when we built the training set. For the fastener-vs-rest classifiers, we aligned the negative samples to the most confusing position, so the learning machine could focus on the best way to separate classes on the most distinctive parts of the object.

In summary, we believe that the system described here is a good step towards automated visual track inspection and will help railroads maintain their tracks in optimal conditions. Possible extensions to this work will be discussed in chapter 7.

Chapter 5: Deep Learning Methods for Anomaly detection

5.1 Background

5.1.1 Convolutional Neural Networks

The idea of enforcing translation invariance in neural networks via weight sharing goes back to Fukoshima's Neocognitron [73]. Based on this idea, LeCun *et al.* developed the concept into Deep Convolutional Neural Networks (DCNN) and used it for digit recognition [74], and later for more general optical character recognition (OCR) [75]. During the last few years, DCNNs have become ubiquitous in achieving state-of-the-art results in image classification [76, 77] and object detection [78]. This resurgence of DCNNs has been facilitated by the availability of efficient GPU implementations and open source libraries such as Caffe [79] and Torch7 [80]. More recently, DCNNs have been used for semantic image segmentation. For example, the work of [81] shows how a DCNN can be converted to a Fully Convolutional Network (FCN) by replacing fully-connected layers with convolutional ones.

5.1.2 Multi-task Learning

Multi-task learning (MTL) is an inductive transfer learning technique in which two or more learning machines are trained cooperatively [82]. It is a generalization of multi-label learning in which each training sample has only been labeled for one of the tasks. In MTL settings, there is a mechanism in which knowledge learned for one task is transferred to the other tasks [83]. The idea is that each task can benefit by reusing knowledge that has been learned while training for the other tasks. Backpropagation has been recognized as an effective method for learning distributed representations [84]. For instance, in multitask neural networks, we jointly minimize one global loss function

$$\Phi = \sum_{t=1}^{T} \lambda_t \sum_{i=1}^{N_t} E_t \left(f(x_{ti}), y_{ti} \right) \tag{5.1}$$

where T is the number of tasks, N_t is the number of training samples for task t, y_{ti} is the ground truth label for training sample x_{ti}, f is the the multi-output function computed by the network, and E_t is the loss function for task t. This contrasts with the Single Task Learning (STL) setting, in which we minimize T independent loss functions

$$\Phi_t = \sum_{i=1}^{N_t} E_t \left(f_t(x_{ti}), y_{ti} \right), \quad t \in \{1 \dots T\} \tag{5.2}$$

In MTL, the weighting factor λ_t is necessary to compensate for imbalances in the complexity of the different tasks and the amount of training data available. When using back-propagation, it is necessary to adjust λ_t's to ensure that all tasks are learning at optimal rates.

5.1.3 One-shot Learning

To achieve good generalization performance, traditional machine learning methods require a minimum number of training examples from each class. This is necessary for the machine to learn a model that can handle variations in image appearance that result from changes in illumination, scale, rotation, background clutter, and so on. However, the occurrence of each type of anomaly is very infrequent, so in anomaly detection settings it is only possible to find one or a few number of examples from which to learn from. If we try to learn a complete model for a new class using such a limited number of examples, this model would overfit and would not be able to generalize to new data. However, if we reuse knowledge that has been learned while learning other related classes, we can learn better models. This is known as one-shot learning [2]. We pose this one-shot learning problem as a special case of multi-task learning, in which one task consists of learning the abundant classes, while the other task learns the uncommon classes.

In coarse to fine-grained object categorization problems, such as anomaly detection on objects with multiple configurations, both the task of detecting and classifying the objects as well as determining whether each object is in good or bad condition, share a common low-level representation because all object classes are made of common parts. In the railway fastener inspection application described in this chapter, we will train an auxiliary network on a 5-class fastener classification using more than 300K fasteners for the sole purpose of learning a good representation that regularizes the broken fastener detector.

5.2 Learning with Weakly Labeled Data

Learning with ambiguously labeled data refers to the learning problem where each training sample has multiple labels and only one of them is the correct one [85]. A related problem is Multiple Instance Learning (MIL) [86]. MIL refers to the learning problem in which the training data is not annotated at the instance level, but instead there are bags of instances and there is only one label per bag. These labels indicate that either the bag contains only instances from one class (the normal class), or whether it contains at least one instance from another class (the anomaly class). A popular approach for addressing MIL is the Diverse Density (DD) framework by Maron and Pérez [87], which finds similar instances of the ambiguous samples across different bags to determine which instances are likely to be from the alternative class. More recently, Shrivastava et al. [88] proposed a general DD-based algorithm using dictionaries that is more robust than previous approaches.

For example, in our experiments in chapter 4, due to the requirements of fully-annotated data, we could only use 1,816 samples for training, while we used $\sim$200K samples for testing. There are many other applications where the availability of fully-annotated data is very limited, but there are vast amounts of unlabeled as well as weakly labeled data. For weakly labeled data we refer to cases where the exact data label is unknown or the instance the label corresponds to is unknown. For instance, the ground truth label categories are broad, such as good vs. anomalous, and the location and type of each image element is unknown. For those images that are deemed to be anomalous, the specific image elements that constitute the anomaly, as well as the type of anomaly are also unknown. In contrast to the direct algorithms for learning from fully-annotated data presented in previous chapters, learning from incomplete training data, such as when there is a dependency on hidden or latent parameters is much harder. The main issue is that simultaneously learning the model and the latent parameters is usually a highly non-convex optimization problem. Therefore, the parameters learned using descend algorithms are only guaranteed to be locally optimal and the quality of the estimate depends on the initialization of such latent parameters.

The MTL framework described in this chapter allows us to handle weakly labeled data. The primary task is the estimation of the full label and is trained with the subset of the data that contains exact labels. The secondary task is trained on the subset of the data that only contains weak labels. The training objective of the secondary task will be different from exact label prediction. For example, in the ambiguously labeled setting, we could use one of the following training objectives for the secondary task:

- *Reduction to multi-label learning using super-classes:* If classes can be clustered into groups that often appear together, the problem of learning with ambiguously labeled data could be converted into a multi-label learning problem. In this setting, the set of multiple ambiguous class labels will be converted to a set multiple super-class labels. Each super-class label will set to 1 if all the ambiguous labels belong to such super-class or 0 if none of the labels belong to the super-class. Otherwise, the label for such superclass would be left undefined and the value of the loss function would be set to constant zero for predicting such class.

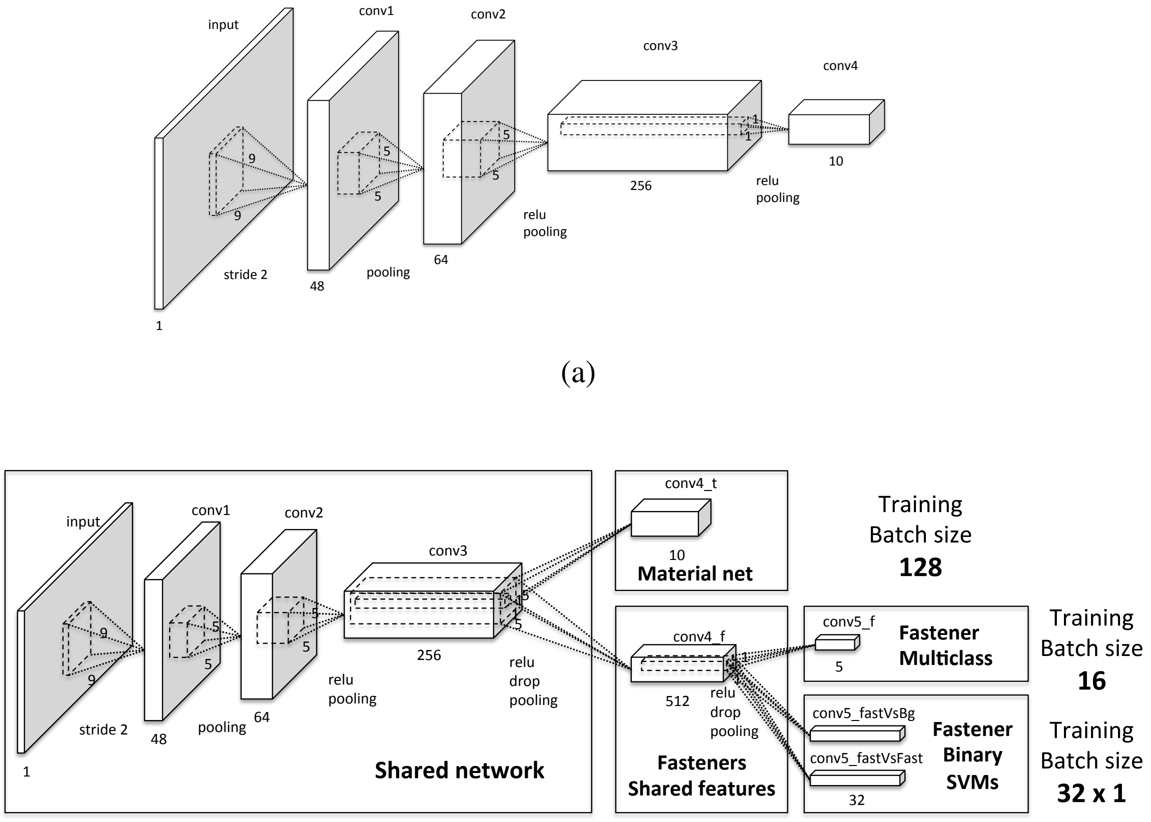
- *Reduction to several one-vs-rest binary problems:* Train a number of binary classifiers where each classifier learns whether a sample belongs to a group of classes or not.

In the multiple instance learning setting, we could add a max-pooling layer that takes predictions for each sample in the bag as input and generates a single prediction indicating whether the bag contains at least one sample of the class being tested.

The extreme case is when some of the data is unlabeled (the semi-supervised learning case). The motivation of using unlabeled data is that, while collecting weakly labeled data may be one or two orders of magnitude cheaper than collecting fully annotated data, collecting unlabeled data is virtually free, as it requires very little human intervention. In this scenario, the unlabeled data can still be used to learn a representation. In this case, the secondary learning task would be selected to enforce some desirable property in this representation, such as invariance to some transformation or sparsity of such representation.

5.3 Overall Architecture

Our design is a Fully Convolutional Network [81] based on the architecture introduced in [7]. That network was trained with 10 classes of materials and produces feature maps with 10 different channels. In this chapter, we extend that architecture by adding two additional branches to the network. The first one is a coarse-level fastener classifier trained on a large number of examples. The second branch produces 32 binary outputs. These outputs correspond to the same binary SVMs that we used in our previous version of the detector introduced in chapter 4.



(a)

(b)

Figure 5.1: Network architectures evaluated in this chapter. (a) Single-task learning (material classification only) (b) Multi-task learning (material and fasteners)

The implementation is based on the BVLC Caffe framework [79]. For the material classification task, we have a total of 4 convolutional layers between the input and the output layer, while for fastener detection tasks we have 5 convolutional layers. The first three layers are shared among all the tasks. The fasteners task is, in turn, divided in two subtasks: coarse-level and fine-grained classification (see section 5.5 for more details). The network uses rectified linear units (ReLU) as non-linear activation functions, and overlapping max pooling units of size 3×3. All max pooling units have a stride of 2, except the one on top of that has a stride of 1. We use dropout [89] regularization on layer

3 (with a ratio of 0.1) and layer 4 on the fasteners branch (with a ratio of 0.2). The network also uses weight decay regularization. On the fasteners branch, we increase the weight decay factors on layers 4 and 5 by $10\times$ and $100\times$ respectively to reduce overfitting.

We first apply global gain normalization on the raw image to reduce the intensity variation across the image. This gain is calculated by smoothing the signal envelope estimated using a median filter. We estimate the signal envelope by low-pass filtering the image with a Gaussian kernel. Although DCNNs are robust to illumination changes, normalizing the image to make the signal dynamic range more uniform improves accuracy and convergence speed. We also subtract the mean intensity value, which is calculated on the whole training set. The network architecture is illustrated in figure 5.1.

5.3.1 Data Annotation

In section 4.4 we introduced the customized software tool that we implemented to efficiently visualize and annotate the data. The tool allows assigning a material category to each tie as well as its bounding box. It also allows defining polygons enclosing regions containing crumbling, chips or ballast. We used the output of our fastener detection algorithm [6] to extract fastener examples. The tool also allows the user to change the threshold of the defect detector and select a subset of the data for display and review. It also has the capability of exporting lists of detected defects as well as summaries of fastener inventories by mile.

5.3.2 Training Procedure

The training set used for material classification is exactly the same that used in [7]. The training set for fastener classification is the one used in chapter 4. As we described in the preceding chapter, to generate our training set, we initially selected $\sim$30 good quality (with no occlusion and clean edges) samples from each object category at random from the whole repository and annotated the bounding box location and object class for each of them. Our training software also automatically picks, using a randomly generated offset, a background patch adjacent to each of the selected samples. Once we had enough samples from each class, we trained binary classifiers for each of the classes against the background and tested on the whole dataset. Then, we randomly selected misclassified samples and added those that had good or acceptable quality (no occlusion) to the training set. To maintain the balance of the training set, we also added, for each difficult sample, 2 or 3 neighboring samples. Since there are special types of fasteners that do not occur very frequently (such as the c-clips or j-clips used around joint bars), in order to keep the number of samples of each type in the training set as balanced as possible, we added as many of these infrequent types as we could find.

After spending several days carefully annotating the fasteners, our training set only contains 2819 fully-annotated fasteners. Moreover, some of the classes had very few examples. For instance, there are only 28 broken fast-clips, and just 38 j-clips in all the data we have. Had we added more examples from the abundant classes, we would have made the imbalance problem even worse. On the other hand, if we just had used this limited data, we would not have been able to learn a good representation. Fortunately, both of these two uncommon classes of fasteners share parts with the other ones. Therefore, if we can make layer *conv4_f* learn a good model for fastener parts, layer *conv5_f* would be able to learn how to distinguish between fasteners by combining such parts, even if the number of training examples is limited.

Therefore, we created an auxiliary fastener data set. Since the only purpose of this dataset is to help learn parts, we just used the bounding boxes and labels automatically generated by our previous detector [6], whose error rate is just 0.37%. We sampled 62,500 fasteners from each of 5 coarse classes. The first class contains missing and broken fasteners, the next 3 classes contain fasteners corresponding to each of the classes containing the most samples (PR-clips, e-clips, and fast-clips), and the last class contains everything else.

We train the network using stochastic gradient descent on mini-batches of 128 image patches of size 75×75 plus 48 fastener images of 182×182. The fastener images include 16 from the auxiliary fastener dataset and 1 from each of the binary SVM tasks. We do data augmentation on material classification by randomly mirroring vertically and/or horizontally the training samples. The patches are cropped randomly among all regions that contain the texture of interest. To increase robustness against adverse environment conditions, such as rain, grease or mud, we identified images containing such difficult cases and automatically resampled the data so that at least 50% of the data is sampled from such difficult images. We do data augmentation on fasteners by randomly mirroring vertically the symmetric classes and randomly cropping the fastener offset uniformly distributed within a +/-9 pixel range in both directions.

