MOBILE HYPERSPECTRAL IMAGING FOR

STRUCTURAL DAMAGE DETECTION

A THESIS IN
Civil Engineering

Presented to the Faculty of the University
Of Missouri-Kansas City in partial fulfillment of
The requirement for the degree

MASTERS OF SCIENCE

by
SAMEER ARYAL

B.E. Tribhuwan University, Kathmandu, Nepal, 2016

Kansas City, Missouri
2020

MOBILE HYPERSPECTRAL IMAGING FOR STRUCTURAL DAMAGE DETECTION

Sameer Aryal, Candidate for the Master of Science Degree

University of Missouri -Kansas City, 2020

ABSTRACT

Numerous optical-imaging and machine-vision based inspection methods are found that aim to replace visual and human-based inspection with an automated or a highly efficient procedure. However, these machine-vision systems have not been entirely endorsed by civil engineers towards deploying these techniques in practice, partially due to their poor performance in object detection when structural cracks coexist with other complex scenes. A mobile hyperspectral imaging system is developed in this work, which captures hundreds of spectral reflectance values at a pixel in the visible and near-infrared (VNIR) portion of the electromagnetic spectrum bands. To prove its potential in discriminating complex objects, a machine learning methodology is developed with classification models that are characterized by four different feature extraction processes. Experimental validation with quantitative measures proves that hyperspectral pixels, when used conjunctly with dimensionality reduction, possess outstanding potential in recognizing eight different structural surface objects including cracks for concrete and asphalt surfaces, and outperform the gray-values that characterize the texture/shape of the objects. The authors envision the advent of computational hyperspectral imaging for automating structural damage inspection, especially when dealing with complex structural scenes in practice.

APPROVAL PAGE

The faculty listed below, appointed by the Dean of the School of Computing and Engineering have examined the thesis titled “Mobile Hyperspectral Imaging for Structural Damage Detection”, presented by Sameer Aryal, candidate of the Masters of Science degree, and certify that in their opinion it is worthy of acceptance.

Supervisor Committee

ZhiQiang Chen, Ph.D., Committee Chair
Department of Civil and Mechanical Engineering

John Kevern, Ph.D., P.E., LEED AP
Department of Civil and Mechanical Engineering

Ceki Halmen, Ph.D., P.E.
Department of Civil and Mechanical Engineering

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

Figure Page

LIST OF TABLES

Table Page

ACKNOWLEDGMENTS

First and foremost, I would like to take this opportunity to express my deepest gratitude and thanks to my advisor, Dr. ZhiQiang Chen, for his countless effort in guiding me through studies and work, his patience with my learning and providing me with an excellent atmosphere for research. This thesis would not have been complete without his attitude and diligent effort which not only influences the content of this thesis but also the language in which it has been conveyed. I am sure it would have not been possible without his help.

Also, I would like to thank my colleague at the University of Missouri Kansas City: Prativa Sharma, Shimin Tang, and Mostafa Badroddin for their effortless help, valuable advice, and discussion. We had great and unforgettable times during all these years.

Last but not least, I am so thankful to my family whom they have been a continuous source of encouragement and supports in all directions during my life.

CHAPTER 1. INTRODUCTION

Civil engineering structures are complexly planned systems that are vital for a society's prosperity and quality of life in general. Ensuring the reputation, civil engineering structures have grown its dynamic demand around the globe over the past few decades. Apparently in the United States, there are over 610,000 bridges, 5,500,000 commercial buildings, 160,000 miles of railroad tracks, 4,000,000 miles of roads, 84,000 dams, 19,000 airports and 400,000 miles of electric transmission lines providing services to the population (ASCE, 2017). These structures are built and maintained to support the daily routine load as well as the additional unexpected loads and the unavoidable severe environmental conditions. For critical infrastructure systems, mandatory inspection practices and standards exist for adoption by stakeholders to ensure the serviceability and safety of the structure. One of them is for a comprehensive diagnostics and prognostics of serviceability of the national infrastructures, American Society of Civil Engineers (ASCE) has developed the 'Infrastructure Report Card' which grades the infrastructures as A- Exceptional, Fit for the Future; B- Good, Adequate for now; C- Mediocre, Requires Attention; D- Poor, At-Risk; and F- Critical Unfit for purpose.

Civil engineering structure inspection is more on the overall and general conditions, as can be directly observed or measured. The task of inspection for evaluation of civil engineering structure status has become increasingly challenging due to age, scale, and magnitude of structures. Different civil engineering inspection techniques are in practice to assist visual inspection. Exclusively these practices are to stipulate valuable information for structural assessment and decision support for maintenance through relevant measures of structural responses. These technologies can be generally categorized into two types of methodologies. The first is Nondestructive Testing (NDT),

utilizing advanced sensing technologies (microwaves, thermal and ultrasonic) (Cawley, 2018), most of which aim to detect subsurface damage. The second methodology includes various structural health monitoring (SHM) methods, which aim to monitor the dynamic responses and identify the intrinsic parameters or changes in structure. Most NDT techniques have become a growing field that has attracted a considerable amount of research efforts. Given these technology-based inspections or monitoring methods, the reality is that, at least for transportation structures that are managed by the department of transportation (DOT) agencies across most of the states in the US, manual or visual inspection is considered the mainstream approach. Sadly, despite the critical roles of these structures in public safety and economy, human-based visual inspection is common and consistent in quantitative evaluation and accessibility (Graybeal, Phares, Rolander, Moore, & Washer, 2002). Visual inspection is widely used mostly due to the expensive approach of the NDT techniques which demands a significant operational cost including training and deployment of manpower and technology in the field. Due to the low-cost and ubiquitous availability of optical imaging sensors or commonly speaking, digital cameras, it is of no surprise that optical imaging has become a widely adopted equipment for structural inspection, wherein besides visual inspection, digital images are recorded for records or for post-inspection analysis (M. J. Olsen et al., 2016). Among many methods for surveillance with the digital camera, one of them includes placing cameras at different critical locations around the structure and constantly monitoring the deformation and deterioration. This method covers only a small section of the structure and records mostly the textural information which alone is never enough for a complete structural assessment. To overcome the limitation, an alternative system is implemented which involves gathering images and registering the conditions of the surface by a skilled

technician traveling along the surface while taking pictures. After the structural surface images are captured, skilled technicians analyses each image and determine the existence of any distress and classify damage type based on visual descriptions. This process usually is time-consuming and requires a huge effort to analyze the full set of acquired images. Hence there arises a need for a rapid data acquisition and classification platform to collect, process and classify the structural surfaces in interest. These techniques have emerged with an essential goal of safeguarding the operational safety of structures, through deploying various types of sensors, monitoring diversified physical quantities, assessing structural condition and performance, and instructing routine inspection and maintenance. Subsequently, this has motivated the movement of developing machine vision techniques to aid or event to automate engineering inspection of civil structures.

Machine vision is a technical field that concerns the development of digital imaging methods and the use of image processing or computer vision algorithms for the extraction of useful information from images (Morris, 2004). With the advent of early digital cameras, researchers in the last eighties and nighties used simple digital filters, including various edge detection methods, for realizing image-based structural damage detection (Cheng & Miyojim, 1998; Ritchie, 1987; Ritchie Stephen, 1990). To further automate the process of image capturing, researchers further strive to develop other imaging methods that are expected to mitigate the human cost of professional inspectors. These novel methods include ground vehicle-based imaging, aerial vehicle-based imaging, and crowdsourcing based imaging [e.g., (Isawa et al., 2005; Kim, Sim, & Cho, 2015; Lattanzi & Miller, 2013; Tung, Hwang, & Wu, 2002; C. Zhang & Elaksher, 2012) (Ozer, Feng, & Feng, 2015)]. For example, Ho et al. (2013) developed a system with three cameras attached to a cable climbing robot to detect surface damage (Ho, Kim,

Park, & Lee, 2013). Yeum and Dyke (2015) proposed an unmanned aerial vehicle (UAV) for remote imaging and image-based detection (Yeum & Dyke, 2015). In Chen et al. (2015), a mobile-cloud infrastructure enabled approach was proposed that exploits collaborative mobile and cloud computing to harness crowdsourcing-based structural inspection (Chen, Chen, Shen, & Lee, 2015).

Unlike, regular digital imaging process, the author develops a mobile hyperspectral imaging (HSI) system for both ground and aerial vehicle-based remote sensing. With this HSI system and preliminary observation (e.g., by plotting spectral profiles for different structural surface objects), it is hypothesized that structural damage (e.g., cracks) and other complex artifacts can be effectively detected on structural surfaces. Furthermore, this HSI system equipped with a machine learning approach can outperform the performance of regular imaging methods with a high spatial resolution (i.e., those based on panchromatic or true-color imaging). The essential contribution of this thesis is the proven effectiveness of mobile HSI for structural surface damage detection with complex scenes. Different from any existing image-based structural damage detection method, in this study, the proposed framework deals with the detection problem with much semantically rich structural-surface materials and objects, including concrete, asphalt, crack, dry vegetation, green vegetation, water, oil, and artificial markings, which are dealt with in the literature of image-based damage detection but commonly found in engineered structures in service. Another significant contribution is the semantically labeled dataset resulting from this research, which provides an unprecedented basis for research in hyperspectral machine vision and engineering inspection automation.

Literature Survey

With the abundance of these optical imaging platforms, one promising fact is the ease of obtaining imagery structural-damage databases. Other than early vision methods, these databases enable the adoption of a machine learning paradigm for image-based structural damage detection. Most of the techniques in these early efforts used either of one of the gradient-based edge detection (Shrivakshan & Chandrasekar, 2012), Hugh transformed based line detection methods (Song & Lyu, 2005), wavelet-based processing method (Abdel-Qader, Abudayyeh, & Kelly Michael, 2003; M. Olsen, Chen, Hutchinson, & Kuester, 2012), image binarization method (Cheng, Shi, & Glazier, 2003; Oliveira & Correia, 2009), percolation method (Tomoyuki, Shingo, & Shuji, 2008) or, shape-based modeling method (Chen & Hutchinson, 2010; Huo, Yang, Li, & Zhou, 2017). Some studies explored the methodology for automated surface cracks monitoring and assessment of concrete surface, based on adaptive digital image processing Adhikari et al. (2014) and infrared thermography Sakagami (2015) whereas some other incorporated the displacement and strain measurement with digital imaging for crack defragmentation (Adhikari, Bagchi, & Moselhi, 2014; Sakagami, 2015; Valença, Dias-da-Costa, Gonçalves, Júlio, & Araújo, 2014). Many different integration approaches of two or more sensors were explored and broadened to be used in more specific application categories. Vaghefi et al. (2015) developed a combined nondestructive imaging technology on the bridge deck to yield both surface and subsurface indicators of the condition (Vaghefi, Ahlborn Theresa, Harris Devin, & Brooks Colin, 2015). Stabile et al. (2012) used a suite of microwaves radar interferometer and a thermal camera to monitor the dynamic displacement of bridges (Stabile et al., 2012). Waldbjorn et al. (2014) obtained the feedback signals i.e. strain and displacement by fiber Bragg grating and digital image

correlation aligned to monitor the mandrel position by measuring the rigid body displacement based on a multivariate least-squares algorithm (Waldbjørn et al., 2014). Other than early vision methods, these databases enable the adoption of a machine learning paradigm for image-based structural damage detection. As of today, many machine learning methods are found, which feature the use of supervised or non-supervised classifiers (Chen, Derakhshani, Halmen, & Kevern, 2011; Gavilán et al., 2011; Kaseko & Ritchie, 1993; Liu, Suandi, Ohashi, & Ejima, 2002; Prasanna et al., 2014; Zakeri, Nejad, & Fahimifar, 2017). In recent years, coincident with the advances in artificial intelligence (AI), and particularly the development of deep learning techniques, many have heralded the era of AI-enabled structural inspection. To this end, a simple search through Google Scholar, using the combined keywords of "Crack Detection", "Convolutional Neural Network" (CNN), and "Image" returns more than 700 articles within the period of January 2016 to October 2019. Notably, Zhang et al. firstly used a CNN model as a feature extractor then fed the features into a classification model for the detection of cracks in images (L. Zhang, Yang, Zhang, & Zhu, 2016). Such a CNN-based machine learning approach is then adopted in many other similar efforts [e.g., (Alipour, Harris, & Miller, 2019; Cha, Choi, & Büyüköztürk, 2017; Ni, Zhang, & Chen, 2019)]. One may expect that by duly considering the advances in these AI-enabled image-based damage detection methods and the lowering cost of mobile or edge computing devices, the notion of an autonomous structural inspection may become a reality. The authors in this paper argue that if a fundamental fact is not acknowledged, the pace of automation would ultimately be hindered. This fact is the complexity of structural scenes captured in digital images. In the case of concrete structures, the scenes in images are often a mixture of structural materials, possible damage, and other artifacts, such as artificial marking,

vegetation, moisture, oil spill, discoloring, and uneven illumination (Chen & Hutchinson, 2010). This implies that any image-based machine learning method or an end-to-end deep learning method may encounter the infamous issue of generalization. In other words, if such an autonomous image-based system is deployed in the field, its core detection component (i.e., a classification model) even trained based on a relatively large dataset with complex scenes, can over-fit the training data but cannot generalize to an arbitrary scene that is more complex than the data used for training.