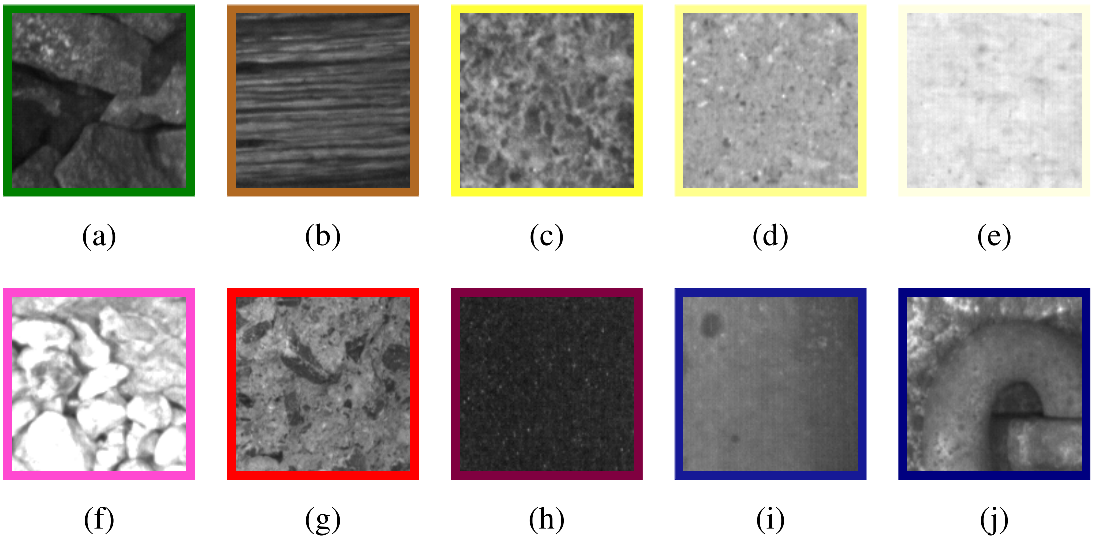


Figure 5.2: Material categories. (a) ballast (b) wood (c) rough concrete (d) medium concrete (e) smooth concrete (f) crumbling concrete (g) chipped concrete (h) lubricator (i) rail (j) fastener

5.4 Material Identification and Segmentation Task

5.4.1 Architecture

The material classification task at layer *conv4_t* generates ten score maps at 1/16th. Each value $\Phi_i(x, y)$ in the score map corresponds to the likelihood that pixel location (x, y) contains material of class i. The ten classes of materials are defined in Figure 5.2.

5.4.2 Score Calculation

To detect whether an image contains a broken tie, we first calculate the scores at each site as

$$S_b(x, y) = \max_{i \notin \mathcal{B}} \Phi_i(x, y) - \Phi_b(x, y) \tag{5.3}$$

where $b \in \mathcal{B}$ is a defect class (crumbling or chip). Then we calculate the score for the whole image as

$$S_b = \frac{1}{\beta - \alpha} \int_{\alpha}^{\beta} \widehat{F}^{-1}(t) dt \tag{5.4}$$

where $\widehat{F}^{-1}$ refers to the t sample quantile calculated from all scores $S_b(x, y)$ in the image. The detector reports an alarm if $S > \tau$, where τ is the detection threshold. We used $\alpha = 0.9$ and $\beta = 1$.

5.5 Fasteners Assessment Task

In this section, we describe the details of the fastener assessment task.

5.5.1 Overview

Due to surface variations that result from grease, rust and other elements in the outdoor environment, segmentation of railway components is a very difficult task. Therefore, we avoid it by using a detector based on a sliding window that we run over the inspectable area of the tie. The features learned at layer *conv4_f* are computed from the shared features at *conv3*. The reason for sharing the features with the material classification task is that there is overlap between both tasks. For instance, the material classification task needs to learn to distinguish between fasteners and the other materials, regardless of the type of fastener. Also, the fastener detection class needs to discriminate between fasteners and background, regardless of the type of background. In our previous work, we used the Histogram of Oriented Gradients (HOG) [70] descriptor for detecting fasteners. Although the results that we obtained using HOG features were better than previously proposed methods, this approach still has its limitations. For instance, the dimensionality of the feature vector is very large (12,996), consuming a lot of memory and computational resources, and the linear classifier cannot handle occlusions very well. Therefore, in this chapter we attempt to learn the features by training the network end to end.

5.5.2 Classification

Our goal is to simultaneously detect, within each predefined Region of Interest (ROI), the most likely fastener location and then classify such detections into one of three basic conditions: background (or missing fastener), broken fastener, and good fastener. Then, for good and broken fastener conditions, we want to assign class labels for each fastener type (PR clip, e-clip, fastclip, c-clip, and j-clip). Figure 5.2 shows the complete categorization that we use, from coarsest to finest. At the coarsest level, we want to classify fastener vs. unstructured background clutter. The background class also includes images of ties where fasteners are completely missing. We have done this for these reasons: 1) it is very difficult to train a detector to find the small hole left on the tie after the whole fastener has been ripped off, 2) we do not have enough training examples of missing fasteners, and 3) most missing fasteners are on crumbled ties for which the hole is no longer visible. Once we detect the most likely fastener location, we want to classify the detected fastener between broken vs. good, and then classify it into the most likely fastener type. Although this top-down reasoning works for a human inspector, it does not work accurately in a computer vision system because both the background class and the fastener class have too much intra-class variations. As a result, we have resorted to a bottom-up approach.

To achieve the best possible generalization at test time, we have based our detector on the maximum margin principle of the SVM. The SVM separating hyperplane is obtained by minimizing the regularized hinge loss function,

$$E = \sum_i \max\left(0, 1 - y_i(w \cdot x_i + b)\right) + \frac{\lambda}{2}\|w\| \tag{5.5}$$

where $x_i \in \mathbb{R}^{512}$ are the outputs of layer *conv4_f* and $y_i \in \{-1, +1\}$ their corresponding ground truth labels (whose meaning will be explain later). The gradients with respect to the parameters w and b are

$$\frac{\partial E}{\partial w} = -\sum_i y_i x_i \delta[y_i(w \cdot x_i + b) < 1] + \lambda w \tag{5.6}$$

$$\frac{\partial E}{\partial b} = -\sum_i y_i \delta[y_i(w \cdot x_i + b) < 1] \tag{5.7}$$

where $\delta\{\text{condition}\}$ is 1 if condition is true and -1 otherwise. The gradient of the hinge loss function with respect to the data (which is back-propagated down to the lower layers) is

$$\frac{\partial E}{\partial x_i} = -y_i w \delta[y_i(w \cdot x_i + b) < 1] \tag{5.8}$$

Once the parameters converge, these gradients become highly sparse and only the difficult training samples contribute to to updating the parameters on layer *conv4_f* and all the layers below.

Instead of training a multi-class SVM, we use the one-vs-rest strategy, but instead of treating the background class as just another object class, we treat it as a special case and use a pair of SVMs per object class. For instance, if we had used a single learning machine, we would be forcing the classifier to perform two different unrelated tasks: a) reject that the image patch that does not contain random texture and b) reject that the object does not belong to the given category. Therefore, given a set of object classes $\mathcal{C}$, we train two detectors for each object category. The first one, with weights b_c, classifies each object class $c \in \mathcal{C}$ vs. the background/missing class $m \notin \mathcal{C}$, and the second one, with weights f_c classifies object class c vs. other object classes $\mathcal{C} \backslash c$. As illustrated in Figure 4.3, asking our linear classifier to perform both tasks at the same time would result in a narrower margin than training separate classifiers for each individual task. Moreover, to avoid rejecting cases where all f_c classifiers produce negative responses, but one or more b_c classifiers produce strong positive responses that would otherwise indicate the presence of a fastener, we use the negative output of f_c as a soft penalty. Then the likelihood that sample x belongs to class c becomes

$$L_c(x) = b_c \cdot x + \min(0, f_c \cdot x), \tag{5.9}$$

where $x = HOG(I)$ is the feature vector extracted from a given image patch I. The likelihood that our search region contains at least one object of class c is the score of the union, so

$$L_c = \max_{x \in \mathcal{X}} L_c(x), \tag{5.10}$$

where $\mathcal{X}$ is the set of all feature vectors extracted within the search region, and our classification rule becomes

$$\hat{c} = \begin{cases} \arg\max_{c \in \mathcal{C}} L_c & \max_{c \in \mathcal{C}} L_c > 0 \\ m & \text{otherwise.} \end{cases} \tag{5.11}$$

5.5.3 Score Calculation

For the practical applicability of our detector, it needs to output a scalar value that can be compared to a user-selectable threshold τ. Since there are several ways for a fastener to be defective (either missing or broken), we need to generate a single score that informs the user how confident the system is that the image contains a fastener in good condition. This score is generated by combining the output of the binary classifiers introduced in the previous section.

We denote the subset of classes corresponding to good fasteners as $\mathcal{G}$ and that of broken fasteners as $\mathcal{B}$. These two subsets are mutually exclusive, so $\mathcal{C} = \mathcal{G} \cup \mathcal{B}$ and $\mathcal{G} \cap \mathcal{B} = \emptyset$. To build the score function, we first compute the score for rejecting the missing fastener hypothesis (i.e, the likelihood that there is at least one sample $x \in \mathcal{X}$ such that $x \notin m$) as

$$S_m = \max_{c \in \mathcal{G}} L_c \tag{5.12}$$

where L_c is the likelihood of class c defined in Eq. 5.10. Similarly, we compute the score for rejecting the broken fastener hypothesis (i.e, the likelihood that for each sample $x \in \mathcal{X}$, $x \notin \mathcal{B}$) as

$$S_b = -\max_{c \in \mathcal{B}} \max_{x \in \mathcal{X}} f_c \cdot x, \tag{5.13}$$

The reason why the S_b does not depend on a c-vs-background classifier b_c is because mistakes between missing and broken fastener classes do not need to be penalized. Therefore, S_b need only produce low scores when x matches at least one of the models in $\mathcal{B}$. The negative sign in S_b results from the convention that a fastener in good condition should have a large positive score. The final score becomes the intersection of these two scores

$$S = \min(S_m, S_b). \tag{5.14}$$

The final decision is done by reporting the fastener as good if $S > \tau$, and defective otherwise.

5.5.4 Training Procedure

The advantage of using a maximum-margin classifier is that once we have enough support vectors for a particular class, it is not necessary to add more inliers to improve classification performance. Therefore, we can potentially achieve relatively good performance with only having to annotate a very small fraction of the data.

5.5.5 Alignment Procedure

For learning the most effective object detection models, the importance of properly aligning the training samples cannot be emphasized enough. Misalignment between different training samples would cause unnecessary intra-class variation that would degrade detection performance. Therefore, all the training bounding boxes were manually annotated, as tightly as possible to the object contour by the same person to avoid inducing any annotation bias. For training the fastener vs. background detectors, our software cropped the training samples using a detection window centered around these boxes. For training the fastener vs. rest detectors, our software centered the positive samples using the user annotation, but the negative samples were re-centered to the position where the fastener vs. background detector generated the highest response. This was done to force the learning machine to learn to differentiate object categories by taking into account parts that are not common across object categories.

5.6 Experimental Results

To evaluate the accuracy of our fastener detector, we have tested it on 85 miles of continuous trackbed images. These images were collected on the US Northeast Corridor (NEC) by ENSCO Rail's Comprehensive Track Inspection Vehicle (CTIV) (see Figure 4.4). The CTIV is a hi-rail vehicle (a road vehicle that can also travel on railway tracks) equipped with several track inspection technologies, including a Track Component Imaging System (TCIS). The TCIS collects images of the trackbed using 4 Basler sprint (spL2048-70km) linescan cameras and a custom line scan lighting solution [71].

The sprint cameras are based on CMOS technology and use a CameraLink interface to stream the data to a rack-mounted computer. Each camera contains a sensor with 2 rows of 2,048 pixels that can sample at line rates of up to 70KHz. The cameras can be set to run in dual-line mode (high-resolution) or in "binned" mode, where the values of each pair of pixels are averaged inside the sensor. During this survey, the cameras were set up in binned mode so, each camera generated a combined row of 2,048 pixels at a line rate of 1 line/0.43mm. The sampling rate was controlled by the signal generated from a BEI distance encoder mounted on one of the wheels. The camera positions and optics were selected to cover the whole track with minimal perspective distortion and their fields of view had some overlap to facilitate stitching.

The collected images were automatically stitched together and saved into several files, each containing a 1-mile image. These files were preprocessed by ENSCO Rail using their proprietary tie detection software to extract the boundary of all the ties in each image. We verified that the tie boundaries were accurate after visually correcting invalid tie detections. We downsampled the images by a factor of 2, for a pixel size of 0.86 mm. To assess the detection performance under different operating conditions, we flagged special track sections where the fastener visible area was less than 50% due to a variety of occluding conditions, such as protecting covers for track-mounted equipment or ballast accumulated on the top of the tie. We also flagged turnouts so we could report separate ROC curves for both including and excluding them. All the ties in this dataset are made of reinforced concrete, were manufactured by either San-Vel or Rocla, and were installed between 1978 and 2010.

5.6.1 Material Identification

We divided the dataset into 5 splits and used 80% of the images for training and 20% for testing and we generated a model for each of the 5 possible training sets. For each split of the data, we randomly sampled 50,000 patches of each class. Therefore, for each model was trained with 2 million patches. We trained the network using a batch size of 64 for a total of 300,000 iterations with a momentum of 0.9 and a weight decay of 5×10^{-5}. The learning rate is initially set to 0.01 and it decays by a factor of 0.5 every 30,000 iterations.

In addition to the method described in Section 5.4, we evaluated the classification performance using the following methods:

- **LBP-HF with approximate Nearest Neighbor:** The Local Binary Pattern Histogram Fourier descriptor introduced in [90] is invariant to global image rotations while preserving local information. We used the implementation provided by the authors. To perform approximate nearest neighbor we used FLANN [91] with the 'autotune' parameter set to a target precision of 70%.

- **Uniform LBP with approximate Nearest Neighbor** The $LBP^{u2}_{8,1}$ descriptor [92] with FLANN.

- **Gabor features with approximate Nearest Neighbor:** We filtered each image with a filter bank of 40 filters (4 scales and 8 orientations) designed using the code from [93]. As proposed in [94], we compute the mean and standard deviation of the output of each filter and build a feature descriptor as $f = [\mu_{00}\ \sigma_{00}\ y_{01}\ \ldots\ \mu_{47}\ \sigma_{47}]$. Then, we perform approximate nearest neighbor using FLANN with the same parameters.

The material classification results are summarized in Table 5.1 and the confusion matrices in Figures 5.3 and 5.4.

Since we are using a fully convolutional DCNN, we directly transfer the parameters learned using small patches to a network that takes one 4096×320 image as an input, and generates 10 score maps of dimension 252×16 each. The segmentation map is generated by taking the label corresponding to the maximum score. Figure 5.6 shows several examples of concrete and wood ties, with and without defects and their corresponding segmentation maps.

Table 5.1: Material classification results.

Method	Accuracy
Deep CNN MTL 3	**95.02**%
Deep CNN MTL 2	93.60%
Deep CNN STL [7]	93.35%
LBP-HF with FLANN	82.05%
LBP$^{u2}_{8,1}$ with FLANN	82.70%
Gabor with FLANN	75.63%

5.6.2 Crumbling Tie Detection

The first 3 rows in Figure 5.6 show examples of a crumbling ties and their corresponding segmentation map. Similarly, rows 4 through 6 show examples of chipped ties. To evaluate the accuracy of the crumbling and chipped tie detector described in Section 5.4.2 we divide each tie in 4 images and we evaluate the score (5.4) on each image independently. Due to the large variation in the area affected by crumbling/chip we assigned a severity level to each ground truth defect, and for each severity level we plot the ROC curve of finding a defect when ignoring lower level defects. The severity levels are defined as the ratio of the inspectable area that is labeled as a defect. Figure 5.5 shows the ROC curves for each type of anomaly. Because of the choice of the fixed $\alpha = 0.9$ in equation (5.4) the performance is not reliable for defects under 10% severity. For defects that are bigger than the 10% threshold, at a false positive rate (FPR) of 10 FP/mile the true positive rates (TPR) are 89.42% for crumbling and 93.42% for chips. This is an improvement of 3.36% and 1.31% compared to the STL results reported in [7]. Table 5.2 summarizes the results.

Table 5.2: Tie condition detection. For chipped and crumbling, only ties with at least 10% affected area are included.

Condition	FPR	MTL	STL
Crumbling Tie ($\geq 10\%$ area)	10 FP/mile	**89.42**%	86.54%
	2 FP/mile	**82.21**%	74.52%
Chipped Tie ($\geq 10\%$ area)	10 FP/mile	92.76%	**94.08**%
	2 FP/mile	**90.13**%	88.52%
Fastener (only clear ties)	10 FP/mile	**99.91**%	98.41%
	2 FP/mile	**96.74**%	93.19%
Fastener (clear + switch)	10 FP/mile	**98.43**%	94.54%
	2 FP/mile	**89.35**%	88.70%
Fastener (all ties)	10 FP/mile	**95.40**%	87.38%
	2 FP/mile	**87.76**%	–

5.6.3 Fastener Categorization

To evaluate the fastener categorization talk of the multi-task network, we followed the same procedure as we described in Section 4.4.1.

We can observe in Figure 5.7 (a) that the proposed method is the most accurate, followed by the method described in chapter 4 and the HOG with OT-MACH method. The other methods are clearly inferior. In the third row of Table 5.2 we compare the fastener detection performance of MTL with the STL baseline.

5.6.4 Defective Fastener Detection

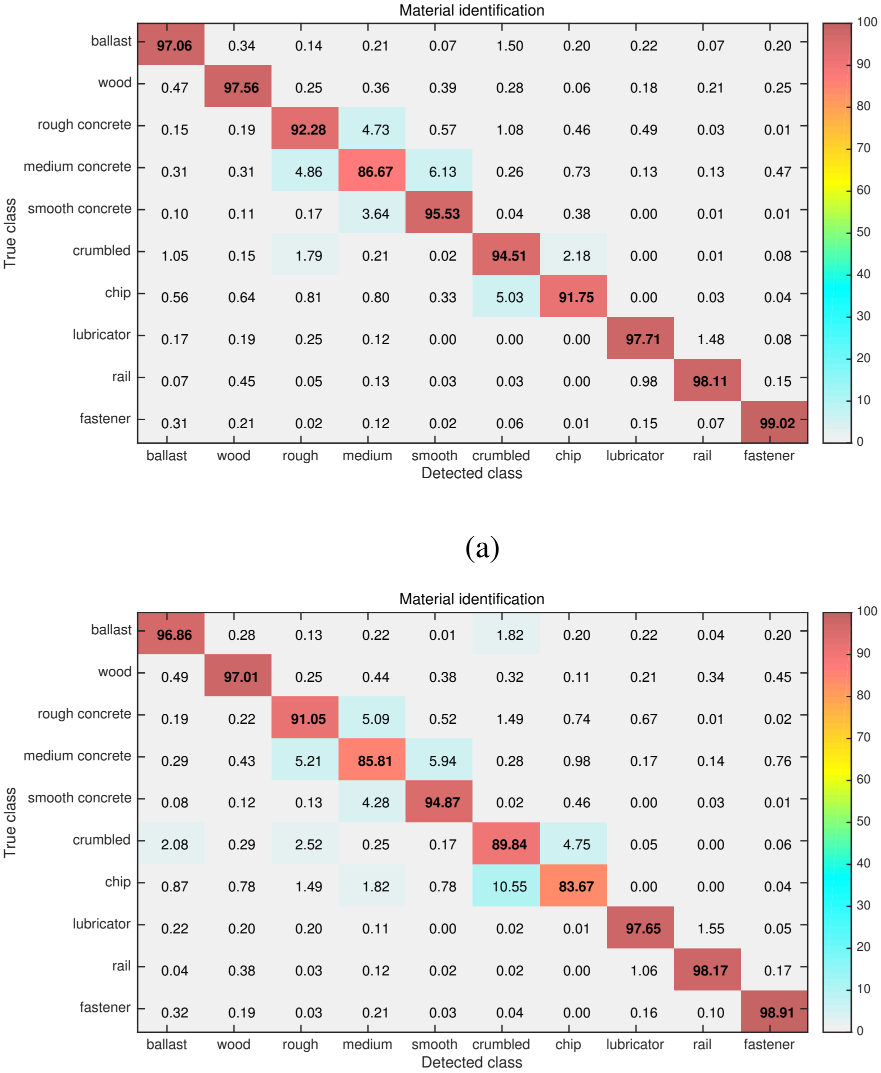
To evaluate the performance of our defect detector, we divided each tie into 4 regions of interest (left field, left gage, right gage, right field) and calculated the score defined by (5.14) for each of them. Figure 3.7 shows the ROC curve for crossvalidation on the training set as well as for the testing set of 813,148 ROIs (203,287 ties). The testing set contains 1,052 ties images with at least one defective fastener per tie. The total number of defective fasteners in the testing set was 1,087 (0.13% of all the fasteners), including 22 completely missing fasteners and 1,065 broken fasteners. The number of ties that we flagged as "uninspectable" is 2,524 (1,093 on switches, 350 on lubricators, 795 covered in ballast, and 286 with other issues).

We used the ROC on clear ties (blue curve) in Figure 3.7 (b) to determine the optimal threshold to achieve a design false alarm rate of 0.07% ($\tau = 0.1070$). This target is a bit lower than the 0.1% that we used in the for the baseline experiments. The reason for lowering the sensitivity is that the detection rate plateaus at PFA $> 0.06\%$. Using this sensitivity level, we ran our defective fastener detector at the tie level (by taking the minimum score across all 4 regions). Results are shown in Table 5.3.

At this sensitivity level, our MTL detector only misses one defect (compared to 17 type II errors with the baseline detector). The false alarm rate on clear ties goes down to 0.25%, which is 34% lower than the baseline. Figure 5.8 shows the single defective fastener that was missed. It could be argued that the clip is still holding the rail in place, so it is a very close call.

Table 5.3: Results for detection of ties with at least one defective fastener.

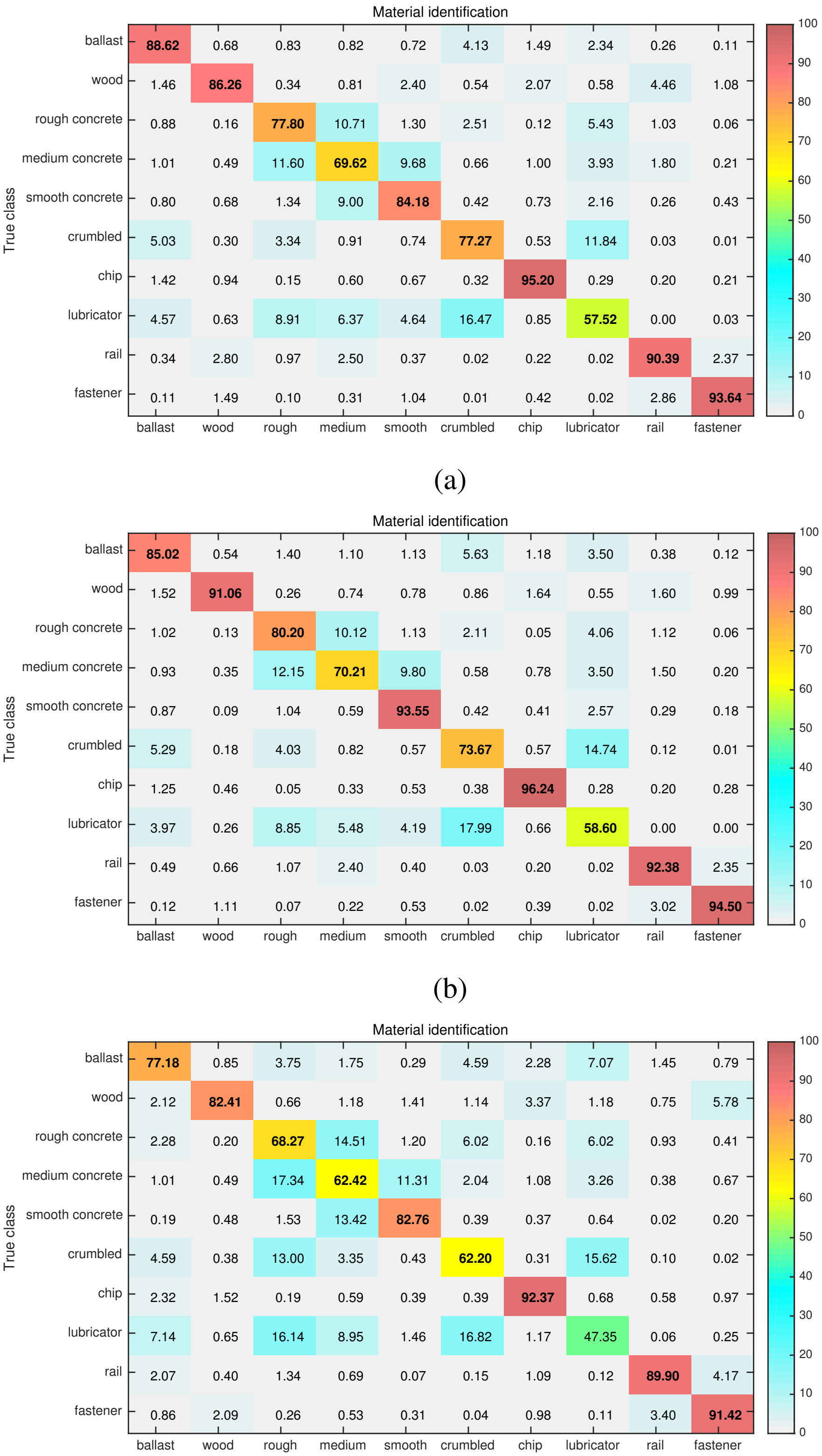
Subset	Total	# Bad	PD		PFA	
			MTL	STL	MTL	STL
clear ties	200,763	1,037	**99.90%**	98.36%	**0.25%**	0.38%
clear + sw.	201,856	1,045	**99.90%**	97.99%	**0.61%**	0.71%
all ties	203,287	1,052	**99.90%**	98.00%	**1.01%**	1.23%



(a)

(b)

Figure 5.3: Confusion matrix of material classification on 2.5 million 80×80 image patches with Deep Convolutional Neural Networks using (a) multi-task learning (b) single task learning [7].

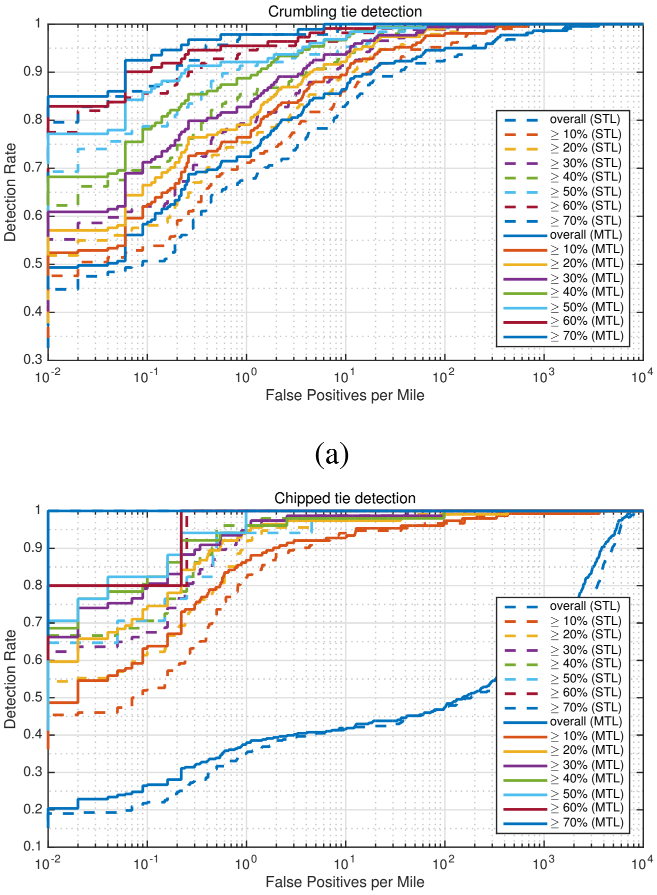


(a)

(b)

(c)

Figure 5.4: Confusion matrix of material classification on 2.5 million 80×80 image patches with (a) LBP-HF with FLANN (b) $\mathrm{LBP}_{8,1}^{u2}$ with FLANN (c) Gabor with FLANN.



(a)

(b)

Figure 5.5: (a) ROC curve for detecting crumbling tie conditions. (b) ROC curve for detecting chip tie conditions. Each curve is generated considering conditions at or above a certain severity level. Note: False positive rates are estimated assuming an average of 10^4 images per mile. Confusion between chipped and crumbling defects are not counted as false positives.