To resolve this challenge, one obvious solution is to continue developing much larger dataset given the power of deep learning with an architecture that can potentially accommodate any scale of data sizes and any complexity in field scenes, when regular images (i.e., true-color images with red, green, and blue bands or RGB images) are continuously used. However, this inevitably triggers the issue of labeling big data (e.g., pixel-wise labeling of cracks and other artifacts), which is expensive and time-consuming (Roh, Heo, & Whang, 2019). Another approach is to resort to transfer learning and use small data sets enhanced by effective data augmentation technique to obtain the notion of learning from small data using DL models. A recent effort of such is reported (Shimin Tang & Chen, 2017), which develops a crack pixels-based data augmentation technique for fine-tuning of DL models. Regardless of the potential success in these solutions, it is asserted that with the use of RGB images, the outcomes of developing these methods can only asymptotically match the intelligence of trained inspectors, though possibly with much higher efficiency than human inspectors. In other words, there is a performance 'ceiling' that tops the capacity of regular RGB images unless that machine intelligence supersedes human beings.

An alternative solution is to break out the normal of matching human vision. An emerging technology for structural inspection is hyperspectral imaging (HSI). In a hyperspectral image, a pixel contains tens to thousands of digital values at different spectral bands in the visible and near-infrared (VNIR) portion of the electromagnetic spectrum bands, at which each digital value represents either the reflectance or transmittance property of a material at one band. Such a high-dimensional spectral profile hence is not directly visible to human eyes that respond, roughly speaking, only to three discrete bands (namely, red, green, and blue)(Kaiser & Boynton, 1996). Scientific knowledge in hyperspectral imaging and analysis is well archived and is, in general, termed hyperspectral spectroscopy (Siesler, Ozaki, Kawata, & Heise, 2008). In the context of image-based structural damage detection, it is stated that HSI provides a significant possibility of detecting and identifying the presence of either structural damage or noisy artifacts at the material level.

In this work, the authors develop a mobile hyperspectral imaging (HSI) system for both ground and aerial vehicle-based remote sensing. With this HSI system and preliminary observation (e.g., by plotting spectral profiles for different structural surface objects), it is hypothesized that structural damage (e.g., cracks) and other complex artifacts can be effectively detected on structural surfaces. Furthermore, this HSI system equipped with a machine learning approach can outperform the performance of regular imaging methods with a high spatial resolution (i.e., those based on panchromatic or true-color imaging). Different from any existing image-based structural damage detection method, in this study, the proposed framework deals with the detection problem with much semantically rich structural-surface materials and objects, including concrete, asphalt, crack, dry vegetation, green vegetation, water, oil, and artificial markings, which

are dealt with in the literature of image-based damage detection but commonly found in engineered structures in service. Another significant contribution is the semantically labeled dataset resulting from this research, which provides an unprecedented basis for research in hyperspectral machine vision and engineering inspection automation.

Literature Survey Summary

To summarize the literature review, certain areas are yet to be explored and new improved systems are yet to be developed for the damage detection in civil engineering structures. While there have been researches focused on the detection of cracks, defragmentation and damage assessment on the surface using images and videos but these results lack when considering the complex and realistic scenes. This review shows that the implementation of a computer vision-based method for non-destructive testing and its potential to provide more valuable information for the visual inspection and structural condition assessments through integration with other sensing techniques as well as presents some critical limitations and challenges of the system. Most of the current research is conducted through the images captured in a controlled environment. The quality of the image captured by the vision device will be significantly affected by the surrounding environment condition such as mixer of the contrast from other similar materials, light variation, presence of oil or water on the surface and artificial marks on the surface which are very common on the structural surface. Along with image quality limitation, the majority of the current literature focuses on binary classifications using simple machine learning techniques or threshold-based heuristics. However, multiclass classification has not yet been explored for different classes of damages found on the civil engineering structures.

In the following, first, the concept of HSI is briefly introduced, and a mobile HSI system is described. In the next, the machine learning methodology is introduced with a focus on proving the concept of HSI-based detection and its competitive performance. Performance evaluation and discussion are further conducted with four classification models, followed by a summary of conclusions and vision at the end.

CHAPTER 2. HYPERSPECTRAL IMAGE

When a beam of white light is dispersed by passing through a prism, a continuous range of colors, the so-called color the spectrum is formed. All object gives off electromagnetic radiation and it has been known that different materials emit, reflect and absorb a different proportion of lights and this proportion is the function of the frequency of the light wave (Richards & Jia, 1999). Since the color spectrum visible to the human eye is only a small region of the much wider electromagnetic spectrum thereby detecting and analyzing the energy emitted or reflected, an enormous amount of information about the material into consideration can be obtained. This specific property of the physical object is called reflectance. The reflectance of an object varies at different wavelength producing a unique electromagnetic spectrum profile for each object.

The imaging spectroscopy is defined as "the simultaneous acquisition of the measurement, processing, and analysis of images in many narrow, contiguous spectral bands" (Goetz, Vane, Solomon, & Rock, 1985). The concept of HSI originated in the 1980s when Goetz and his colleagues at the Jet Propulsion Laboratory (JPL) began developing the seminal instrument of the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) (R. O. Green et al., 1998; Plaza et al., 2009). Different from gray-level or RGB images, in a hyperspectral image, a hyperspectral pixel consists of a large number of intensity values sampled at different narrow spectral bands that represent the contiguous spectral curve at the pixel. A Hyperspectral image, in general, can be assumed as a 3D data cube structure, where a 2-D spatial-domain resides over a 1-D spectral-domain. One may view each hyperspectral data cube as a stack of spatially registered 2D images at different wavelengths (bands). Each pixel is a 1-D vector, corresponds to the reflectance energy spectrum within its field of view (FOV) (Richards & Jia, 1999). Figure1 shows

full three-dimensional hyperspectral (two spatial dimensions plus one wavelength dimension) data cube.

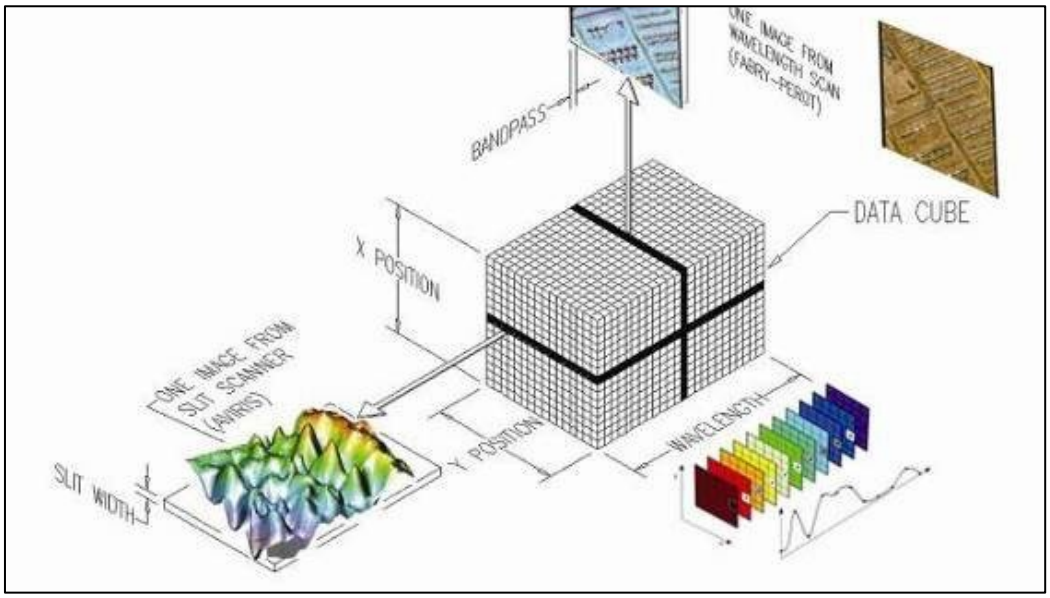


Figure1. A hyperspectral image data (Bodkin et al., 2009a)

For hyperspectral images obtained by advanced hyperspectral cameras, detailed spectral information and fine spatial resolution enable an analysis of both materials and structures of the object in a scene. Therefore, it is necessary to develop new techniques to exploit these underlying spatial and spectral information in hyperspectral images, thus advancing the limitation of human vision, computer vision, and remote sensing. Some attempts have been made in both computer vision and remote sensing over time but still, there has been a huge gap between hyperspectral imaging and material classification application due to lack of effective spectral-spatial feature extraction method as well as due to lack of enough data and robust classification method.

Hyperspectral Imaging Technology

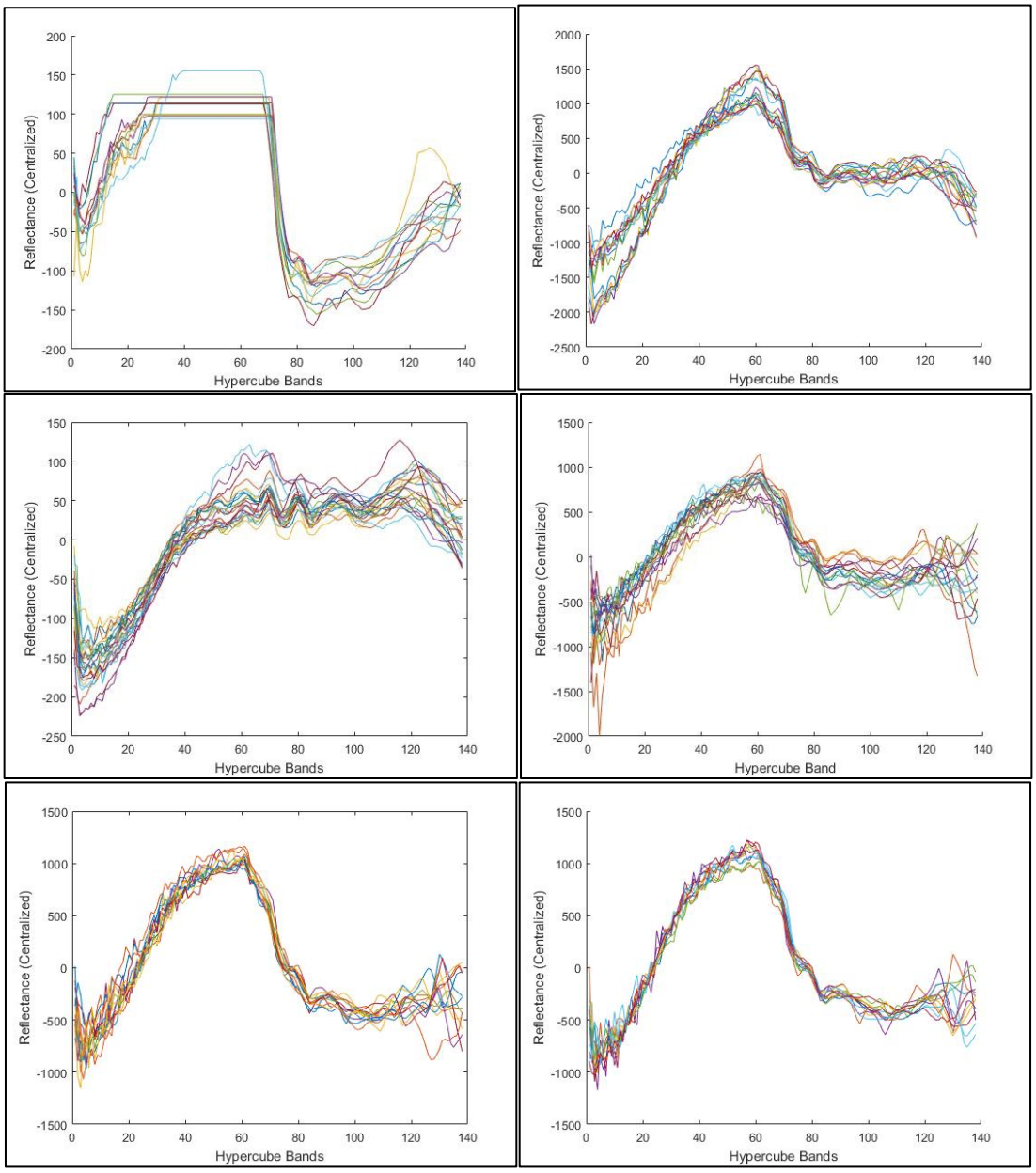
As advances in HSI and especially sensors that are not for orbital or airborne platforms, the acquisition of hyperspectral data cubes can be realized with other mechanisms. Besides the spatial scanning or push-broom imaging mechanism, two other