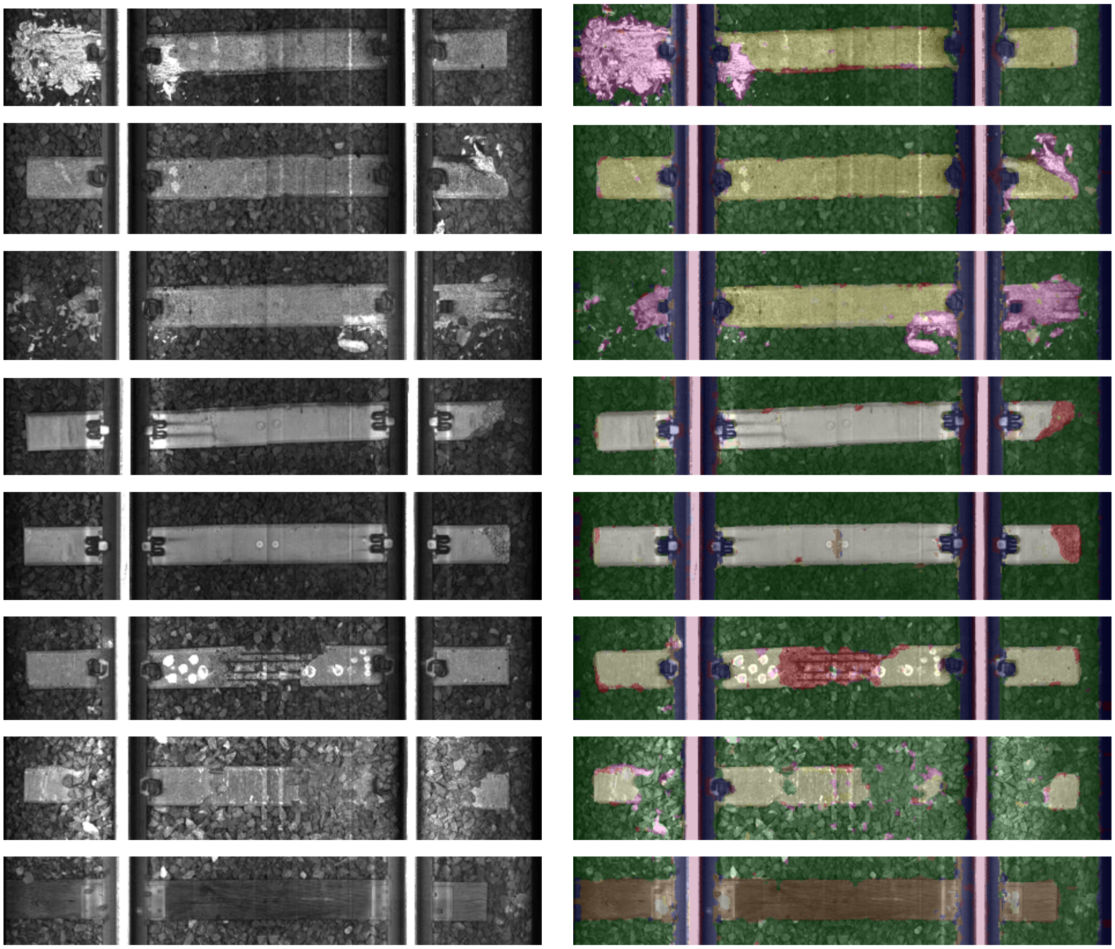
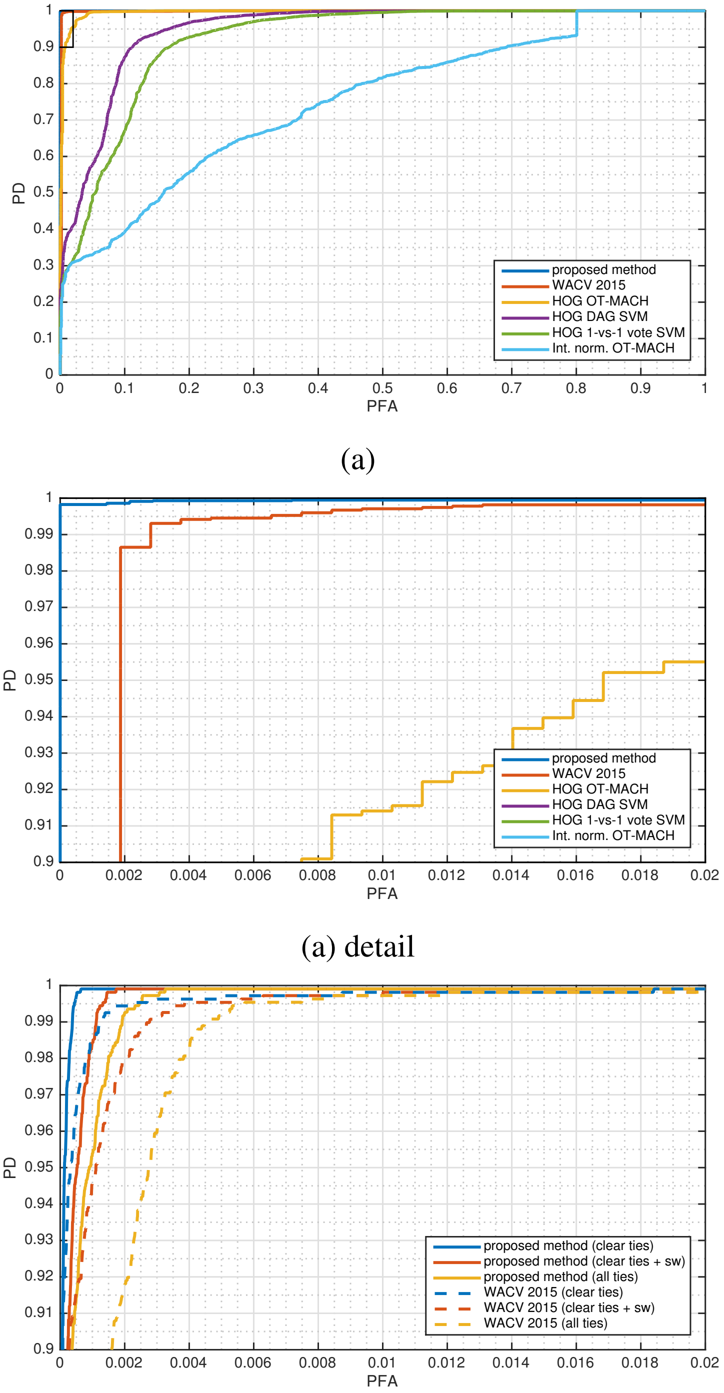


Figure 5.6: Semantic segmentation results (images displayed at 1/16 of original resolution). See Figure 5.2 for color legend.



(a)

(a) detail

(b)

Figure 5.7: ROC curves for the task of detecting defective (missing or broken) fasteners (a) using 5-fold cross-validation on the training set (b) on the 85-mile testing set.

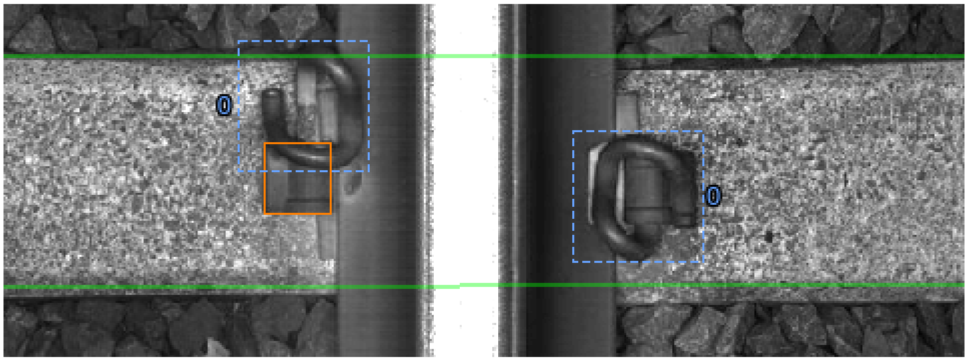


Figure 5.8: The single defect missed by our detector. Solid bounding boxes correspond to ground truth annotations. Dashed bounding boxes correspond to the output of the detector. The number 0 corresponds to the PR-clip class, which is correctly classified. The clip has not completely popped out.

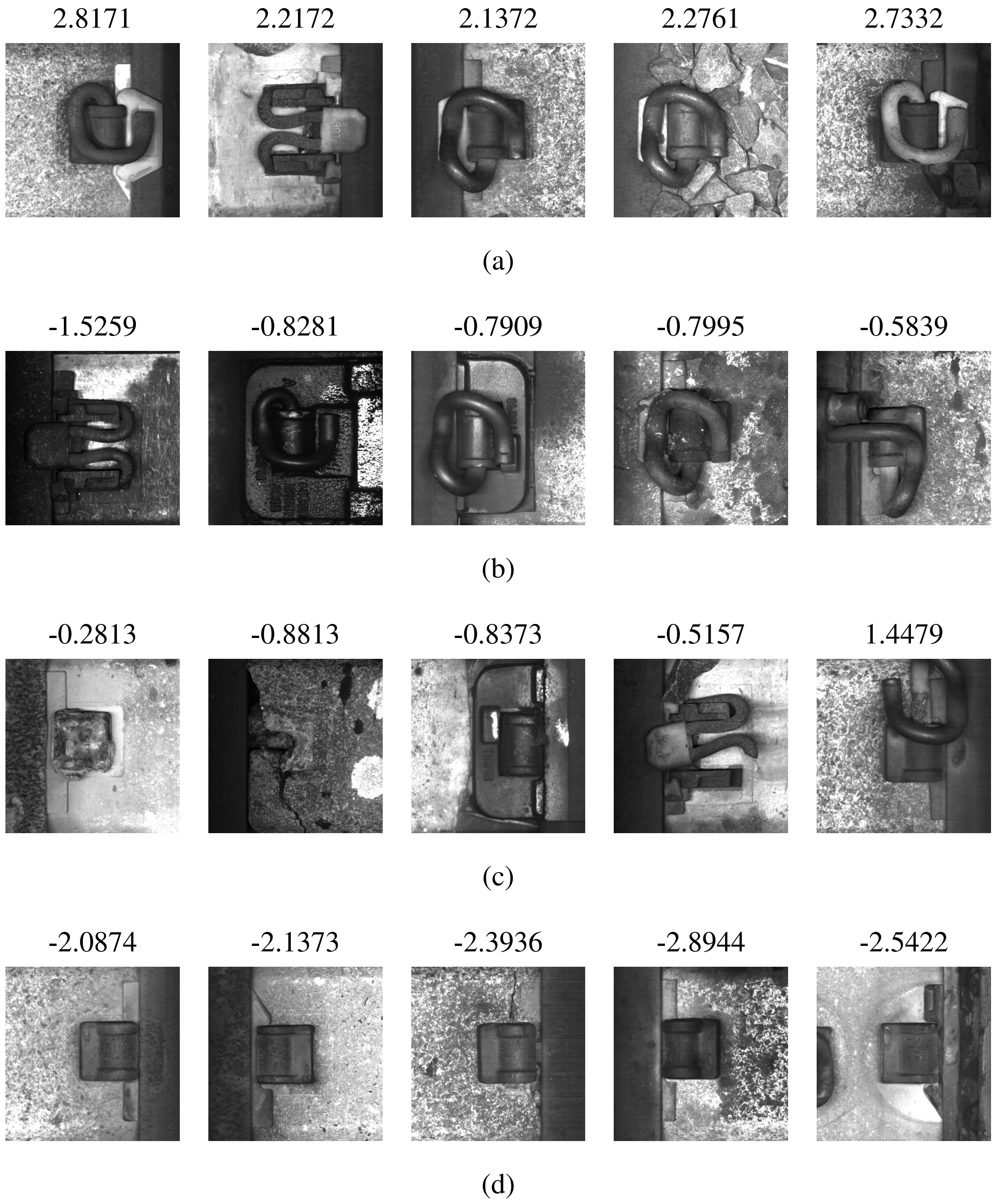
Chapter 6: Sequential Anomaly Detection with Adaptive Thresholding via Extreme Value Theory

In previous chapters, we introduced several techniques that enable anomaly detection in noisy images. These techniques include an iterative shrinkage algorithm for separating normal image components from anomalous ones, as well as a pattern recognition approach for simultaneous detection and categorization of normal and anomalous components. Our approach used so far involved using dictionaries of normal and abnormal patterns for the purpose of scoring the elements of an image according to their likelihood of being anomalous. Dictionaries can be derived from the application of a transformation of the data (as was done in chapter 3), or directly learned from the data (as in chapter 4). In the first case, we used the training data to select the best parameters associated with the transformation, while in the second case, we directly learned the representation. Then in chapter 5 we have gone a step further and we have trained a deeper model using multiple tasks that share the same representation. Both training and testing samples have been assumed to be independent and identically distributed (i.i.d.). In this chapter, we extend these techniques by exploiting the time dependency to make false alarm rate as independent from time as possible.

6.1 Introduction

In sequential inspection problems, such as visual railway track inspection, a video feed is streamed from one or more cameras to a detection system, and we are interested in designing a detector that can find abnormal patterns in such data. There is a limit to the number of false alarms that the operator can handle, so it is necessary to select the optimal operating point at which the false alarm rate does not exceed such limit. Indeed, most of the data that an autonomous inspection vehicle will collect will be discarded without anyone ever looking at it. Therefore, an excessively high false alarm rate will result in a waste of storage space and bandwidth. The only relevant images are the ones that correspond to unexpected patterns, so we are actually interested in finding such anomalous patterns.

Anomaly detection is a hypotheses testing problem in which the null hypothesis is that an image is normal and the alternative hypothesis is that it is anomalous. Due to the complexity of the scene and image formation process, both hypotheses are composite, with nuisance parameters arising from changes in illumination, occlusion, background clutter, and many other uncontrollable factors. Rather than trying to model each of these variables individually, we adapt the detection scores with the objective of reducing the variability in type I error rate. This is known as constant false alarm rate (CFAR) detection. We adopt the Bayesian view that such parameters are random variables with one realization per image. The images have a natural order based on the time they were captured at, so the sequence of these random parameters forms a random process. A key observation is that this random process has strong long-term dependencies. The effect of such slowly varying nuisance parameters is that false alarms are concentrated in small



(a)

(b)

(c)

(d)

Figure 6.1: Examples of fastener scores (a) Good fasteners with high scores (b) Good fasteners with low scores (c) Defective fasteners with high scores (d) Defective fasteners with low scores

segments of the image sequence.

Figure 6.1 shows examples of good and defective fasteners and their detection scores generated by the multi-task learning method in the previous chapter. Although most fasteners have high scores and most defective ones have low scores, when good fasteners have low scores, there is an underlying phenomenon that causes scores of nearby images to also be low.

6.2 Background

6.2.1 Robust Anomaly Detection

The presence of outliers is a challenge that many computer vision systems have to deal with. The RANdom SAmple Consensus (RANSAC) algorithm [95] has been used in many applications for removing outliers when fitting a model to data. This method is especially useful when most of the samples follow a linear model plus additive i.i.d. Gaussian noise, but a few samples are gross errors that do not follow this model. However, in many applications, it not clear which samples should be treated as inliers and which of them are outliers. For instance, in big data applications, the data just appears to have a distribution with long tails that decay at slower rate than the corresponding Gaussian distribution that best fits the data in the least squares sense. Indeed, what appears to be an outlier in feature space may just be a normal sample that has been subject to some kind of degradation for which the feature extractor was not designed for. These degradation modes may include impulse noise, partial occlusion, and in some cases, changes in appearance due to blur, shadows, or pose. In anomaly detection problems, the samples of interest are those in the tail of such data distribution. Therefore, any method that discards outliers have the potential of discarding anomalies, so in order to successfully find anomalies in such images it is necessary to use other methods.

The field of robust statistics [96, 97] provides the tools for estimation of unknown quantities when the underlying probability distribution is non-Gaussian and it is not known exactly. In practice, the data can be modeled as the mixture of a Gaussian distribution and a heavy-tailed distribution (the contaminated Gaussian model). In this case, it is be desirable to design an estimator whose performance is minimax over a family of distributions that includes the Gaussian as a special case. There are basically three types of robust estimates: M-estimates [98] (Maximum likelihood type), L-estimates (Linear combination of order statistics), and R-estimates (Estimates derived from rank tests).

In supervised learning problems, there is a distinction on how to handle outliers at training time vs. testing time. Supervision at training time usually mitigates the problem of outliers as it is possible to manually select the inliers. Moreover, in chapters 3, 4, and 5 we described methods where we optimized a cost function based on the ℓ_1 instead of ℓ_2 norm. In both cases, the use of the ℓ_1 minimization was motivated as a convex relaxation of the ℓ_0 to promote a sparse representation of the data. The solution of the ℓ_1 minimization is the Maximum Likelihood Estimate of the location parameter when the data follows a Laplacian distribution, and a straightforward way of robustifying a regression procedure is by replacing the ℓ_2 norm in the cost function by the ℓ_1 norm. A related L-estimator that results from such ℓ_1 optimization is the Least Median of Squares (LMS), which was introduced in the computer vision field by Kim et al. [99]. The drawback of the LMS is that the median estimator's efficiency is only $\frac{2}{\pi} = 0.637$ when the true distribution is Gaussian.

The M-estimator based on the Huber loss function [98]

$$\rho(t) = \begin{cases} \frac{1}{2}t^2 & \text{for } |t| < k \\ k|t| - \frac{1}{2}k^2 & \text{for } |t| \geq k \end{cases} \tag{6.1}$$

is more flexible because it has the sample mean ($k = \infty$) and sample median ($k = 0$) as special cases and it can be tuned to handle different degrees of contamination in the contaminated Gaussian model. However, since this estimator depends on a scale parameter k (unlike L-estimators, which are scale-invariant), it is necessary to first estimate this parameter it using a robust scale estimator.

6.2.2 Extreme Value Theory for Adaptive Anomaly Detection

Due to illumination and viewpoint changes, clutter distribution, and other image degradation, the distribution of features extracted from images at test time, does not match what was observed during training. Moreover, such a distribution may not be stationary, but slowly changes over time, so a fixed threshold would result in large variability in the false alarm rate. Broadwater and Chellappa [100] proposed a technique to find adaptive thresholds for Constant False Alarm Rate (CFAR) detectors based on Extreme Value Theory (EVT) [101] that can be used even when limited training data is available. EVT is applicable to problems where the probability of a rare event must be estimated even if such a rare event has never occurred. Scheirer et al. [102, 103] also used EVT for score normalization and showed its applicability to sensor fusion problems.

For completeness, we recall the EVT basic results below. Let $X_1, \ldots, X_n$ be i.i.d. samples from an unknown distribution F and $M_n = \max(X_1, \ldots, X_n)$, the maximum of n i.i.d. variables. The fundamental EVT theorem, the Fisher-Tippett-Gnedenko theorem

[101], states that if there exist a sequence of pairs of real numbers (a_n, b_n) such that $a_n > 0$ for all n and a distribution function $\Lambda(x)$ such that

$$\lim_{n\to\infty} P\left(\frac{M_n - b_n}{a_n} \leq x\right) = \Lambda(x), \tag{6.2}$$

for all x at which $\Lambda(x)$ is continuous, then the limit distribution $\Lambda(x)$ belongs to either the Gumbel, the Fréchet or the Weibull family. These three families can be grouped into the Generalized Extreme Value Distribution (GEVD)

$$\Lambda(x;\mu,\sigma,\xi) = \exp\left\{-\left[1+\xi\left(\frac{x-\mu}{\sigma}\right)\right]^{-1/\xi}\right\}, \tag{6.3}$$

where $\mu \in \mathbb{R}$ is the location parameter, $\sigma > 0$ the scale parameter and $\xi \in \mathbb{R}$ the shape parameter. The Gumbel distribution is a special case of the GEVD when $\xi = 0$, the Fréchet when $\xi > 0$, and the Weibull when $\xi < 0$. When the limiting distribution exists, we say that $F(x)$ lies in the "domain of attraction" of $\Lambda(x)$.

In many practical applications, we are interested in the tail distribution of the distribution F. Given an upper threshold u, we select the N_n samples that exceed such threshold and define the excesses $Y_1, \ldots, Y_{N_n}$ as $Y_i = X_j - u$, where i is the excess index and j is the index of the original sample. The probability of exceeding the threshold is $\lambda = 1 - F(u)$. For sufficiently large u, the upper tail distribution function $F_u(y)$ (the conditional distribution function of the excesses),

$$F_u(y) = \frac{F(u+y) - F(u)}{1 - F(u)} \tag{6.4}$$

can be approximated by a Generalized Pareto Distribution

$$G(y;\sigma,\xi) = 1 - \left(1 + \frac{\xi y}{\sigma}\right)_+^{-1/\xi}, \quad y > 0. \tag{6.5}$$

where $\sigma > 0$, $\xi \in \mathbb{R}$, and $x_+ = \max(x, 0)$. This approximation is justified by the Pickands theorem [104], which states that

$$\inf_{\xi} \lim_{u \uparrow \omega_F} \inf_{\sigma} \sup_{y>0} |F_u(y) - G(y; \sigma, \xi)| = 0 \tag{6.6}$$

if and only if F is in the domain of attraction of the GEVD. Note that the exponential distribution is a special case of the GPD for $\xi = 0$, i.e. $G(y; \sigma, 0) = 1 - e^{-y/\sigma}$.

These results can be extended to the multivariate case, for example to model the tail distribution of the maximum of a cluster of observations. Under stationarity of observations, this can be achieved by incorporating both the tail of the marginal distribution and the so-called extremal index. Let $\{X_n : n \geq 1\}$ be a (strictly) stationary sequence of r.v.'s with marginal distribution F. Then, for sufficiently large n

$$P\{M_n \leq u_n\} \approx F^{n\theta}(u_n), \tag{6.7}$$

where u_n is any high threshold such that $n(1 - F(u_n))$ converges to a positive number as $n \to \infty$ and θ is a fixed number in $[0, 1]$. θ is the extremal index that measures the strength of dependence of $\{X_n\}$. If $\{X_n\}$ are independent, then $\theta = 1$. On the other hand, if $\{X_n\}$ are highly dependent, then $\theta \approx 0$. A method for estimating the extremal index for a real-valued Markov chain was proposed by Yun [105].

6.3 Proposed Approach

In this section we describe our approach for normalizing the scores of an anomaly detector deployed in an application in which the distribution of the normal samples gradually changes over time. This may be caused by changes in illumination, change in viewpoint, addition or removal of clutter, or other uncontrollable factors. The approach is similar to the method proposed by Broadwater and Chellappa [100] in which an adaptive threshold is estimated from the GPD fit to the upper tail of the distribution after removing the outliers or targets using a Kolmogorov-Smirnov statistical test. The difference is that our method is Bayesian and we work with sequential data and estimate the adaptive threshold for each sample.

6.3.1 Bayesian Model

We want to adapt the scores of an anomaly detector applied to a sequence of images so that, when we apply a given threshold, we get an approximately constant false alarm rate (a CFAR detector). The images have been collected from a moving vehicle, so the environmental conditions and clutter distribution are not stationary, but slowly change over time. In EVT-based threshold estimation, it is necessary to estimate the parameters σ and ξ of the GPD from the upper- or lower-tail of the empirical distribution. For the rest of this chapter we will refer to the upper tail of the distribution of the random variable X, but the same applies to the lower tail since the lower tail of X is the upper tail of $Z = -X$. The threshold u needs to be set high enough so that the tail of $F(x)$ converges in distribution to the GPD. However, since we are dealing with a non-stationary random process, we need to work on a small window centered at the sample of interest. This window needs to be long enough so that we can fit the parameters of the GPD to its tail (for example the largest 5% of the samples), but short enough that the distribution has not changed much. In applications in which the dynamics of the process change quickly, our options are rather limited. If we fit a GPD to the extreme samples of a short window, the estimated threshold has so much variance that the resulting performance is worse than using a fixed threshold. On the other hand, if the window is too long, the threshold does not adapt at all. For example, if we use a window of 100 samples and select the upper threshold to the 95th percentile, we would only have 5 samples to estimate the 2 parameters of the GPD, resulting in severe overfitting.

To overcome this limitation, we will make one simplification by fixing $\xi = 0$, so we only need to estimate one parameter instead of two. Under $\xi = 0$, the GPD reduces to the Exponential distribution

$$g(y; \sigma, \xi = 0) = e^{-y/\sigma} \tag{6.8}$$

For convenience, we apply the parameterization $\lambda = 1/\sigma$ and write the Exponential in its canonical form

$$g(y; \lambda) = e^{-\lambda y} \tag{6.9}$$

As opposed to the general case of the GPD, the Exponential distribution is a member of the exponential family, so it has a non-trivial sufficient statistic from which we can easily compute the MLE of its parameter. Its conjugate prior is the Gamma distribution,

$$\pi(\lambda; \alpha, \beta) = \frac{\beta^\alpha}{\Gamma(\alpha)} \lambda^{\alpha-1} e^{-\beta\lambda}, \tag{6.10}$$

the non-informative (improper) prior is given by $\alpha = 1$, $\beta = 0$, and the parameters of the Gamma posterior under a Gamma$(\lambda; \alpha_0, \beta_0)$ prior can be computed as

$$\alpha_1 = \alpha_0 + n \tag{6.11}$$

$$\beta_1 = \beta_0 + \sum_{i=1}^{n} y_i \tag{6.12}$$

This simplified model allows us to derive a very fast adaptation algorithm that we describe in the following section. We believe that this approximation is good enough in practice, specially when the scores are trained with a sparsity promoting loss function such as the hinge loss described in chapter 5.

6.3.2 Training

Our training set $\mathcal{T}$ contains a number of sequences of scores $\mathbf{x}$ with their corresponding sequences of labels $\mathbf{y}$. During training, we compute the sufficient statistics for all the samples that are not labeled as anomalies and re-scale them based on our belief that at test time the tail distribution will be close to that in the average training sequence. The steps of the training procedure are described in Algorithm 1. The parameter p_u is the probability of the tail, and w_0 is the weight in sample counts given to the training set. In our experiments we used $p_u = 0.05$ and $w_0 = 400$.

6.3.3 Proposed Adaptive Thresholding Algorithm

During testing, we first perform a series of Kolmogorov-Smirnov tests to find and remove anomalies. Then, using the prior estimated during training, we compute the posterior for the whole sequence. This posterior is used as the prior for estimating the tail distribution on each shift of a window centered on each of the samples. The details of the adaptation procedure are described in Algorithm 2. The input to the adaptation procedure is a sequence of scores $\mathbf{x}$, the parameters of the prior Gamma distribution α_0 and β_0, the size of the upper tail p_u, the target false alarm rate p_f, the weight w_1 given to the the prior

Algorithm 1 EVT training algorithm

```
procedure TRAIN($\mathcal{T}$, $p_u$, $w_0$)
    $n \leftarrow 0$, $s \leftarrow 0$                          ▷ Initialize sufficient statistics
    for all $(\mathbf{x}, \mathbf{y}) \in \mathcal{T}$ do          ▷ Training set $\mathcal{T}$ contains $\mathbf{x}$ scores, $\mathbf{y}$ labels
        $\mathbf{g} \leftarrow \{x_i \mid y_i = 0\}$                ▷ Select negative samples
        $u \leftarrow u \mid \#\{g_i > u\} = \#\mathbf{g}\, p_u$     ▷ Find upper threshold
        $\mathbf{t} \leftarrow \{g_i \mid g_i > u\}$ - u             ▷ Extract upper tail
        $n \leftarrow n + \#\mathbf{t}$                           ▷ Update counts
        $s \leftarrow s + \sum \mathbf{t}$                        ▷ Update sum
    end for
    $\alpha_0 \leftarrow 1 + s$
    $\beta_0 \leftarrow \frac{w_0\, s}{n}$
    return $\alpha_0$, $\beta_0$                              ▷ Parameters of the Gamma prior
end procedure
```

contribution of the whole sequence, the window length L, and the maximum number of anomalies n_a in the sequence. The output sequence $\mathbf{y}$ has been adapted so that when it is thresholded at 0, the false alarm rate is p_f. For our experiments we have used $p_u = 0.05$, $p_f = 0.001$, $w_1 = 100$, $L = 101$, and $n_a = 12$.

6.4 Experimental Results

To validate the effectiveness of the proposed approach, we have used the 340 sequences of fastener detections corresponding to each of the 4 cameras in each of the 85 miles of the Amtrak NEC concrete tie dataset introduced in chapter 4. This dataset contains a total of 203,287 ties and each tie is divided into 4 regions (left field, left gage, right gage, and right field), so the total number of images is 813,148. The detection problem

Algorithm 2 EVT adaptive thresholding algorithm

```
procedure AdaptScores($\mathbf{x}$, $\alpha_0$, $\beta_0$, $p_u$, $p_f$, $w_1$, $L$, $n_a$)
    $\widehat{a}_0 \leftarrow \frac{\beta_0}{\alpha_0 - 1}$                          ▷ MLE in training set
    $\mathbf{y} \leftarrow \mathbf{sort\_desc}(\mathbf{x})$                          ▷ Sort scores in descending order
    $k \leftarrow \#\mathbf{y}\ p_u$
    for $i \leftarrow 1, n_a$ do                      ▷ Training set $\mathcal{T}$ contains $\mathbf{x}$ scores, $\mathbf{y}$ labels
        $u \leftarrow y_{i+k}$                          ▷ Find upper threshold
        $\mathbf{t} \leftarrow \{y_i, \dots, y_{i+k}\} - u$                          ▷ Extract upper tail
        $D_{n,i} = \sup_x \left|\widehat{G}_n(x) - G(x)\right|$                          ▷ Compute KS statistic
    end for
    $\hat{i} \leftarrow \min_i\{D_{n,i}\}$                          ▷ Estimate number of outliers
    $u' \leftarrow y_{\hat{i}}$                          ▷ Set outlier rejection threshold
    $\mathbf{t} \leftarrow \{y_{\hat{i}}, \dots, y_{\hat{i}+k}\} - u$                          ▷ Extract upper tail
    $\alpha_1 \leftarrow \alpha_0 + \sum \mathbf{t}$
    $\beta_1 \leftarrow \beta_0 + \frac{w_1 \sum \mathbf{t}}{\#\mathbf{t}}$
    for $i \leftarrow 1, n$ do
        $\mathbf{w} \leftarrow \mathbf{x}_{i-(L-1)/2:i+(L-1)/2}$                          ▷ Window centered at sample $x_i$
        $u \leftarrow u \mid \#\{w_i > u\} = \#\mathbf{w}\ p_u$                          ▷ Find upper threshold
        $\mathbf{t} \leftarrow \{w_i \mid w_i > u\}$ - u                          ▷ Extract upper tail
        $\alpha \leftarrow \alpha_1 + \#\mathbf{t}$                          ▷ Posterior
        $\beta \leftarrow \beta_1 + \sum \mathbf{t}$                          ▷ Posterior
        $\widehat{a} \leftarrow \frac{\beta}{\alpha - 1}$                          ▷ MAP estimate
        $y_i \leftarrow x_i + u - \widehat{a}\ log(p_f/p_u)$                          ▷ Adapt score
    end for
    return $\mathbf{y}$                          ▷ Adapted scores
end procedure
```

consists of determining whether an image contains a fastener attached to one of the rails. The dataset contains bounding boxes for all the images that are known to contain a defect. The total number of defects is 1,087 (0.13% of all the fasteners). The defective fastener class contains two subclasses: broken fastener and missing fastener.