mechanisms (spectral scanning and spatial-spectral scanning) are developed for HSI applications in medical and biological sciences (Lu & Fei, 2014). It is noted that towards producing a hyperspectral cube, these three scanning techniques require complex post-processing steps to achieve the end product, a data cube. The fourth mechanism is the non-scanning or 'snapshot' imaging (Johnson, Wilson, Bearman, & Backlund, 2004). The snapshot imaging, different from other, acquires spectral pixels in a 2-D field-of-view simultaneously, without the requirement of trajectory flights or using any moving parts in the imager. Therefore, this 'snapshot' mechanism is also referred to as real-time HSI by researchers (Bodkin et al., 2009b). This HSI mechanism can achieve much higher frame rates and higher signal-to-noise ratios and can provide hyperspectral cubes immediately after every action of capturing as in a regular digital camera. Due to this property, real-time or 'snapshot' HSI opens up significant opportunities for its use in portable, mobile, or low-altitude remote sensing.

Hyperspectral Image Computing

Given a hyperspectral cube, one can denote it as $h(x, y, s)$ acquired from several spectral bands (i.e., for visible bands $s \in [400, 600]$ nm; and visible to near-infrared, $s \in [400, 1{,}000]$ nm). At a select location of (x, y), therefore, $h(x, y, s)$ represents a spectral profile when plotted against the variable spectral s. In the remote-sensing context (not in a medical or biological context), namely, the data cube is acquired in the air, and the measurement at the sensor is the upwelling radiance. In general, it is the reflectance property of a material at the ground that nominally does not vary with solar illumination or atmospheric disturbance. Therefore, the acquired spectral profile reflects the characteristics or signatures of the material. Therefore, a raw radiance data cube needs to be corrected to generate a reflectance cube, considering the environmental lighting and

the atmospheric distortion. This process is called atmospheric correction (Adler-Golden et al., 1998). Figure 2. Below represents the plot for the reflectance plot for each material class we have opted to work within this thesis work.



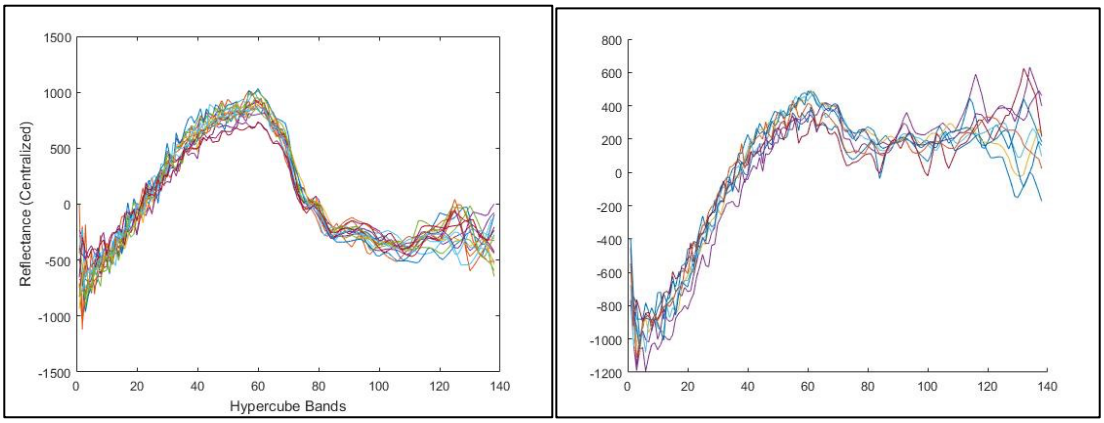


Figure 2. Reflectance plot for asphalt, color, concrete, crack, dry vegetation, oil, green vegetation, and water respectively with 20 random pixels.

Camera Calibration

The raw spectral image collected using the hyperspectral imaging system is detector signal intensity. To calibrate the raw intensity images into reflectance, calibration of the camera is performed with the help of a black and white reflectance image. This process corrects the significant signal vibrations, which are caused by non-uniformity of the illumination and the focal plane array of the camera, known as pattern noise (Nouri, Lucas, & Treuillet, 2013). Natively, the imager captures radiance images, and with the internal processing and a proper calibration procedure, the camera can output reflectance images directly. To do so, a reflectance calibration process starts with the use of a standard white reference board, achieving a data cube for the standard whiteboard (denoted as h_W). Second, a 'perfect dark' cube is obtained (by simply covering the lens tightly with a black cap), denoted as h_B. The relative reflectance image, h, is calculated given a radiance cube h_R,

$$h = \frac{h_R - h_W}{h_W - h_B} \quad (1)$$

Following Eq. (1), a reflectance image can be produced by the camera directly or can be post-processed from the produced radiance image.

CHAPTER 3. PREPROCESSING

This section describes the data collection process with the developed application, specifications of the hardware, data format, and information about the environmental setup are also described in detail. For the data acquisition and processing, the most important components are the camera and its specification that determines the resulting quality of the collected data.

Imaging System

A mobile HSI system for ground-level and low-altitude remote sensing is developed by the authors. The imaging system consists of a Cubert S185 FireflEYE snapshot camera that combines the precision of hyperspectral camera with the ease of snapshot camera, accurately capturing data over the whole field of view, and a mini-PC server for onboard computing and data communication (Cubert Gmbh, 2018). For ground-based imaging, the system is mounted to a DJI gimbal that provides two 15-W and 1580 mAh batteries for powering both the imaging payload and the operation of the gimbal. Figure 3 shows the gimbaled imaging system, which is ready for hand-held or other ground-based HSI. To enable low-altitude remote sensing, an unmanned aerial vehicle (UAV) is used, and the gimbaled system can be easily installed to the UAV for remote sensing.

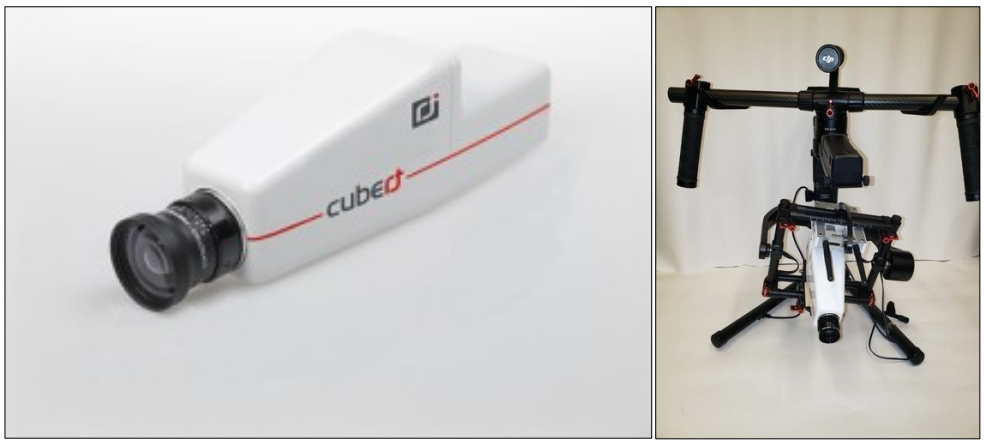


Figure 3. Cubert hyperspectral camera and assembly at UMKC

This device has a wavelength range of 450nm to 950nm with a spectral resolution of 8nm capturing 139 channels and a pan resolution of 2500 spectral per cube providing a complete hyperspectral cube with a global shutter in 1/1000 of a second, without the need of IMU. As per the manufacturer, the wavelength accuracy at 532nm and 808nm are respectively ±2.5nm and ±4.5nm. One unique feature of the Cubert HSI system is its dual acquisition of hyperspectral cubes and a companion image, a gray-level intensity image. The gray-level image has an identical field of view as the hyperspectral cube but has a much higher spatial resolution, which has a size of 1,000 × 1,000. Denoting this gray image as $g(u, v)$, one can 'fuse' $g(u, v)$ and $h(x, y, s)$ to achieve a hyperspectral cube with a higher resolution, and at its peak, one can obtain a cube with the size of 1,000 × 1,000 × 139. This process is called pan-sharpening, and to obtain smooth sharpening effects, many algorithms exist (Loncan et al., 2015). Nonetheless, it is noted that pan-sharpening, which can provide visually appealing hyperspectral images (if visualized in terms of pseudo-color images), does not provide new information compared to the original low-resolution hyperspectral cube and the high-resolution gray image. Therefore, in this paper, the low-resolution data cubes are directly used towards the goal of pattern classification-based object detection.

The Cube-Pilot is the official graphical user interface (GUI) to the Cubert Hyperspectral cameras making it possible to calibrate the camera before taking any pictures and aiding in the process of image capturing. A window of the Cube-pilot application is shown in Figure 4.

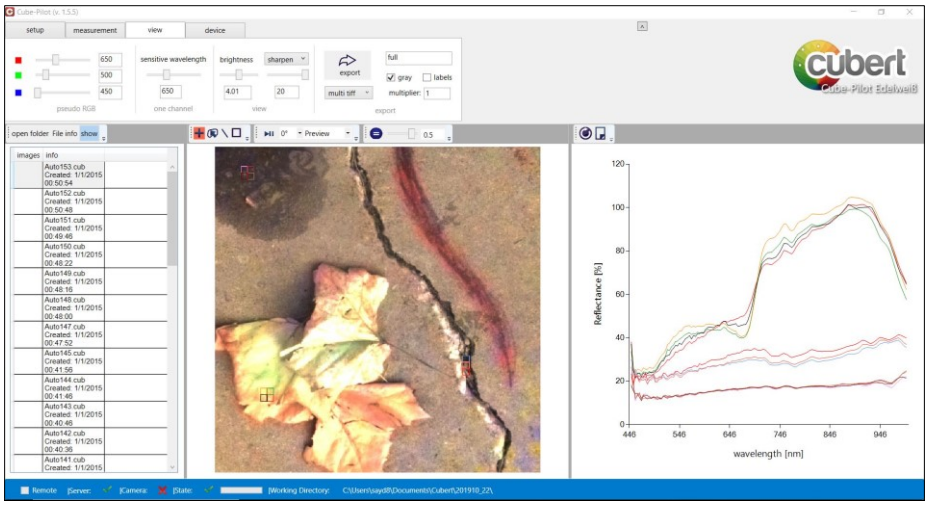


Figure 4. Cube pilot application to visualize and extract the hyperspectral data.

Semantic Labelling

With the mobile HSI system (Figure. 3), a total of 68 instances of hyperspectral images (and their companion gray-level images) were captured in the field. Among these images, 43 images come from concrete surfaces and 25 images from asphalt surfaces. To create scene complexity, artificial markings, oil, water, green and dry vegetation were added in 34 of concrete images and 16 of asphalt images that have hairline or apparent cracks. Of the remaining 18 images, 9 images were taken from the surfaces of concrete and asphalt pavements without cracks and any of the other artifacts, respectively.

To create a supervised learning-ready dataset, manual and semantic labeling is carried out. Semantic labeling is the process of labeling each pixel in an image with a corresponding class label. In this work, an image-segmentation (or image parsing) based labeling approach is considered in which clustered segment with pixels belong to the same class is delineated in the image domain and rendered with a select color. The image of labeling is based on the gray-level image that accompanies a hyperspectral cube. In this work, this process was conducted by using an open-source image processing program, GIMP (Kimball & Mattis, 2019). As shown in Figure 5, during the labeling

process, a total of six different classes, including cracking, green vegetation, dry vegetation, water, oil, and artificial marking, are assigned with the color of black, green, brown, blue, red, and yellow, respectively. It is noted that in this effort, the background materials (concrete and asphalt) are not classified in these complex-scene images as well as in the plain (concrete/asphalt) images. Figure 5 shows two samples of the original gray-level images and the resulting color-rendered mask images for a concrete surface and an asphalt surface, respectively.

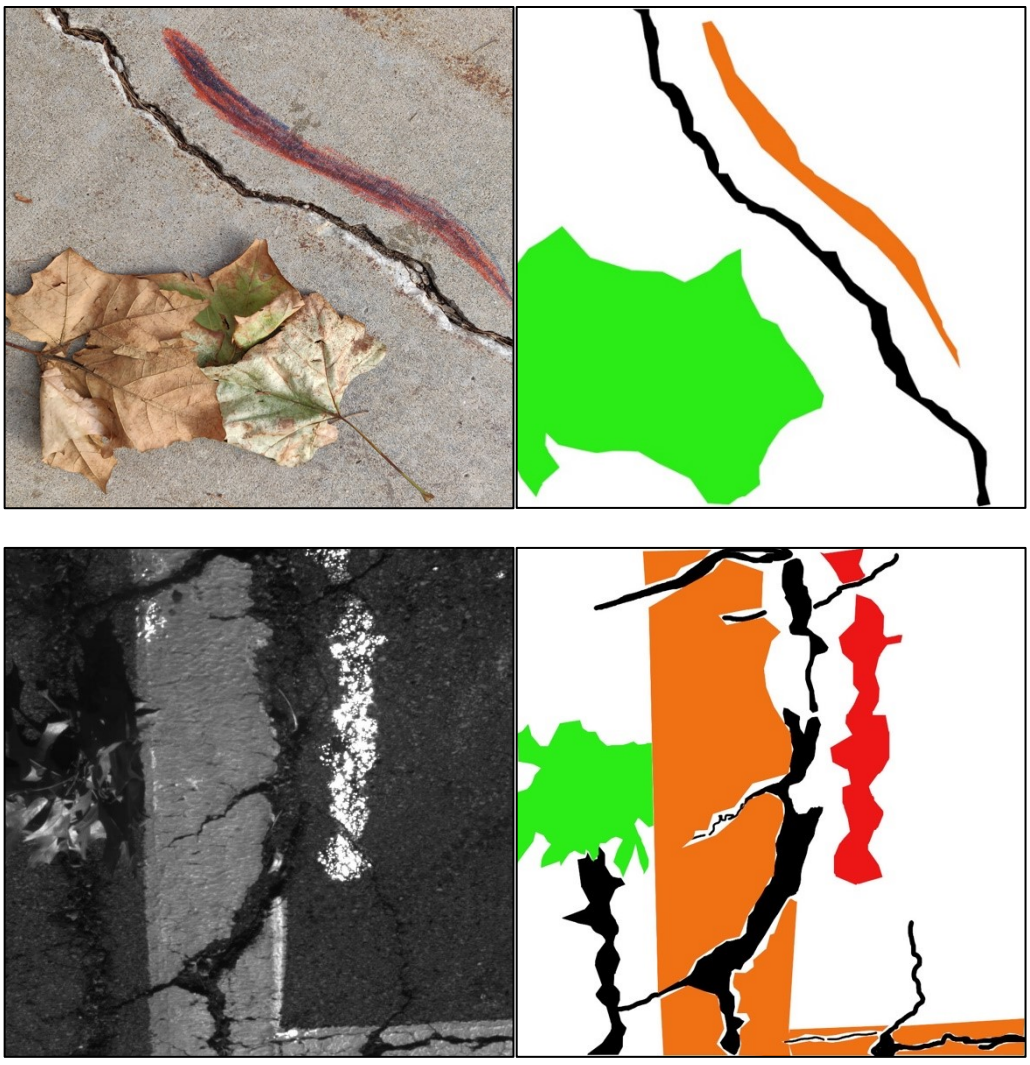


Figure 5. Images of concrete and asphalt surface with features and their respective ground truth images

CHAPTER 4. METHODOLOGY

This section describes the proposed architecture and methodology followed in this thesis. Four machine learning algorithms based on the traditional machine-learning paradigm (namely, manually tuned feature extraction and select classification are employed) are designed in this work. With these algorithms, the specific objectives are two-fold:

1. Hyperspectral data can improve the accuracy of detection compared to gray-level images.
2. Dimensionality reduction can further improve the accuracy and robustness compared against the case without dimensionality reduction.

Depending on the feature extraction methods, the following models are obtained. The list below summarizes these four models with abridged notations and their primary testing goals.

1. Model-1 or M1: feature extraction based on hyperspectral pixels with spectral values directly used as feature vectors. Namely, $h(x, y, s)$ at (x, y) is directly used as a feature vector, where $s \in \{1,2, \dots, 139\}$. To reflect this characteristic, an abridged notation M1(HYP) is used, where HYP represents the feature extraction process.
2. Model-2 or M2: feature extraction based on hyperspectral pixels with spectral values subject to a linear PCA as an additional feature selection step to reduce the dimensionality. Namely, the profile of $h(x, y, s)$ at (x, y) is reduced to six dimensions only and becomes $h'(x, y, k)$, where $k \in \{1,2,3,\dots,6\}$. For this model, M2(HYP_PCA) is used for simplicity. The flowchart for model
3. Model-3 or M3: feature extraction based on the companion gray-level images, $g(u, v)$, where the feature vectors at a 20 × 20 neighborhood in $g(u, v)$ maps to the hyperspectral pixel at (x, y). To extract the gray-level features within a sliding 20×20 neighborhood in

$g(u, v)$, the widely used gradient-based feature extractor, the histogram of gradients, or HOG, is considered and a variant of HOG is adopted in this paper. The resulting model is denoted by M3(GL_HOG.

4. Model-4 or M4: feature vectors based on the combined use of the feature vectors used in Model-2 and Model-3. Namely, by concatenating the two feature vectors, it fuses imagery information from both the hyperspectral pixel-based spectrum and the gray-value based spatial distribution. Hence, the notation of M4(GL_HOG+HYP_PCA) is used for simplicity, and GL_HOG+HYP_PCA represents the fourth feature extraction process in this paper.

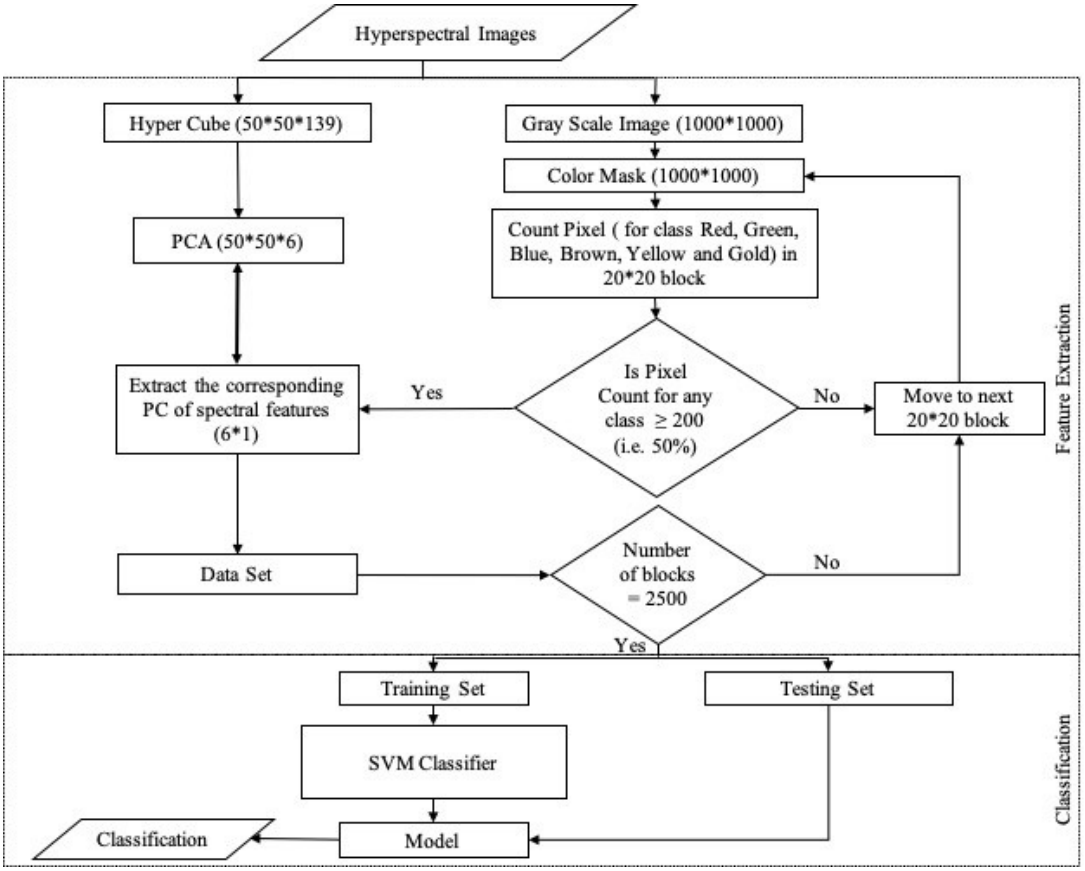


Figure 6. Dimensionality reduction and classification approach for the hyperspectral image.

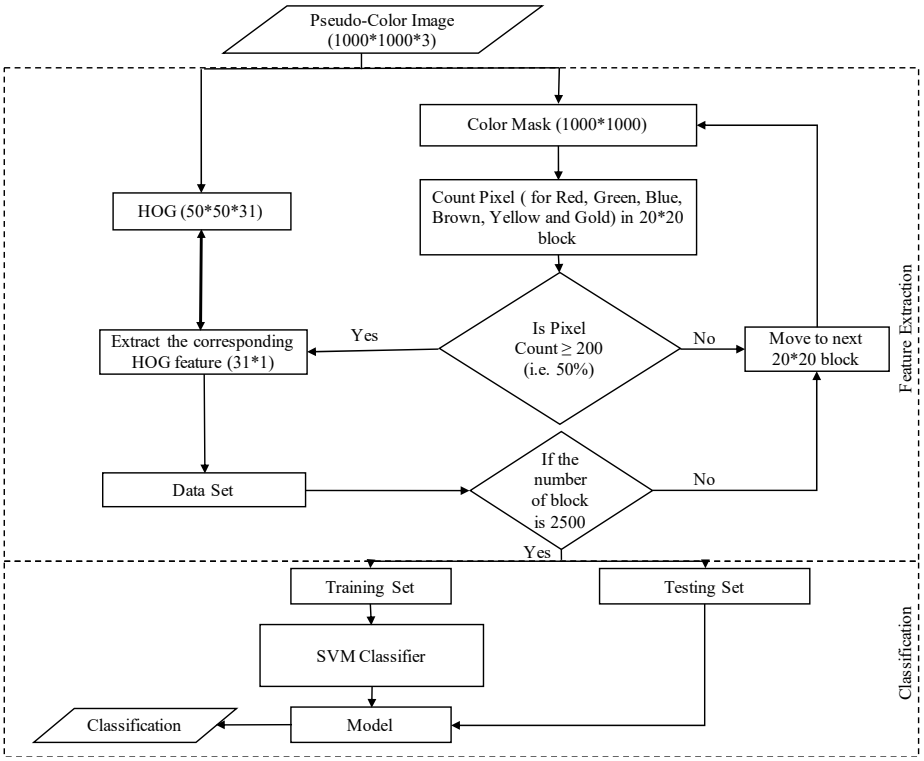


Figure 7. HOG feature extraction and classification approach.