We have used the scores generated by the multi-task learning (MTL) detector described in chapter 5. This detector uses deep learning with multiple tasks that are trained in parallel. The reason for using multiple tasks is to prevent overfitting. By sharing a common low-level representation between the fastener inspection task and a separate material classification task, there is a data augmentation effect that results in better generalization for both classifiers. We also compare the performance with the baseline single-task learning (STL) method in chapter 4. This detector produces a scalar-valued score for each image by spatially pooling all the detections in the image. Scores are high when the image contains a good fastener, and low when the fastener is either missing or broken. Figure 6.1 shows several detection examples of the MTL detector.

To facilitate the evaluation of fastener detection performance under difficult scenarios, whenever the fastener is not directly attached to the rail or tie, or when for some reason a fastener is not visible at all, those ties are marked as uninspectable with a special label. Depending on the value of such label, the dataset is divided into 3 subsets:

- *Clear ties:* 200,763 ties (1,037 ties with at least one defect).
- *Clear ties plus switches:* 201,856 ties (1,045 ties with at least one defect). See Figure 6.2 for an example of a switch section.
- *All ties:* 203,287 ties (1,052 ties with at least one defect). This includes switches,

and ties for which some fasteners are not visible because they are covered by ballast or a lubricator. See Figures 6.3 and 6.4 for examples of high ballast and lubricator sections.

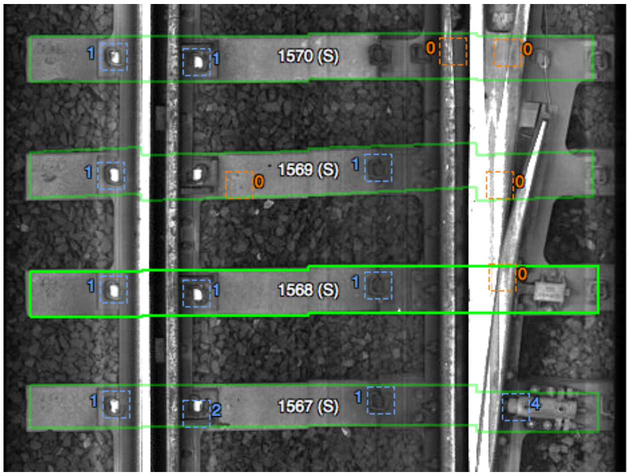


Figure 6.2: Example of section marked as switch.

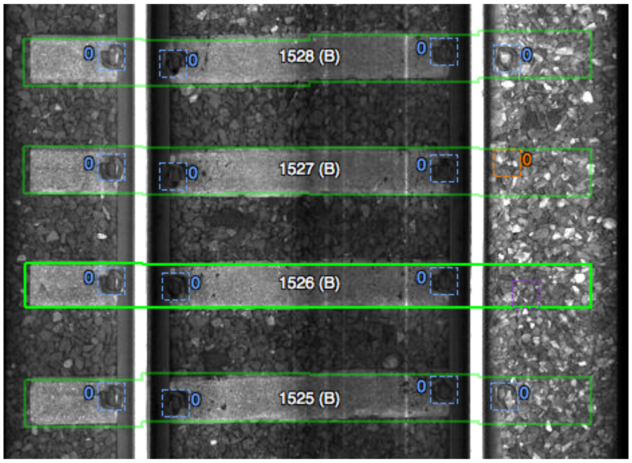


Figure 6.3: Example of section marked as ballast.

For training, we use all the available data after setting aside the sequence being tested. Table 6.1 and Figure 6.5 show the detection results on the normalized scores. The overall improvement is significant. The detection rate on the whole dataset at $PFA =$

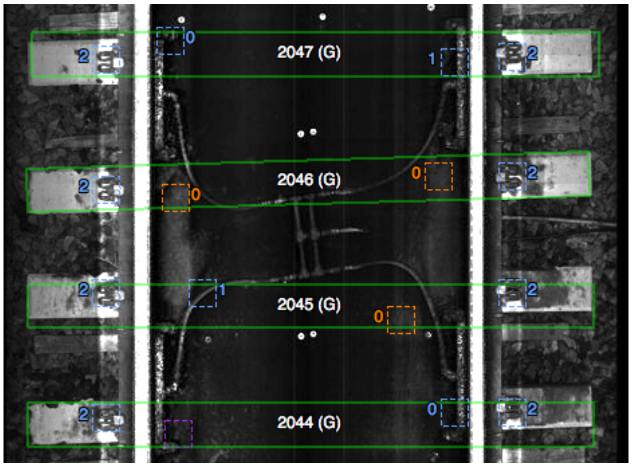
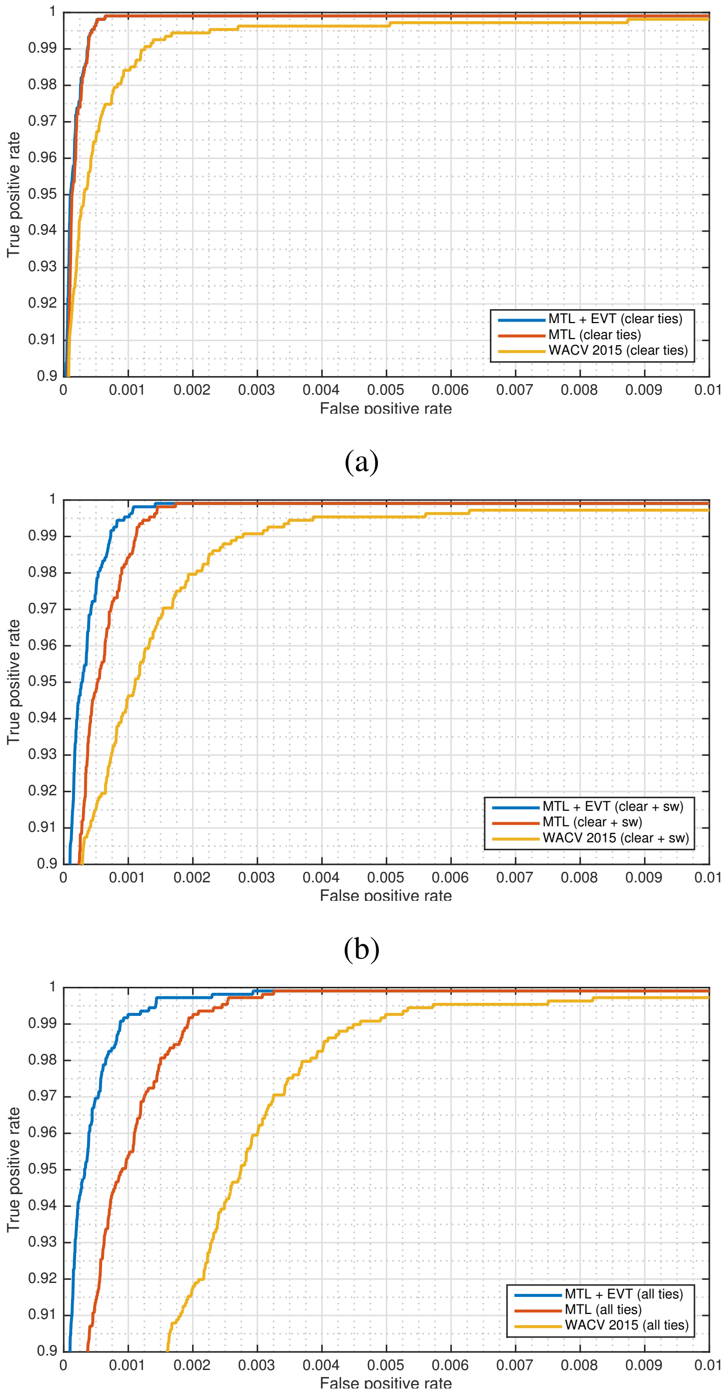


Figure 6.4: Example of section marked as lubricator.

0.1% increases from 95.40% to 99.26%. This is a $6\times$ reduction in the missed rate. Moreover, the performance on the clear tie subset does not degrade at all. The running time of our EVT adaptation algorithm implemented in MATLAB for adapting all 813,148 scores is only of 17 seconds on a Mid-2012 MacBook Pro with a 2.5 GHz Intel Core i5 processor, so this dramatic improvement comes at negligible computational cost (running the detector process takes several hours).

Condition	PFA	MTL + EVT	MTL [106]	STL [6]
Fastener (only clear ties)	0.1%	**99.91%**	**99.91%**	98.41%
	0.02%	**97.20%**	96.74%	93.19%
Fastener (clear + switch)	0.1%	**99.54%**	98.43%	94.54%
	0.02%	**93.80%**	89.35%	88.70%
Fastener (all ties)	0.1%	**99.26%**	95.40%	87.38%
	0.02%	**93.47%**	87.76%	–

Table 6.1: Fastener detection results before and after score normalization.



(a)

(b)

(c)

Figure 6.5: ROC curves comparing defective fastener detection performance on the 85-mile testing set using normalized vs. unnormalized scores (a) on the clear ties subset (b) on the clear with with switches subset (c) on all ties. Detections are per image (each tie has 4 images).

Chapter 7: Conclusions and Future Work

7.1 Summary

In previous chapters, we introduced several techniques that enable anomaly detection in noisy images. These techniques include an iterative shrinkage algorithm for separating normal image components from anomalous ones, as well as a pattern recognition approach for simultaneous detection and categorization of normal and anomalous components. The approaches that we described involved using dictionaries of normal and abnormal patterns for the purpose of scoring the elements of an image according to their likelihood of being anomalous. Dictionaries can be derived from the application of a transformation of the data (as was done in chapter 3), or directly learned from the data (as in chapter 4). In the first case, we used the training data to select the best parameters associated with the transformation, while in the second case, we directly learned the representation. Then, in chapter 5 we have explored the use of deep learning for simultaneous analysis of textures and anomaly detection. We have shown that it is possible to use partially labeled data by using a shared representation and adjusting the objective function to the available data. A possible limitation of this technique is that training and testing samples have been assumed to be independent and identically distributed (i.i.d.) with the same distribution. This caused bursts of false alarms due to the intermittent presence of clutter in the images. To mitigate this problem, in chapter 6 we have proposed an adaptive thresholding algorithm for sequential anomaly detection using results from extreme value theory. This has resulted in more robust anomaly detection results.

7.2 Future Work

In this section we discuss possible extensions, generalizations and improvements to each of the algorithm presented in previous chapters.

1. **Discrete Shearlet Transform on GPU with Applications in Anomaly Detection and Denoising:**

   From the computational point of view, the following could be useful extensions to our GPU implementation:

   - Port to OpenCL, so it can run on other devices.
   - Support non-square (rectangular) images.
   - Compute filter coefficients in the GPU.

   From the algorithmic point of view, it is well known that with large amounts of training data, dictionaries learned from data tend to perform better than predefined filter banks. Although the shearlet transform described in chapter 3 has shown good performance on crack detection and denoising applications, this is not the only tool available for such tasks. The algorithm that we used for denoising consists of a linear transformation (the direct DST) followed by a non-linearity (shrinkage) followed by another linear transformation (the inverse DST). The algorithm for decomposing images into morphologically distinct components, such as anisotropic textured and directional edges, follows a similar (but deeper) structure. Moreover, these linear transformations are convolutional filters. Indeed, this algorithms have a DAG structure of convolutional layers and non linearities, and could be mapped into a DCNN like those used in Chapter 5. In future work, it would be interesting to train DCNN for performing crack detection and denoising and study whether the filters learned for these tasks share any of the properties of the shearlet transform. Also, it would be interesting to introduce regularization terms in the loss function of DCNNs to promote shearlet-like properties to the learned filters, namely localization in space and frequency domains and maximally flat frequency response.

2. **Robust Fastener Detection for Autonomous Visual Railway Track Inspection:**

   The algorithm introduced in chapter 4 is an example of how to leverage large amounts of training data for mining representative training data for anomaly detection. A possible extension of this method would be to introduce better invariance to rotation and deformation. A brute-force approach would be to synthetically generate training samples using data augmentation techniques. However, using data augmentation introduces bias in the training set. A better alternative would be to use deformable parts models [107]. However, as shown in chapter 5, by using DCNNs and multi-task learning (i.e, sharing the part detectors with other tasks), we already get a significant performance improvement.

3. **Deep Multi-task Learning for Railway Track Inspection:**

   In chapter 5 we solved two problems with a single pass on the data: material classification and object detection. The parameters of the network have been carefully tuned to balance the relative performance of both problems. Future performance improvement will come as the result of having even large datasets. Having more data will introduce other challenges that will need to be addressed, such as in finding representative training samples and training from weakly labeled data [85]. Furthermore, once the inspection system is fielded and used under conditions not seen during training, it will be necessary to adapt existing model to new domains.

4. **Sequential Anomaly Detection with Adaptive Thresholding via Extreme Value Theory:**

   In chapter 6, we presented a new algorithm that normalizes scores from a sequential anomaly detector with the objective of harmonizing its false alarm rate. Extreme value theory provides a solid foundation from which adaptive thresholding algorithms can be derived. When working with sequences of images, we need to take advantage of the statistical dependencies of nuisance parameters of nearby images. If we discard such dependencies and treat each image in the sequence independently, the performance suffers.

   The CFAR detection approach proposed has applicability beyond railway track inspection from a moving vehicle. It could be used, for example, in surveillance video to remove bursts of false alarms caused by sun glare, insects, rain or fog. Its computational cost is negligible compared to that of the underlying detector, so this approach can be easily retrofitted to existing detectors already in operation.

7.3 Conclusion

Anomaly detection can be formulated in many different ways. There are many different ways of posing such problems, ranging from direct binary classification between good and anomalous, to full-scale image segmentation. In this dissertation, we have explored the problem of finding anomalies on noisy images in visual railway track inspection applications. The techniques that we have described are not limited to such domains, and can be extended to many other applications. For example, in Appendix A we will introduce the problem of finding anomalous tissue on cardiac functional nuclear medical images. This research is still in an exploratory phase, and we still have not collected enough data to generate conclusive results. However, the techniques described in this dissertation could be applied to such problem. The biggest challenge in solving any detection problem is coming up with an objective description of what we are trying to accomplish. Once the objective function is defined, the solution can be reached by breaking the problem into smaller pieces that are easier to solve.

Appendix A: Point-specific Matching of Cardiac Electrophysiological Voltage and SPECT Perfusion Measurements for Myocardial Tissue Characterization

A.1 Introduction

Patients with implantable cardioverter defibrillators (ICDs) can experience shocks in response to ventricular tachycardia (VT). VT is usually caused by electrical conduction pathways within scar tissue that can maintain arrhythmias. Up to 90% of patients experiencing hemodynamically unstable VT require radiofrequency (RF) ablation to isolate regions of slow conducting channels [108–110]. About 80% of these procedures rely on electrophysiological (EP) voltage mapping of the endocardial (or epicardial) surface with a catheter-based system to identify scar areas prior to ablation. Voltage sampling is non-uniform with $\sim$200-400 points per patient. About 17,000 patients in the U.S. have ablation procedures annually, lasting $\sim$4-6 hours with mortality of 3% [111]. At 6-month follow-up, 42% of patients have recurrent incessant or intermittent VT [111], indicating the need and potential to improve the VT ablation procedure.

One key to improved VT ablation is a better pre-procedural predictive map of the scar (bipolar voltage $>$ 0.5 mV) and border zone (0.5-1.5 mV) regions. The previous work of Dickfeld et al. [112, 113] investigated the qualitative use of PET and SPECT to localize myocardial scar, including integrating a derived scar map into a commercial mapping system. They recently showed that Tl-201 SPECT uptake can differentiate between normal and abnormal EP tissue categories using a 68 segment heart model and homogeneous regions [114]. Others have also investigated the relation of EP voltages and PET data [115, 116].

These previous efforts have used averages over cardiac regions. For the development of locally accurate prediction models, it is important to use point-by-point comparison of EP voltage values and PET/SPECT amplitudes, and accurate data registration is essential. In this work, we develop a novel software tool, CardioViewer, that integrates cardiac EP values and PET/SPECT cardiac data, allows interactive adjustment of image registration, and outputs spatially matched EP and SPECT/PET data for further analysis.

A.2 Methods

A.2.1 Input Datasets

Cardiac datasets are from clinical electrophysiology mapping systems and nuclear medicine (PET/SPECT) imaging devices. The EP voltage measurements are from the CartoMerge 3D mapping system (Biosense Webster). EP data are exported to a file and each data point contains the point index, bipolar voltage and 3-D coordinates in the EP mapping system reference frame. The PET/SPECT datasets are from short axis cardiac images in DICOM or other format. Polar maps are derived from the PET/SPECT images using PMOD (Adliswil, Switzerland). Peak transmural intensities are determined at 10 degree angular increments in 20 slices from apex to base. These polar plot values are exported to a file.

Control points for initial registration are chosen by a trained electrophysiologist identifying EP points at the apex and at 90 increments in short axis view within the CartoMerge viewing system. That is, EP points at the 0, 90, 180 and 270 degree locations (lateral, inferior, septal, anterior walls) are identified.

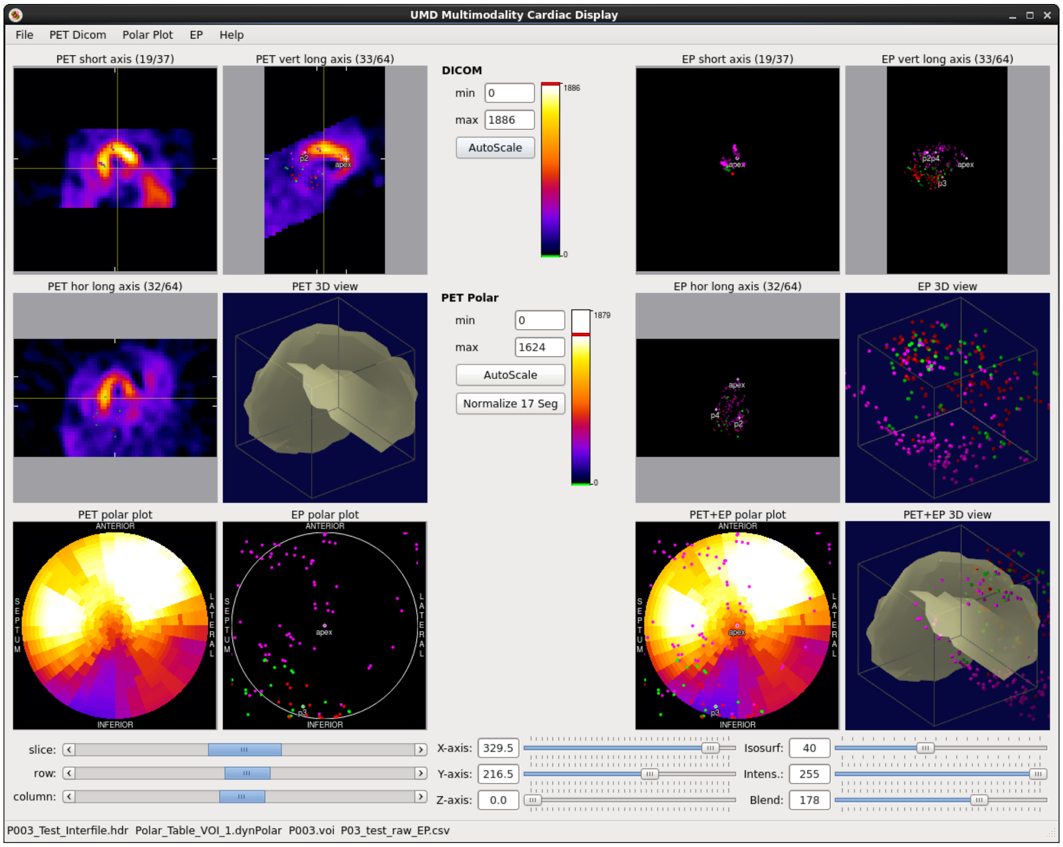
A.2.2 Software Development

The software development environment is C++ using the Qt framework, Qt Creator IDE, Open GL, ITK, freeglut and dicomlib. Orthogonal slices through PET/SPECT datasets and projections of EP points are shown in the upper part of the main screen, and EP and PET/SPECT polar plots are at the bottom. EP points can be overlaid on the SPECT/PET data.

A.3 Results

CardioViewer allows GUI menu-driven input and display of EP and PET/SPECT data as well as their fusion in orthogonal slice, perspective and polar map views. An example of the main GUI interface is shown in Figure A.1 for a rest Tl-201 SPECT study and EP voltage points.

The program permits interactive parameter adjustment, including scrolling through short axis cardiac slices and toggling on and off EP point overlay on cardiac orthogonal views and polar plots. The six-degree of freedom registration parameters can be



DICOM min 0 max 1886

PET Polar min 0 max 1624

X-axis: 329.5
Y-axis: 216.5
Z-axis: 0.0
Isosurf: 40
Intens.: 255
Blend: 178

P003_Test_Interfile.hdr Polar_Table_VOI_1.dynPolar P003.voi P03_test_raw_EP.csv

Figure A.1: Main GUI of the CardioViewer program showing SPECT and EP datasets and their integration. For EP points, scar is red (<0.5 mV), border zone is green (0.5-1.5 mV), normal is purple (>1.5 mV).

interactively adjusted or input through dialog boxes from pull-down menus. The program outputs EP voltage and PET/SPECT values at the same points on polar maps, a key to studying multimodal cardiac tissue attributes that avoids averages over regions. CardioViewer is multiplatform and runs on Linux, Windows and Apple OS.

As an example of the program's capabilities, it can compute goodness of fit metrics between the EP and PET/SPECT data as registration parameters are varied. For the dataset of Figure 1, the best visual registration was with a 60° rotation about the left ventricle axis. This yielded an area under the ROC curve (AUC) of 0.93 for prediction of EP tissue as abnormal (<1.5 mV) from normalized SPECT values. An automated search yielded a peak AUC of 0.95 at an 88° rotation and provides insight into AUC dependence on angle (Figure A.2).

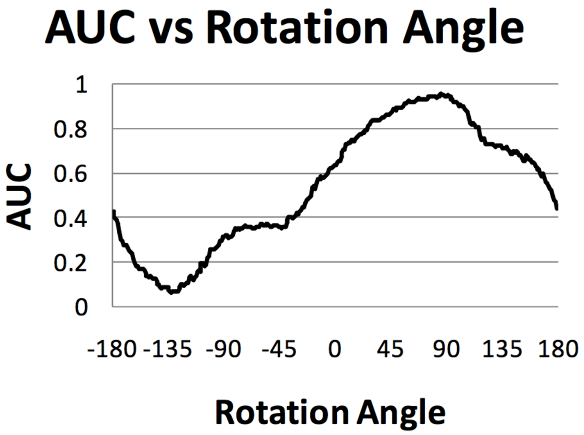


Figure A.2: Sensitivity of AUC vs. rotation angle about the left ventricle axis for prediction of abnormal EP tissue category from SPECT images (dataset of Fig. A.1)

A.4 Discussion

The CardioViewer program provides an easy to use tool for integration of EP and PET/SPECT data. Due to the limited spatial resolution of PET/SPECT images and partial volume effects, it is important to use registered datasets in the polar plot reference frame for generation of paired EP and PET/SPECT values. CardioViewer can be enhanced to integrate EP data with other imaging modalities (e.g. CT, MRI) for multimodal exploration of cardiac tissue properties.

A.5 Conclusion

A novel multiplatform software tool, CardioViewer, has been developed that enables integration of EP voltage data with PET/SPECT perfusion and viability data. It is being used to generate registered datasets to explore multimodal cardiac tissue properties, with the goal of developing pre-procedural predictive maps of cardiac scar and border zone to aid ablation procedures for ventricular tachycardia.

Bibliography

[1] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing Surveys*, 41(3), 2009.

[2] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 28:594–611, 2006.

[3] Hugo Larochelle, Dumitru Erhan, and Yoshua Bengio. Zero-data learning of new tasks. In *AAAI*, volume 1(2), 2008.

[4] K. Chodnicki, X. Gibert-Serra, J. Tian, F. Arrate, R. Chellappa, T. Dickfeld, V. Dilsizian, and M. Smith. Point-specific matching of cardiac electrophysiological voltage and spect perfusion measurements for myocardial tissue characterization. *J Nucl Med*, 55 (suppl 1):602, 2014.

[5] X. Gibert, V. M. Patel, D. Labate, and R. Chellappa. Discrete shearlet transform on GPU with applications in anomaly detection and denoising. *EURASIP Journal on Advances in Signal Processing*, 2014(64):1–14, May 2014.

[6] X. Gibert, V. M. Patel, and R. Chellappa. Robust fastener detection for autonomous visual railway track inspection. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2015.

[7] X. Gibert, V. M. Patel, and R. Chellappa. Material classification and semantic segmentation of railway track images with deep convolutional neural networks. In *IEEE International Conference on Image Processing (ICIP)*, 2015.

[8] X. Gibert, V. M. Patel, and R. Chellappa. Deep multi-task learning for railway track inspection. *IEEE Transactions on Intelligent Transportation Systems*, submitted 10/2015.

[9] X. Gibert, V. M. Patel, and R. Chellappa. Sequential score adaptation with extreme value theory for robust railway track inspection. In *IEEE-ICCV Workshop on Computer Vision for Road Scene Understanding and Autonomous Driving (CVRSUAD)*, 2015.

[10] Michèle Basseville and Igor V. Nikiforov. *Detection of Abrupt Changes: Theory and Application*. Prentice-Hall, Englewood Cliffs, New Jersey, April 1993.

[11] H. Vincent Poor and Olympia Hadjiliadis. *Quickest Detection*. Cambridge University Press, New York, 2009.

[12] Wei Xu, Zhenmin Tang, Jun Zhou, and Jundi Ding. Pavement crack detection based on saliency and statistical features. In *Image Processing (ICIP), 2013 20th IEEE International Conference on*, pages 4093–4097, Sept 2013.

[13] Joseph A. Smak. Evolution of amtrak's concrete crosstie and fastening system program. In *International Concrete Crosstie and Fastening System Symposium*, June 2012.

[14] M. H. Shehata and M. D. Thomas. The effect of fly ash composition on the expansion of concrete due to alkali-silica reaction. *Cement and Concrete Research*, 30:1063–1072, 2000.

[15] S. Sahu and N. Thaulow. Delayed ettringite formation in swedish concrete railroad ties. *Cement and Concrete Research*, 34:1675–1681, 2004.

[16] J.J. Cunningham, A.E. Shaw, and M. Trosino. Automated track inspection vehicle and method, May 2000. US Patent 6,064,428.

[17] M. Trosino, J.J. Cunningham, and A.E. Shaw. Automated track inspection vehicle and method, Mar 2002. US Patent 6,356,299.

[18] F. Marino, A. Distante, P.L. Mazzeo, and E. Stella. A real-time visual inspection system for railway maintenance: Automatic hexagonal-headed bolts detection. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 37(3):418–428, 2007.

[19] Pasquale De Ruvo, Arcangelo Distante, Ettore Stella, and Francescomaria Marino. A GPU-based vision system for real time detection of fastening elements in railway inspection. In *Image Processing (ICIP), 2009 16th IEEE International Conference on*, pages 2333–2336. IEEE, 2009.

[20] X. Gibert, A. Berry, C. Diaz, W. Jordan, B. Nejikovsky, and A. Tajaddini. A machine vision system for automated joint bar inspection from a moving rail vehicle. In *ASME/IEEE Joint Rail Conference & Internal Combustion Engine Spring Technical Conference*, 2007.

[21] A. Berry, B. Nejikovsky, X. Gibert, and A. Tajaddini. High speed video inspection of joint bars using advanced image collection and processing techniques. In *Proc. of World Congress on Railway Research*, 2008.

[22] P. Babenko. *Visual inspection of railroad tracks*. PhD thesis, University of Central Florida, 2009.

[23] Abhijit Mahalanobis, B. V. K. Vijaya Kumar, Sewoong Song, S. R. F. Sims, and J. F. Epperson. Unconstrained correlation filters. *Appl. Opt.*, 33(17):3751–3759, Jun 1994.

[24] E. Resendiz, J.M. Hart, and N. Ahuja. Automated visual inspection of railroad tracks. *Intelligent Transportation Systems, IEEE Transactions on*, 14(2):751–760, Jun 2013.

[25] Y. Li, H. Trinh, N. Haas, C. Otto, and S. Pankanti. Rail component detection, optimization, and assessment for automatic rail track inspection. *Intelligent Transportation Systems, IEEE Transactions on*, 15(2):760–770, April 2014.

[26] Hoang Trinh, Norman Haas, Ying Li, Charles Otto, and Sharath Pankanti. Enhanced rail component detection and consolidation for rail track inspection. In *IEEE Workshop on Applications of Computer Vision (WACV)*, pages 289–295, 2012.

[27] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition (CVPR), 2001 IEEE Computer Society Conference on*, volume 1, pages I–511–I–518 vol.1, 2001.

[28] E. Stella, P. Mazzeo, M. Nitti, C. Cicirelli, A. Distante, and T. D'Orazio. Visual recognition of missing fastening elements for railroad maintenance. In *Intelligent Transportation Systems, 2002. Proceedings. The IEEE 5th International Conference on*, pages 94–99, 2002.

[29] Francescomaria Marino, Arcangelo Distante, Pier Luigi Mazzeo, and Ettore Stella. A real-time visual inspection system for railway maintenance: automatic hexagonal-headed bolts detection. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 37(3):418–428, 2007.

[30] M. Singh, S. Singh, J. Jaiswal, and J. Hempshall. Autonomous rail track inspection using vision based system. In *Computational Intelligence for Homeland Security and Personal Safety, Proceedings of the 2006 IEEE International Conference on*, pages 56–59, Oct 2006.

[31] Hsiang-Yu Hsieh, Nanming Chen, and Ching-Lung Liao. Visual recognition system of elastic rail clips for mass rapid transit systems. In *ASME/IEEE Joint Rail Conference & Internal Combustion Engine Spring Technical Conference*, pages 319–325, 2007.

[32] Y. Xia, F. Xie, and Z. Jiang. Broken railway fastener detection based on adaboost algorithm. In *IEEE International Conference on Optoelectronics and Image Processing (ICOIP)*, volume 1, pages 313–316. IEEE, 2010.