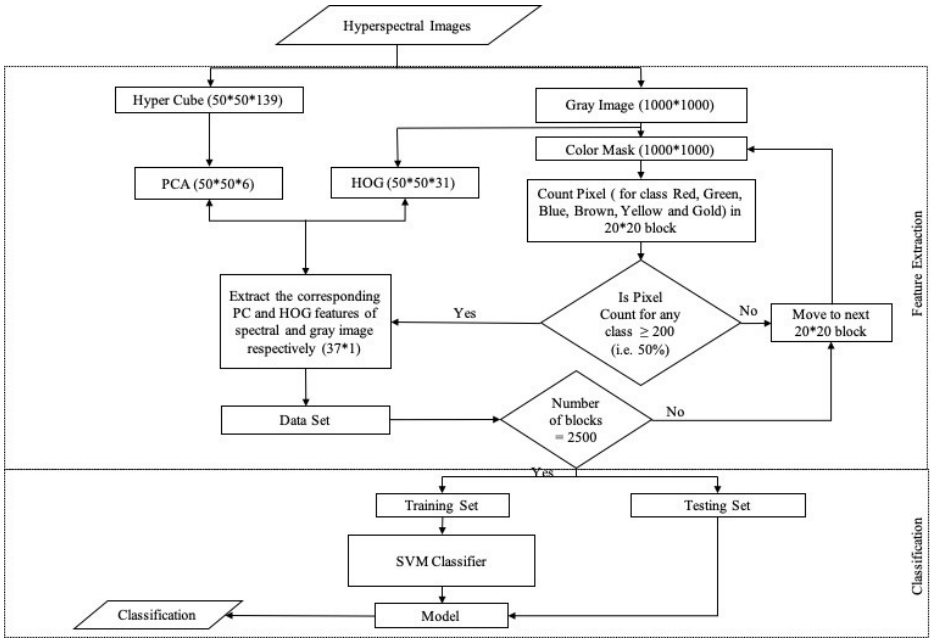


Figure 8. A combined PCA and HOG feature extraction and classification approach.

With these models, the specific objectives towards proving the hypotheses are multi-fold:

- Objective-1: Evaluate the performance of the classification model of M1(HYP) hence to conclude if a hyperspectral pixel is effective in recognizing the underlying object types, including structural damage given a complex scene.
- Objective-2: Evaluate and compare the performance of M1(HYP) and M2(HYP_PCA), hence, to conclude if dimensionality reduction is effective in terms of improving the discrimination of different objects.
- Objective-3: Evaluate and compare the performance of M2(HYP_PCA) and M3(GL_HOG), hence, to conclude if hyperspectral pixels are more effective than high-resolution gray-level images towards identifying complex object types.
- Objective-4: Evaluate the performance of the classification model of M4(HYP_PCA, GL_HOG), hence, to conclude if, through simple data fusion, the combined hyperspectral and gray-level features provide more competitive detection performance.

With the four models defined previously, they essentially differ in the use of different feature extraction processes based on the original hyperspectral data instance (a data cube and a companion gray-level image). With the colored mask images created as described above, it is denoted as $m(u, v)$ sharing the same spatial domain as the underlying gray image $g(u, v)$. For the sake of simplicity, based on the color coding for the mask images, the value of $m(\cdot)$ takes an integer value of 1, 2, …, 6 to indicate the underlying six different surface objects; in addition, the following notations are used to describe the resulting dataset:

$$\mathcal{D} = \{ h_n(x, y, s), g_n(u, v), m_n(u, v) | \, n = 1,2, \dots, 50\} \tag{2}$$

To generate the machine-learning data for the models, the following feature extraction and treatment process is developed. Considering the spatial domains of a pair of $h(x, y, s)$ and $g(u, v)$, the following procedure is proposed:

1) Iterating with the location of (x_i, y_j) with $i, j = 1, 2, \ldots, 50$, the spectral profile is stored in the vector of $\{h(x_i, y_j, s) | s \in [1, 139]\}$, and the gray-values in the corresponding gray image $g(u, v)$ are confined in a neighborhood block of $\mathscr{b} = \{(u', v') \mid u' \in [(x_i - 1) \times 20 + 1, x_i \times 20]$, and $v' \in [(y_j - 1) \times 20 + 1, y_j \times 20]\}$. At this neighborhood of $\mathscr{b}$, the gray-level image patch and the corresponding mask patch corresponding to the hyperspectral pixel at (x, y) are denoted as $g(\mathscr{b})$ and $m(\mathscr{b})$, respectively.
2) Given the mask patch $m(\mathscr{b})$, a simplified process is used to select the underlying class label for the hyperspectral pixel at (x_i, y_j). By counting the number of pixels belong to different object types within the neighborhood block $\mathscr{b}$,
    a. If a dominant class label exists, namely the number of pixels that belongs to a class is greater than 50% of the total pixels in the block (namely, 200 over 400 pixels), this class label is assigned to (x_i, y_j).
    b. If no dominant class label exists, this pixel (x_i, y_j) and the corresponding neighborhood $\mathscr{b}$ is skipped.
3) At a pixel with a dominant class label, and per the feature extraction method (HYP, HYP_PCA, and GL_HOG),
    a. If HYP is used, $\{h(x_i, y_i, s) | s \in [1, 139]\}$ is directly used as the feature vector with a dimension of 139×1.
    b. If HYP_PCA is used, PCA is conducted over the vector $\{h(x_i, y_i, s) | s \in [1, 139]\}$, and the first 6 PC scores are used to form a much low-dimensional 6×1 feature vector.
    c. If GL_HOG is used, the feature extraction is based on the gray-level patch $g(\mathscr{b})$ using the HOG-UoCTTI method, resulting in a 31×1 feature vector.

d. If HYP_PCA + GL_HOG is fused, the two corresponding feature vectors are simply concatenated, resulting in a 37 × 1 feature vector.

4) By iterating this procedure over all the hyperspectral pixels for all the 50 instances of images which includes different types of features in consideration, the following classification data set is obtained for each of the feature extraction methods above.

$$\mathcal{D}^{FEA} = \{ (\boldsymbol{p_k}, c_k) | \, k = 1,2, \dots, K\} \tag{3}$$

where the superscript *FEA* represents one of the feature extraction processes: HYP, HYP_PCA, GL_HOG, or HYP_PCA+GL_HOG. It is noted that by skipping many background pixels or pixels that do not have dominant labels in Step 2b, the resulting number of meaningful pixels (with dominant class labels) is 29546. Among them, 8132, 6495, 6273, and 5312 features are obtained for the class labels (c_k's) of water, oil, artificial marking and green vegetation, respectively. The number of features for cracks (concrete and asphalt cracks) is 2377. The dry vegetation features have the lowest number of 957.

With the data cubes and gray images for the plain concrete and asphalt surfaces (9 pairs each), 2601 features are arbitrarily extracted at each of feature extraction type but without using mask images each for the concrete or the asphalt labels. After adding these features into Eq. 3, the number of labeled features used in this paper, or K in Eq. 3, is 34748. As described above, given the lowest (957) and the largest (8132) number of features, a moderate imbalance indeed exists.

Principal Component Analysis (PCA)

Principal component analysis (PCA) is the most widely used linear-dimension method based on second-order statistics. PCA is also known as the Karhunen-Loeve transformation, singular value decomposition (SVD), empirical orthogonal function

(EOF), and Hotteling transformation. PCA is a mathematical procedure that facilitates the simplification of large data sets by transforming many correlated variables called principal components. PCA finds a new set of orthogonal axes that have their origin at the data mean and are rotated to a new coordinate system so that the spectral variability is maximized. Resulting PC bands are linear combinations of the original spectral bands and are uncorrelated.

Given a hyperspectral profile at (x, y), or denoted as a set $\{h(x, y, s) \mid s \in [1, 2, \ldots, 139]\}$, if treated as a feature vector, it gives rise to a 139×1 feature vector. As mentioned earlier, such high-dimensionality readily leads to poor performance when training a classification model (particularly when the training data is small, and the model itself cannot accommodate the high-dimensional space). Theoretically, assuming that an image had n pixels, measured at k spectral bands, the matrix characterizing the image is as follows.

$$\mathbf{X} = \begin{bmatrix} x_1 \\ \vdots \\ x_k \end{bmatrix} \tag{4}$$

where $x_1 \ldots x_k$ is a vector of n elements.

The first step in the PC procedure is generally the subtraction of the mean from each of the data dimesons. The mean spectrum vector represents the average brightness value of the image in each band and is defined by the expected value as follows:

$$\boldsymbol{A} = \frac{1}{N}\sum_{j=1}^{N} x_j - \begin{bmatrix} x \\ \vdots \\ x_k \end{bmatrix} \tag{5}$$

Where $\boldsymbol{A}$ is the mean spectrum vector, N is the total number of image pixels, and x_j is a vector representing the brightness of the j^{th} pixel of the image. Therefore, the components of the mean spectrum vector $\boldsymbol{A}$ represent the average brightness of the image in each band. The mean shift is calculated by subtracting the mean of the data. The PC

analysis de-correlates the data mainly by rotating the original axes, and therefore, the mean shift does not change the attribute of the resulting PC images. The only difference is the addition of a constant value in each band. This makes the decorrelation more evident in subsequent stages but is not necessary.

The second step in the PC method is to calculate the covariance matrix, which is a square symmetric matrix, where the diagonal elements are variance and the off-diagonal elements are covariance. From a spectral imagery point of view, the variance represents the brightness of each band and the covariances represent the degree of brightness variation between bands in the image. Additionally, covariance that is large compared to the corresponding variance in a spectral pair indicates a high correlation between these bands while covariance close to zero indicates little correlation in these spectral pairs (Richards, 2013).

The covariance matrix is computed by the formula

$$\boldsymbol{C} = \frac{\mathbf{1}}{\boldsymbol{n-1}} (\mathbf{X} - \mathbf{A})(\boldsymbol{X} - \boldsymbol{A})^{\boldsymbol{T}} \tag{6}$$

Where $\boldsymbol{A}$ is the mean spectrum vector of the image and $\boldsymbol{X}$ is the vector representing the brightness values of each pixel. The next step in the PCA analysis is the calculation of the eigenvectors and eigenvalues of the covariance matrix. The eigenvalues $\lambda = \{\lambda_1 \ldots \lambda_k\}$v of a $k \times k$ square matrix is its scalar roots and are given by the solution of the characteristic's equation

$$|\Sigma_x - \lambda \boldsymbol{I}| = 0 \tag{7}$$

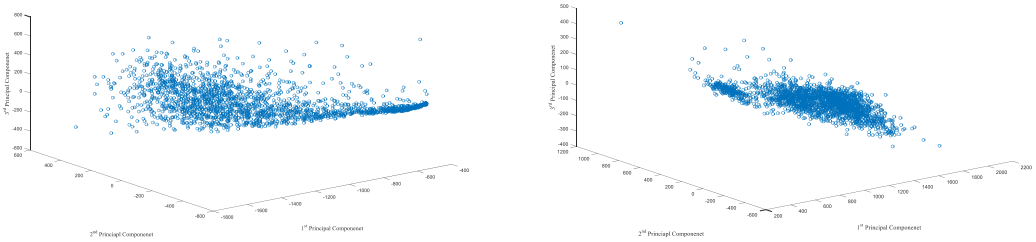
Where $\boldsymbol{I}$ is the identity matrix. The eigenvectors are closely related to the eigenvalues and each one is associated with one eigenvalue. Their length is equal to one and they satisfy the equation

$$\Sigma_x \mathbf{V}_k = \lambda_k \mathbf{V}_k \tag{8}$$

Where $\mathbf{V_k}$ is the eigenvector corresponding to the λ_k eigenvalue and its dimension is 1×k.

The eigenvectors are orthogonal to each other and provide us with information about the patterns of the data. The first eigenvector provides a line that approximates the regression line of the data- this axis is defined by maximizing the variance on this line. Therefore, the second eigenvector provides a line that is orthogonal to the first and contains the variance that is away from the primary vector. Then a regression plane can be defined for the data that maximizes the variance. When more than 3 variables are involved, the principles of maximizing the variance are the same but graphical representation is almost impossible.

The fourth step in the PC analysis is the determination is the components that can be ignored. An important property of the eigenvalue decomposition is that the total variance is equal to the sum of the eigenvalues of the covariance matrix, as each eigenvalue is the variance corresponding to the associated eigenvector. The PC process orders the new data space such that the bands are ordered by variance, from highest to lowest. The eigenvector with the highest eigenvalue is the first principal component (PC) and accounted for most of the variation in an image. The second PC has the second larger variance being orthogonal to the first PC, and so on. Figure. 9 presents the variation of the three principal components for each class that are considered for training and testing the classifier.



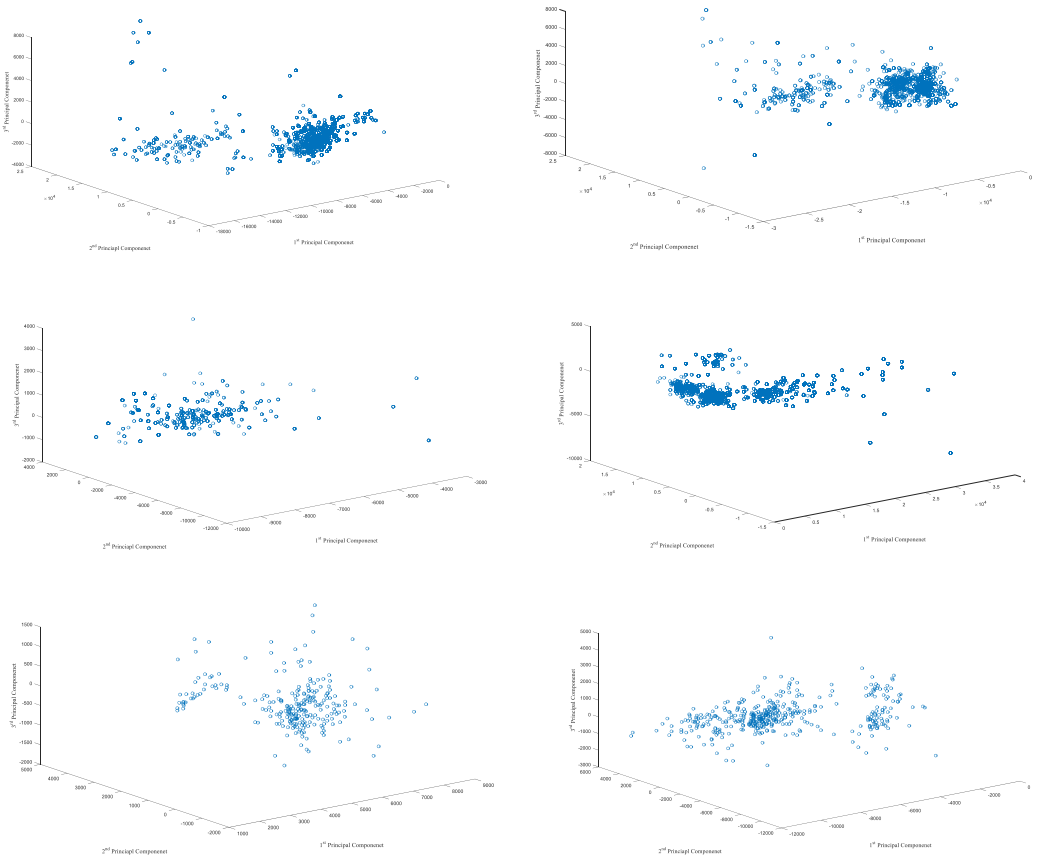


Figure 9. Distribution of the first, second and third principal components of concrete,asphalt, color, crack, dry vegetation, green vegetation, water, and oil dataset.

A transformed data set is created by using the eigenvectors from the diagonalization of the covariance or correlation matrix. After selecting the eigenvectors that should be retained, the following formula is applied:

$$(Final\ Data\ Set) = (Eigenvectors\ Adjusted)' \times (Data\ Adjusted)' \quad (7)$$

Where (*Eigenvector Adjusted*)' is the matrix of eigenvectors transposed so that the eigenvectors are in the row with the first eigenvector on the top and (*Data adjusted*)' is the matrix with the mean-corrected data transposed.

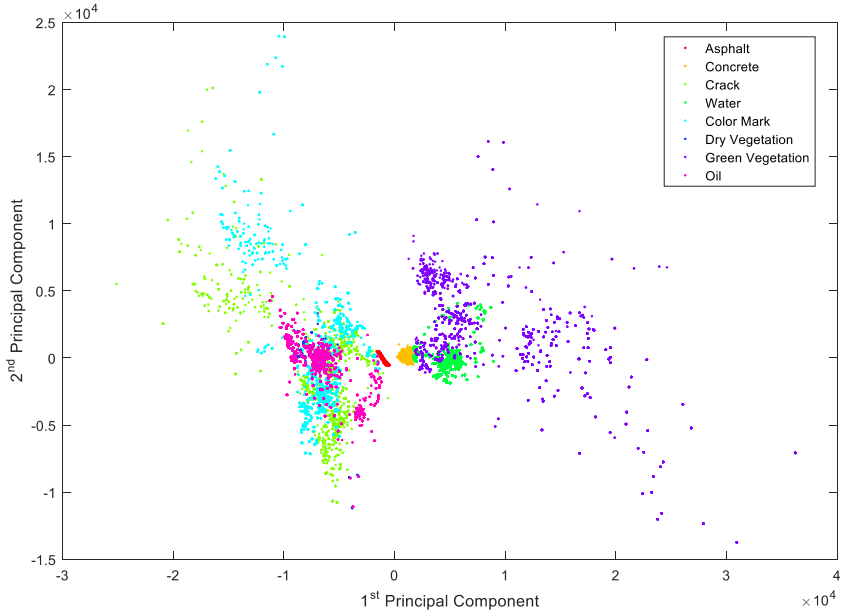


Figure 10. The first and second principal component plot of the dataset.

Histogram of Oriented Gradient (HOG)

The Histogram of Oriented Gradient (HOG) feature descriptor aims to characterize the contextual texture or shape of objects in images through counting the occurrence of gradient orientations in a select block in an image or the whole image. It was first proven effective by Dalal and Triggs (2005) in their seminal effort for pedestrian detection in images (N. Dalal & Triggs, 2005); since then, HOG has been applied extensively for different objective detection tasks in the literature of machine vision. HOG differs from other scale-invariant or histogram-based descriptors in that its extraction is computed over a dense grid of uniformly spaced cells, and it uses overlapping local contrast normalization for improved performance. To this date, there are many variants of HOG descriptors for improving the robustness and accuracy; and a commonly used one is the HOG-UoCTTI as described in (Felzenszwalb, Girshick, McAllester, & Ramanan, 2010).

The basic idea behind HOG is; the appearances and shape of local objects within an image can be well described by the distribution of intensity gradients as the votes for

dominant edge directions. Such a feature descriptor can be obtained by first dividing the image into small contiguous regions of equal size called cells, and collecting a histogram of gradient directions for the pixels within such cells, and hence combining all these obtained histograms from each cell. To improve the detection accuracy against varied illumination and shadowing, local contrast normalization can be applied by computing a measure of the intensities across a larger region of an image, called a block, and using the resultant value to normalize all the cells within the block. Hence HOG consists of gamma and color normalization, gradient and orientation computation, cell histogram computation, normalization across blocks, and flattening into a feature vector. An overview of object detection with HOG is presented in figure 11.

Figure 11: Block diagram for feature extraction with HOG feature descriptors.

The first step of HOG feature extraction is the computation of image gradients. The gradient tells how the image changes in the given direction. Gradient computation is done by applying the 1D centered, point discrete derivative most in both the horizontal and vertical direction while calculating gradient value for each pixel describing the relationship of neighboring pixel values according to the mask. Then, the magnitude and orientation at each pixel $I_{(x, y)}$ is calculated by

$$\begin{cases} G_{mag}(x,y) = \sqrt{G_x^2(x,y) + G_y^2(x,y)} \\ \theta(x,y) = arctan\left(\frac{G_y(x,y)}{G_x(x,y)}\right) + {}^{\pi}/_{2} \end{cases} \quad (9)$$

Where $G_x(x, y)$ and $G_y(x, y)$ are the gradient values at each pixel in the horizontal and vertical direction, respectively.

In the next step, the histogram for each pixel region that is either rectangular or radial is created. The histogram bin is evenly expanded from 0º to 180º for unsigned and 0º

to 360º for signed, so every histogram bin has a spread of 20º. Every pixel in the cell casts weighted voting into one of the 9 histogram bins which can either be the gradient magnitude itself or some function of the magnitude. The voting simply means increasing the frequency of the observed bin by the magnitude of the pixel.

Next, after generating cell histograms, to obtain the robustness against the various illumination and contrast, the gradient strengths must be locally normalized. This can be achieved by grouping the cells into lager pixels regions called bocks. Since the blocks overlap with the neighboring blocks, each block contributes its orientation distribution more than once. Since each scalar cell response contributes several components to the final descriptor vector, each normalized concerning a different block. Overlapping block adds redundant information that can improve the result significantly. There are four variants of the HOG block scheme: Rectangular HOG, Circular HOG, Bar HOG and Center-surround HOG (Navneet Dalal, 2006). (N. Dalal & Triggs, 2005) proposed and compared four different methods for block normalization. Let υ denote the non-normalized feature vector that collects all cell histograms from a given block $||\upsilon||_k$ denotes its k-norm for k = 1, 2 and eps denote some small constant. Then the normalized scheme has the following forms:

$$L2-norm: \quad \hat{\upsilon} = \frac{\upsilon}{\sqrt{||\upsilon||_2^2 + eps^2}} \tag{10}$$

$$L1-norm: \quad \hat{\upsilon} = \frac{\upsilon}{(||\upsilon||_1 + eps)} \tag{11}$$

$$L1-sqrt: \quad \hat{\upsilon} = \sqrt{\frac{\upsilon}{(||\upsilon||_1 + eps)}} \tag{12}$$

L2-Hys is computed by re-normalizing the clipped L2-norm. All the normalization scheme provides much better performance than the non-normalized case. Finally, the HOG feature is the vector containing the elements of the normalized cell histogram from all of the block regions.

In this effort, considering the resolution compatibility between the hyperspectral cube and the companion gray-level image, the feature extraction is conducted in a 20 × 20 sliding-neighborhood in a gray image. Within such a neighborhood four histograms of undirected gradients are averaged to obtain a n_o-dimensional histogram (i.e. binned per their orientation into 9 bins or $n_o = 9$) and a similar operation is performed for the directed gradient to obtain a $2n_o$ dimensional histogram (i.e. binned in accordance of their gradient into 18 bins). Along with both directed and undirected gradient, the HOG-UoCTTI also computes another four-dimensional texture-energy feature. The final descriptor is obtained by stacking the averaged directed histogram, averaged undirected histogram and four normalized factors of the undirected histogram. This leads to the final descriptor of size $4 + 3 \times n_o$ (i.e., a 31 × 1 feature vector).

CHAPTER 5. MACHINE LEARNING APPROACH

Some of the first applications of machine learning to hyperspectral considered the task of classifying land cover, or terrain, into different classes, such as forest, water, agricultural land, and built uplands. Early approach tried to predict the class label c_i at a pixel i from a vector $\boldsymbol{X_i}$ (Benediktsson, Swain, & Ersoy, 1990; Bischof, Schneider, & Pinz, 1992; Paola & Schowengerdt, 1995), with the feature typically just taken to be the values at the different spectral bands at pixels i.