[33] J. Yang, W. Tao, M. Liu, Y. Zhang, H. Zhang, and H. Zhao. An efficient direction field-based method for the detection of fasteners on high-speed railways. *Sensors*, 11(8):7364–7381, 2011.

[34] Hao Feng, Zhiguo Jiang, Fengying Xie, Ping Yang, Jun Shi, and Long Chen. Automatic fastener classification and defect detection in vision-based railway inspection systems. *Instrumentation and Measurement, IEEE Transactions on*, 63(4):877–888, April 2014.

[35] R.A. Khan, S. Islam, and R. Biswas. Automatic detection of defective rail anchors. In *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*, pages 1583–1588, Oct 2014.

[36] E. J. Candès and D. L. Donoho. New tight frames of curvelets and optimal representations of objects with c^2 singularities. *Comm. Pure Appl. Math.*, 57:219–266, 2004.

[37] A.L. Cunha, J. Zhou, and M.N. Do. The nonsubsampled contourlet transform: Theory, design, and applications. *IEEE Transactions on Image Processing*, 15(10):3089–3101, 2006.

[38] D. Labate, W. Lim, G. Kutyniok, and G. Weiss. Sparse multidimensional representation using shearlets. In *Wavelets XI (San Diego, CA, 2005)*, volume SPIE Proc. 5914, pages 254–262. SPIE, Bellingham, WA, 2005.

[39] G. Kutyniok and D. Labate. *Shearlets: Multiscale Analysis for Multivariate Data*. Birkhäuser, Boston, 2012.

[40] J.-L. Starck, F. Murtagh, and J. M. Fadili. *Sparse Image and Signal Processing: Wavelets, Curvelets, Morphological Diversity*. Cambridge books online. Cambridge University Press, 2010.

[41] K. Guo and D. Labate. The construction of smooth parseval frames of shearlets. *Math. Model. Nat. Phenom.*, 8(1):82–105, 2013.

[42] K. Guo and D. Labate. Optimally sparse multidimensional representation using shearlets. *Siam J. Math. Anal.*, 9:298–318, 2007.

[43] K. Guo and D. Labate. Optimally sparse representations of 3d data with c^2 surface singularities using parseval frames of shearlets. *Siam J. Math. Anal.*, 44:851–886, 2012.

[44] G. R. Easley, D. Labate, and W. Lim. Sparse directional image representations using the discrete shearlet transform. *Appl. Comput. Harmon. Anal.*, 25(1):25–46, 2008.

[45] G. Kutyniok, M. Shahram, and X. Zhuang. Shearlab: A rational design of a digital parabolic scaling algorithm. *SIAM J. on Imaging Sciences*, 5(4):1291–1332, 2012.

[46] K. Guo and D. Labate. Representation of fourier integral operators using shearlets. *J. Fourier Anal. Appl.*, 14:327–371, 2008.

[47] F. Colonna, G. R. Easley, K. Guo, and D. Labate. Radon transform inversion using the shearlet representation. *Appl. Comput. Harmon. Anal.*, 29(2):232–250, 2010.

[48] B. Vandeghinste, B. Goossens, R. Van Holen, C. Vanhove, A. Pizurica, S. Vandenberghe, and S. Staelens. Iterative ct reconstruction using shearlet-based regularization. *IEEE Transactions on Nuclear Science*, 60(5):3305–3317, 2013.

[49] K. Guo and D. Labate. Characterization and analysis of edges using the continuous shearlet transform. *SIAM on Imaging Sciences*, 2:959–986, 2009.

[50] K. Guo and D. Labate. Analysis and detection of surface discontinuities using the 3d continuous shearlet transform. *Appl. Comput. Harmon. Anal.*, 30:231–242, 2011.

[51] S. Yi, D. Labate, G. R. Easley, and H. Krim. A shearlet approach to edge analysis and detection. *IEEE Transactions on Image Processing*, 18(5):929–941, 2009.

[52] Gitta Kutyniok and Wang-Q Lim. Image separation using wavelets and shearlets. In *Curves and surfaces*, pages 416–430. Springer, 2012.

[53] G. Easley, D. Labate, and P. S. Negi. 3d data denoising using combined sparse dictionaries. *Math. Model. Nat. Phenom.*, 8(1):60–74, 2013.

[54] V. M. Patel, G. Easley, and D.M. Healy. Shearlet-based deconvolution. *IEEE Transactions on Image Processing*, 18:2673–2685, 2009.

[55] P.S. Negi and D. Labate. 3-d discrete shearlet transform and video processing. *IEEE Transactions on Image Processing*, 21:2944–2954, 2012.

[56] G. Easley, D. Labate, and V. M. Patel. Directional multiscale processing of images using wavelets with composite dilations. *Journal of Mathematical Imaging and Vision*, 2012.

[57] E. J. Candès, L. Demanet, D. Donoho, and L. Ying. Fast discrete curvelet transforms. *SIAM Multiscale Model. Simul.*, (5)(3):861–899, 2006.

[58] P. J. Burt and E. H. Adelson. The laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, 31(4):532–540, 1983.

[59] D. Donoho and I. Johnstone. Adapting to unknown smoothness via wavelet shrinkage. *J. Amer. Statist. Assoc.*, 90:1200–1224, 1995.

[60] S. G. Chang, B. Yu, and M. Vetterli. Adaptive wavelet thresholding for image denoising and compression. *IEEE Transactions on Image Processing*, 9(9):1532–1546, 2000.

[61] P. Subirats, J. Dumoulin, V. Legeay, and D. Barba. Automation of pavement surface crack detection using the continuous wavelet transform. In *IEEE International Conference on Image Processing*, pages 3037–3040, 2006.

[62] S. Chambon and J. Moliard. Automatic road pavement assessment with image processing: Review and comparison. *Int. Journal of Geophysics*, 2011(doi:10.1155/2011/989354), 2011.

[63] C. Ma, C. Zhao, and Y. Hou. Pavement distress detection based on nonsubsampled contourlet transform. *Int. Conf. on Computer Science and Software Engineering*, 1:28–31, 2008.

[64] J.-L. Starck, M. Elad, and D.L. Donoho. Image decomposition via the combination of sparse representation and a variational approach. *IEEE Transactions on Image Processing*, 14(10):1570–1582, 2005.

[65] J. Bobin, J.-L. Starck, M.J. Fadili, Y. Moudden, and D.L. Donoho. Morphological component analysis: an adaptive thresholding strategy. *IEEE Transactions on Image Processing*, 16(11):2675–2681, 2007.

[66] J. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[67] H. Oliveira and P.L. Correia. Automatic road crack detection and characterization. *IEEE Transactions on Intelligent Transportation Systems*, 14(1):155–168, 2013.

[68] L.F. Molina Camargo and J. Riley Edwards. Emerging condition monitoring technologies for railway track components and special trackwork. In *ASME/ASCE/IEEE Joint Rail Conference & Internal Combustion Engine Spring Technical Conference*, 2011.

[69] Tanmay Podder. Analysis & study of AI techniques for automatic condition monitoring of railway track infrastructure. Master's thesis, Dalarna University, Computer Engineering, 2010.

[70] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition (CVPR), 2005 IEEE Computer Society Conference on*, volume 1, pages 886–893, Jun 2005.

[71] Basler AG. Success story: ENSCO deploys Basler sprint and ace GigE cameras for comprehensive railway track inspection. `http://www.baslerweb.com/linklist/9/8/3/6/BAS1110_Ensco_Railway_Inspection.pdf`, Oct 2011.

[72] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[73] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):93–202, 1980.

[74] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989.

[75] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, November 1998.

[76] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Systems (NIPS)*, 2013.

[77] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *arXiv:1409.4842*, 2014.

[78] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Computer Society Conference on*, 2014.

[79] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv:1408.5093*, 2014.

[80] Ronan Collobert, Koray Kavukcuoglu, and Clement Farabet. Torch7: A matlab-like environment for machine learning. In *Advances in Neural Information Systems (NIPS)*, 2011.

[81] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. *arXiv:1411.4038*, 2014.

[82] Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, Jul 1997.

[83] L. Y. Pratt, J. Mostow, and C. A. Kamm. Direct transfer of learned information among neural networks. In *Proc. Of AAAI*, 1991.

[84] Geoffrey Hinton. Learning distributed representation of concepts. In *Proc. of the 8th Int. Conf. of the Cognitive Science Society*, pages 1–12, 1986.

[85] Y.-C. Chen, V. M. Patel, J. K. Pillai, R. Chellappa, and P. J. Phillips. Dictionary learning from ambiguously labeled data. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 353–360, June 2013.

[86] Jaume Amores. Multiple instance classification: Review, taxonomy and comparative study. *Artificial Intelligence*, 201:81–105, August 2013.

[87] Oded Maron and Tomás Lozano-Pérez. A framework for multiple-instance learning. In *Advances in Neural Information Processing Systems (NIPS), 1997 Conference on*, volume 10, pages 570–576, 1997.

[88] Ashish Shrivastava, Vishal M Patel, Jaishanker K Pillai, and Rama Chellappa. Generalized dictionaries for multiple instance learning. *International Journal of Computer Vision: Special Issue on Sparse Coding*, 2015.

[89] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[90] T. Ahonen, J. Matas, C. He, and M. Pietikäinen. Rotation invariant image description with local binary pattern histogram fourier features. In *Image Analysis*, pages 61–70. Springer, 2009.

[91] M. Muja and D.G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Application VISSAPP'09)*, pages 331–340. INSTICC Press, 2009.

[92] Timo Ojala, Matti Pietikäinen, and Topi Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(7):971–987, 2002.

[93] M. Haghighat, S. Zonouz, and M. Abdel-Mottaleb. Identification using encrypted biometrics. In *Computer Analysis of Images and Patterns*, pages 440–448. Springer, 2013.

[94] B.S. Manjunath and W.Y. Ma. Texture features for browsing and retrieval of image data. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 18(8):837–842, 1996.

[95] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[96] Peter J. Huber and Elvezio M. Ronchetti. *Robust Statistics*. Wiley series in probability and statistics. John Wiley & Sons, Hoboken, New Jersey, second edition, 2009.

[97] Ricardo A. Maronna, Douglas R. Martin, and Victor J. Yohai. *Robust Statistics: Theory and Methods*. Wiley series in probability and statistics. John Wiley & Sons, Chichester, England, 2006.

[98] P. J. Huber. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, 1964.

[99] D. Y. Kim, J. J. Kim, P. Meer, D. Mintz, and A. Rosenfeld. Robust computer vision: A least median of squares based approach. In *in Proc. of Image Understanding Workshop*, pages 1117–1134, 1989.

[100] J.B. Broadwater and R. Chellappa. Adaptive threshold estimation via extreme value theory. *IEEE Transactions on Signal Processing*, 58(2):490–500, 2010.

[101] E.J. Gumbel. *Statistics of Extremes*. Columbia University Press, New York, 1958.

[102] Walter Scheirer, Anderson Rocha, Ross Micheals, and Terrance Boult. Robust fusion: Extreme value theory for recognition score normalization. In *European Conference on Computer Vision (ECCV)*, pages 481–495. Springer, 2010.

[103] Walter J Scheirer, Anderson Rocha, Ross J Micheals, and Terrance E Boult. Meta-recognition: The theory and practice of recognition score analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(8):1689–1695, 2011.

[104] J. Pickands. Statistical inference using extreme order statistics. *The Annals of Statistics*, 3(1):119–131, jan 1975.

[105] S. Yun. The extremal index of a higher-order stationary markov chain. *The Annals of Applied Probability*, 8(2):408–437, may 1998.

[106] X. Gibert, V. M. Patel, and R. Chellappa. Deep multi-task learning for railway track inspection. *arXiv:1509.05267*, 2015.

[107] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

[108] F. E. Marchlinski, D. J. Callans, C. D. Gottlieb, and E. Zado. Linear ablation lesions for control of unmappable ventricular tachycardia in patients with ischemic and nonischemic cardiomyopathy. *Circulation*, 101(11):1288–1296, 2000.

[109] J. M. de Bakker, F. J. van Capelle, M. J. Janse, A. A. Wilde, R. Coronel, A. E. Becker, K. P. Dingemans, N. M. van Hemel, and R. N. Hauer. Reentry as a cause of ventricular tachycardia in patients with chronic ischemic heart disease: electrophysiologic and anatomic correlation. *Circulation*, 77(3):589–606, 1988.

[110] J. M. de Bakker, F. J. van Capelle, M. J. Janse, S. Tasseron, J. T. Vermeulen, N. de Jonge, and J. R. Lahpor. Slow conduction in the infarcted human heart. 'zigzag' course of activation. *Circulation*, 88(3):915–926, 1993.

[111] W. G. Stevenson, D. J. Wilber, A. Natale, W. M. Jackman, F. E. Marchlinski, T. Talbert, M. D. Gonzalez, S. J. Worley, E. G. Daoud, C. Hwang, C. Schuger, T. E. Bump, M. Jazayeri, G. F. Tomassoni, H. A. Kopelman, K. Soejima, and H. Nakagawa. Irrigated radiofrequency catheter ablation guided by electroanatomic mapping for recurrent ventricular tachycardia after myocardial infarction: the multicenter thermocool ventricular tachycardia ablation trial. *Circulation*, 118(25):2773–2782, 2008.

[112] T. Dickfeld, P. Lei, V. Dilsizian, J. Jeudy, J. Dong, A. Voudouris, R. Peters, M. Saba, R. Shekhar, and S. Shorofsky. Integration of three-dimensional scar maps for ventricular tachycardia ablation with positron emission. *JACC: Cardiovascular Imaging*, 1(1):73–82, 2008.

[113] J. Tian, M. F. Smith, P. Chinnadurai, V. Dilsizian, A. Turgeman, A. Abbo, K. Gajera, C. Xu, D. Plotnick, R. Peters, M. Saba, S. Shorofsky, and T. Dickfeld. Clinical application of pet/ct fusion imaging for three-dimensional myocardial scar and left ventricular anatomy during ventricular tachycardia ablation. *J Cardiovasc Electrophysiol.*, 20:597–604, 2008.

[114] J. Tian, M. F. Smith, H. Ahmad, V. Dilsizian, A. Jimenez, and T. Dickfeld. Integration of 3-dimensional scar models from spect to guide ventricular tachycardia ablation. *J Nucl Med*, 53(6):894–901, 2012.

[115] T. S. Fahmy, O. M. Wazni, W. A. Jaber, V. Walimbe, L. Di Biase, C. S. Elayi, F. P. DiFilippo, R. B. Young, D. Patel, L. Riedlbauchova, A. Corrado, J. D. Burkhardt, R. A. Schweikert, M. Arruda, and A. Natale. Integration of positron emission tomography/computed tomography with electroanatomical mapping: a novel approach for ablation of scar-related ventricular tachycardia. *Heart Rhythm*, 5(11):1538–1545, 2008.

[116] K. Kettering, H. J. Weig, W. Reimold, A. C. Schwegler, M. Busch, R. Laszlo, M. Gawaz, and J. Schreieck. Catheter ablation of ventricular tachycardias in patients with ischemic cardiomyopathy: validation of voltage mapping criteria for substrate modification by myocardial viability assessment using fdg pet. *Clin Res Cardiol*, 99(11):753–760, 2010.