The Bayes' classifier is one of the simplest and most popular approaches to terrain classification. The Bayes' classifier makes explicit assumptions about the class conditional distribution $p(x_i|c_i = k)$ and the prior class probabilities $P(c_i = k)$ and uses Bayes' rule to obtain the posterior class probabilities $P(c_i = k|x_i)$. Various other simplifying assumptions lead to many popular classifiers. For example assuming that Σ_k is diagonal lead to the Naïve Bayes classifier for continuous inputs while assuming that $P(c_i = k) = 1/K$ lead to what is known in the remote sensing literature as the maximum likelihood classifier (Paola & Schowengerdt, 1995).

The main drawback of Bayes' of the Bayes' classifier is the need to explicitly specify the class-condition distribution $p(x_i|c_i = k)$. Since the multivariate normal distribution is typically used for class-conditional distribution, only linear or quadratic decision boundaries can be learned by such a model. The neural network became a popular alternative to the Bayes' classifier because they directly model $p(x_i|c_i = k)$ as a differentiable function whose parameter is learned (Bischof et al., 1992; Lee, Weger, Sengupta, & Welch, 1990). This both sidesteps the need to specify $p(x_i|c_i = k)$ and allows for richer, non-linear decision boundaries to be learned when at least one hidden layer of units with a non-linear activation function is used. Due to the ability to learn non-linear

decision boundaries, neural networks tend to give higher classification accuracies than various forms of Bayes' classifier (Benediktsson et al., 1990; J. Zhang & Modestino, 1989). (Bischof et al., 1992) explored adding contextual information by using spectral values from a small patch at the pixel of interest as the input to a neural network, allowing it to learn some contextual features. Others aimed to improve classification accuracy by using hand-designed features that encoded local textural information. (Haralick, Shanmugam, & Dinstein, 1973; Lee et al., 1990). (Haralick et al., 1973) introduced a popular set of features derived from gray-level values *i*, and *j* co-occur at distance *d* and angle θ.

Support Vector Machine

Discriminating between object classes with similar features, such as concrete, asphalt, vegetation, water, and oil requires some knowledge of spectral profile and context which in turn leads to much more complex decision boundaries than the ones required to discriminate forest and city areas from imagery. Due to the need to learn such highly nonlinear decision boundaries, applications of machine learning to high-resolution imagery have relied on more sophisticated classifiers. While the neural network can learn nonlinear decision boundaries and have been widely used in remote sensing applications, many researchers found them difficult to train due to the presence of local optima (Benediktsson et al., 1990).

Support Vector Machine (SVM) has been employed in a wide range of real-world problems such as text categorization, handwritten digit recognition, tone recognition, object detection, image classification, regression problem and more colloquially learning from examples since purposed by Vapnik (Cortes & Vapnik, 1995). SVM has been proven to be a good candidate for the machine learning approach due to its high

generalization performance without the need for prior knowledge, even when the dimension of the input space is very high. Given a set of points which belongs to either one of two class, a linear SVM finds the hyperplane leaving the largest possible fraction of points of the same class on the same side while maximizing the distance of either class from the hyperplane. According to Vapnik, this hyperplane minimizes the risk of misclassifying data from the test set. SVM has often been found to provide higher classification accuracies than other widely used pattern recognition techniques, such as maximum likelihood (Mondal, Kundu, Chandniha, Shukla, & Mishra, 2012) and the multilayer perceptron neural network classifier (Osowski, Siwek, & Markiewicz, 2004). Furthermore, SVM appears to be especially advantageous in the presence of heterogeneous classes for which only a few training samples are available. In the context of hyperspectral image classification, some pioneering experimental investigations preliminary pointed out the effectiveness of SVM to analyze the hyperspectral data directly in the hyperdimensional feature space, without the need of any feature reduction techniques (J. A. Gualtieri & Chettri, 2000; J. Anthony Gualtieri & Cromp, 1999)
In Figure 12, triangular data points belong to one of the classes and circular data points belong to another class. SVM tries to find a hyper-plane (P1 and P2) that separates the two classes. As shown in the figure there may be many hyperplanes that can separate the data but SVM chooses the best decision boundary based on the maximum margin hyperplane concept.

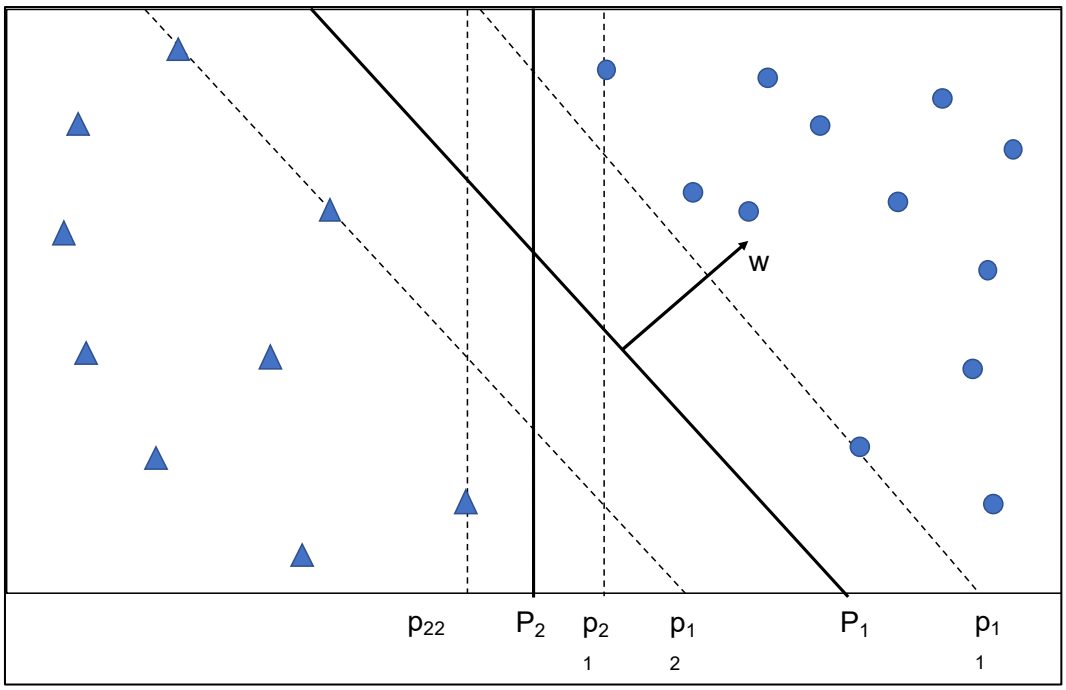


Figure 12. Decision boundary and margin of SVM classifier.

Each hyperplane (P_i) is associated with a pair of supporting hyper-plane (p_{i1} and p_{i2}) that are parallel to the decision boundary (P_i) and pass through the nearest data point. The distance between these supporting planes is called margin. In the figure, even though both the hyperplane (P_1 and P_2) divide the data points, P_1 has a bigger margin and tends to perform better for the classification of unknown samples than P_2. Hence bigger the margin is, less the generalization error for the classification of unknown samples is. Therefore, in the case of the above figure hyperplane P1 is preferred over hyperplane P_2. For a linear SVM, the equation for the decision boundary is

$$\boldsymbol{w} \cdot \boldsymbol{x} + b = 0 \tag{13}$$

Where w and x are vectors and the direction of w is perpendicular to the linear decision boundary. Vector w is determined using the training dataset. For any set of data points (x_i) that lies above the decision boundary the equation is

$$\boldsymbol{w} \cdot \boldsymbol{X_i} + b = k, \ where\ k > 0, \tag{14}$$

And for the data points (x_j) which lie below the decision boundary, the equation is

$$\boldsymbol{w} \cdot \boldsymbol{X_j} + b = k \cdot,\ where\ k < 0, \tag{15}$$

By rescaling the value of w and b the equations of the two supporting hype planes (p_{11} and p_{12}) can be defined as

$$p_{11}: \boldsymbol{w} \cdot \boldsymbol{X} + b = 1 \tag{16}$$

$$p_{12}: \boldsymbol{w} \cdot \boldsymbol{X} + b = -1 \tag{17}$$

The distance between the two hyperplanes (margin "d") is obtained by

$$d = {}^{2}/_{||\boldsymbol{w}||} \tag{18}$$

The objective of the SVM classifier is to maximize the value of *d*. The margin can be seen as a measure of generalization ability: the larger the margin, the better the generalization is expected to be (Palhang, 2009; Vapnik, 1998). This objective equivalent is to minimize the value of $||\mathbf{w}||^2/2$. The value of **w** and b are obtained by solving this quadratic optimization problem under the constraints.

$$\boldsymbol{w} \cdot \boldsymbol{X_i} + b \geq 1\ if\ y_i = 1 \tag{19}$$

$$\boldsymbol{w} \cdot \boldsymbol{X_i} + b \geq -1\ if\ y_i = -1 \tag{20}$$

Where y_i is the class variable for x_i. Imposing these restrictions will make SVM to place the training instances with $y_i = 1$ above the hyperplane p_{11} and the training instances with $y_i = -1$ below the hyperplane p_{12}. The optimization problem can be solved using the Lagrange multiplier method. The objective function to be minimized in the Lagrangian form can be written as:

$$\boldsymbol{L_P} = \frac{1}{2}||\boldsymbol{w}||^2 - \sum_{i=1}^{N} \alpha_i (y_i(\boldsymbol{w} \cdot \boldsymbol{X_i} + b) - \mathbf{1}) \tag{21}$$

α_i are Lagrange multiplier and N are the number of samples. The Lagrange multiplier should be non-negative ($\alpha_i \geq 0$). To minimize the Lagrangian form, its partial derivatives are obtained with respect to **w** and b are equated to zero and the equation is transformed to its dual form.

$$\boldsymbol{L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \alpha_i \alpha_j y_i y_j X_i X_j} \quad (22)$$

The training instances for which the value if $\alpha_i > 0$ lies on the hyperplane p_{11} or h_{12} are called support vectors. Only these training instances are used to obtain the decision boundary parameters **w** and b. Hence the classification of unknown samples is based on the support vectors.

Multi Classification System

In the problem, which is dealt with in this thesis work, the recognition of different civil engineering features, binary classification is of course not sufficient since there are more than two different classes of features. Although being a binary classifier, SVM can be formulated to solve a multi-class classification problem as opted in this research. Thus within SVM, there are two well-known methods: “one versus all” and “pairwise classification” (or “one versus one”) (Duan, Rajapakse, & Nguyen, 2007). The basic idea is to formulate the problem differently: instead of learning “class 1 against class 2 against class 3 and so on…”, the problem can be interpreted as “class 1 against the rest, class 2 against the rest and so on…”.

Kernel Trick

The basic idea with nonlinear SVM is to map training data into higher dimensional features via some mapping Φ(x) and construct a separating hyperplane with maximum margin in the input space. Sometimes, even with the fair amount of slack, linear classification is not possible and thus finding the optimal hyperplane in the higher dimensional feature space is both complicated and computationally expensive. The issue can be handled with a kernel trick. The kernel trick takes all the point and map them into a higher dimensional space. To do so, a kernel K is defined such that two-point x and x’

on the feature vector have a kernel value *K (x, x')*. The mathematical formulation is shown in Equation 21.

$$\boldsymbol{K(x, x') = exp(-\frac{\|x-x'\|^2}{2\sigma^2})} \quad (23)$$

Where || *x-x'*|| is Euclidean distance between two feature vector and $\gamma = \frac{1}{2\sigma^2}$

It can be simply thought of as a transformation of features into infinite-dimensional space, allowing the linear classification which is the basis of SVM. The hyperparameter (C and γ) optimization for this research is achieved by using the Bayesian optimization algorithm which implements the 10-fold cross-validation and iteratively evaluating and updating the promising hyperparameter configuration based on current cross-validation model. In other words, the hyperparameter of the classifier is set by searching the space for the best performance metrics: precision and recall score for each cross-validation model.

Performance Evaluation

To proceed with the modeling and performance evaluation, a data partition strategy is needed to split the data; hence one part is for the training and the other for model validation. In the literature, the widely used scheme is to use 75% of the total data for training and the rest 25% for testing. Indeed, if there are sufficient amount of data, the data splitting ratio is flexible and up to the analyst. In this paper, it is meaningful to examine if the data size is sufficient, which can be reflected if the prediction performance increases with the size of training data. Three data splitting schemes are considered for any of the obtained feature dataset as expressed above. In the first scheme namely Test-1, and by carrying out a random shuffling, 25% (8147) of the data set is considered for training and the rest 75% (26601) for testing purposes. In Test-2, the total dataset is divided equally (i.e., 17374 for training and testing, separately). In Test-3, 75% (26601)

of the dataset is considered for training and the rest 25 % (8147) for testing. With these three schemes, a total of 12 different models are evaluated in this thesis work.

The accuracy of the classifier needs to be defined for estimating and comparing the quality of the classification result. In this effort, due to the presence of a higher number of classes, it is important to properly analyze each parameter as a higher number of classes in general decrease the classification accuracy. To quantify the performance of the classifiers and more importantly their predictive capacity and robustness, two commonly used performance analytics, receiver operating characteristics (ROC) curve, and precision-recall (PR) curve, are adopted, which are constructed by setting a variable decision threshold in the classifier. The area under ROC (AU-ROC) and precision-recall (AU-PR) curve are used as lumped measures summarizing the two curves

Receiver Operating Characteristic Curve (ROC)

Receiver Operating Characteristics Curve (ROC Curve) is a graphical depiction of correctly and incorrectly predicting an outcome condition. ROC curve is plotted on the coordinate system with sensitivity (TPF) values along the y-axis and one minus specificity (FPR) value along the x-axis. Sensitivity and specificity are two measure which can capture model performance. Sensitivity is the percentage of cases in which the outcome (individual class in interest) is correctly predicted. In other words, it is a measure of the proportion of positive classes that are correctly predicted by the model. This statistic is also referred to as the true positive fraction (TPF). Specificity is the percent of cases in which the opposite of the outcome (in the multi-class system, classes belonging to some other class when a particular class is into consideration) is incorrectly predicted, also referred to as true negative fraction (TNF). These measures can be visually combined to characterize the model behavior concerning data called the ROC curve. ROC curve is

also valuable because they permit the comparison of variables and summarize accuracy across a range of tradeoffs between correct and incorrect classification probabilities. ROC curve analysis involves a generally simple graphical representation of classification and is generally used in engineering and imaging to qualify how accurately a detection system can discriminate between binary classes. In practice, the ROC curve analysis evaluates the classification ability of one independent (predictor) variable that is continuously measured and one dependent (outcome) variable that is dichotomously measured.

Area Under Curve (AUC)

(D. M. Green & Swets, 1966) first suggested the area under the ROC curve (ROC-AUC) as an important accuracy index for the measure across all possible decision thresholds. AUC statics is a robust measure because it represents the probability of correct classification across all possible decision thresholds. AUC values of 1.0 indicate perfect classification whereas 0 indicates no accuracy whatsoever for all classes. AUC value of 0.5 corresponds with chance and are presented along the diagonal. Generally, the AUC value above 0.7 indicates the test possesses good accuracy levels. Moreover, any AUC values above 0.5 with a significant F1 score indicate some good ability to discriminate. Since the statistics of the AUC curve are based on the proportion of cases, the result is irrespective of the underlying group size, unlike other classification measures such as accuracy. (Hosmer, Lemeshow, & Sturdivant, March 2013) provided quantitative guidance for the labeling of discrimination ability that this will consolidate and adopt: 0.5 (no ability), ≥0.60 (low ability), ≥ (0.70) (accepted ability), ≥0.80 (excellent ability), ≥0.90 (outstanding ability), and 1.0 (perfect ability).

Confusion Matrix

In the problem of statistical classification, confusion matric is also called an error matrix having a specific layout allowing a user to visualize the performance of a supervised classifier algorithm. The confusion matrix is meant to visualize the per class prediction performance of the chosen model. Each row of the matrix represents the instance for the actual class whereas each column represents the instances for the predicted class. Hence, we can infer the correctly classified points are grouped corresponding to the classes in the diagonal entries of the confusion matrix. A sample M_{ij} here $i=j$ indicates the true class and the predicted class are the same thus representing an accurate classification (diagonal position). A sample that goes into M_{ij} where $i \neq j$ indicates that the true class i and the predicted class j are not the same thus representing a misclassification.

Precision-Recall Curve (PRC Curve)

Precision and recall are the matrices that give us a picture of the model performance and can be evaluated from the confusion matrix. Precision is a measure of how many of the predicted values in a class is correctly classified are part of the true label of the class. Hence it is the measure of the positive prediction by the model. Recall on the other hand is the measure of the amount of information correctly retrieved or in other words, the number of samples correctly predicted. Models can be optimized on a measure that combines or balances both precision and recall. This is called F measure which is a weighted average of the precision and recall of the model. Precision and recall can be computed from the confusion matrix to determine the model performance using the following formulations.

$$Precision = \frac{TP}{TP+FP} \tag{24}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \tag{25}$$

This concept can be extended the multi-class cases for the precision and recall formulation. If M represents a confusion matrix for multiple classes, M being a $k \times k$ matrix where k is the number of classes.

$$Precision = \frac{M_{ii}}{\sum_j M_{ji}} \tag{26}$$

$$Recall = M_{ii} / \sum_i M_{ij} \tag{27}$$

AUC for the precision-recall curve is calculated as the area under the precision-recall curve, where each point on the curve is defined by different values of the threshold to convert continuous binary predictions. Unlike ROC, AUC-PR does not depend on the number of true absent observations. While the AUC-ROC curve is maximized by a curve in the upper left-hand corner, AUC-PR is maximized in the upper right-hand corner, reflecting the norm of placing sensitivity on the y-axis in a receiver operating characteristics curve but the x-axis in a precision-recall curve. AUC-PR varies on a scale from zero to one, with random performance equal to sample prevalence in the focal dataset

Results

Test 1: Comparison between model M1(HYP) and model M2(HYP_PCA)

We first examined the performance of the model M1(HYP) (i.e. classifier trained on the full spectral dataset) and model M2(HYP_PCA) (i.e. the classifier trained on the principal components of the spectral dataset). This test aims to endorse the hypothesis that dimensionality reduction does increase the performance of the classifier. The parameters acquired from the confusion matrix for each case both models M1(HYP) and M2(HYP_PCA) are summarized in Table 1 and Table 2 respectively.

Table 1: Performance summary of model M1(HYP).

		Concrete	Asphalt	Color	Crack	Dry Vegetation	Green Vegetation	Water	Oil
Test-I	Precision	0.998	1.000	0.598	0.414	0.488	0.776	0.743	0.612
	Recall	0.999	0.998	0.531	0.130	0.164	0.705	0.946	0.717
	F-Score	0.999	0.999	0.563	0.198	0.246	0.739	0.832	0.661
	AU-ROC	1.000	1.000	0.889	0.783	0.915	0.947	0.965	0.909
	AU-PR	1.000	1.000	0.551	0.237	0.265	0.811	0.830	0.638
Test-II	Precision	1.000	1.000	0.611	0.567	0.558	0.818	0.787	0.628
	Recall	0.999	1.000	0.655	0.167	0.192	0.704	0.957	0.702
	F-Score	1.000	1.000	0.632	0.257	0.286	0.756	0.864	0.663
	AU-ROC	1.000	1.000	0.906	0.808	0.921	0.952	0.971	0.912
	AU-PR	0.999	0.999	0.592	0.312	0.290	0.800	0.845	0.642
Test-III	Precision	1.000	1.000	0.501	0.918	0.402	0.706	0.572	0.553
	Recall	1.000	1.000	0.629	0.057	0.213	0.733	0.962	0.724
	F-Score	1.000	1.000	0.558	0.107	0.278	0.719	0.717	0.627
	AU-ROC	1.000	1.000	0.869	0.774	0.906	0.940	0.963	0.890
	AU-PR	0.999	0.999	0.490	0.495	0.234	0.745	0.685	0.586

Table 2: Performance summary for model M2(HYP_PCA)

		Concrete	Asphalt	Color	Crack	Dry Vegetation	Green Vegetation	Water	Oil
Test-I	Precision	0.930	0.901	0.957	0.914	0.878	0.986	0.973	0.955
	Recall	0.886	0.910	0.964	0.921	0.664	0.979	0.992	0.974
	F-Score	0.907	0.906	0.961	0.917	0.757	0.982	0.983	0.964
	AU-ROC	0.993	0.995	0.997	0.995	0.985	0.999	0.999	0.998

	AU-PR	0.957	0.960	0.977	0.960	0.839	0.993	0.985	0.982
Test-II	Precision	0.970	0.936	0.973	0.973	0.900	0.990	0.994	0.979
	Recall	0.959	0.945	0.989	0.952	0.808	0.994	0.995	0.981
	F-Score	0.964	0.941	0.981	0.962	0.851	0.992	0.994	0.980
	AU-ROC	0.998	0.998	0.999	0.997	0.993	1.000	1.000	0.998
	AU-PR	0.987	0.983	0.985	0.987	0.893	0.995	0.997	0.985
Test-III	Precision	0.987	0.942	0.984	0.971	0.908	0.989	0.988	0.992
	Recall	0.955	0.954	0.991	0.950	0.867	0.997	0.996	0.995
	F-Score	0.971	0.948	0.988	0.960	0.887	0.993	0.992	0.993
	AU-ROC	0.998	0.997	0.999	0.997	0.994	1.000	1.000	1.000
	AU-PR	0.981	0.981	0.993	0.985	0.886	0.996	0.993	0.994

Firstly, from the F score of the model M1(HYP) based on Table 1, one can see that the model identifies successfully (with F1 > 0.7) on the plain concrete, asphalt, green vegetation, and water. For the color marking and oil, it is around 0.6. However, for cracks and dry vegetation, the F1 measurements are lower than 0.3, indicating comparatively lower prediction accuracy. First, this reflects the challenge in recognizing cracks primarily caused by its inherent spectral complexity. Second, it is primarily due to the smaller size of the dry-vegetation data points. Last and for all classes, this is presumptively attributed to the high dimensionality of the feature vectors of the hyperspectral data set. Nonetheless, as one expects from the AU-ROC measurements, as shown in Table 1, there is no doubt that the model M1(HYP) is highly effective in recognizing these structural surface objects. At the class label of concrete, the average AU-ROC is1. Therefore, it is stated herein that the hyperspectral pixels as feature vectors are effective in recognizing most of the structural surface objects, and have relatively less accuracy only in the detection of cracks and dry vegetation. The area under the ROC curve for all cases in Test 1 showcases good

discriminative power of the classifier for the provided dataset. With all that stated, in some cases, the visual representation from ROC plots can be deceptive when there is an imbalance in the dataset as presented in this thesis work. Hence for evaluation of the performance of the classifier, along with values of ROC curves, the area under the precision-recall curve is also considered.

By observing the accuracy of the model M2(HYP_PCA) compared to M1(HYP) in Table 1 and Table 2, it is desirable to observe that the detection accuracy is significantly increased. At the two weak prediction instances of cracks and dry vegetation pertinent to M1(HYP), the F1 score arises from 0.198 to 0.917 at Test-1 and 0.107 to 0.96 at Test-3, when M2(HYP_PCA) is tested. For dry vegetation, the F1 scores changes from 0.246 to 0.757 at Test-1 and 0.278 to 0.887 at Test-3. At all other class labels, the classification accuracy still mounts up from their initially high F1 values from M1 to M2. When the AU-ROC is concerned, besides that mostly they increase from M1 to M2, the measurements are all greater than 0.99 at predicting all class labels. This again signifies that the underlying model, M2(HYP_PCA), has nearly perfect capacity towards detecting all structural surface objects. The comparative tests herein provide the direct evidence that performing dimensionality reduction over hyperspectral profiles (i.e., as HYP feature vectors) can substantially unleash the embedded discrimination capacity of the data that is otherwise not exploitable.

Test 2: Comparison between model M2(HYP_PCA) and model M3(GL_HOG)

The secondary yet important hypothesis in this thesis work is to prove the effectiveness of low spatial-resolution hyperspectral data when compared to high-resolution gray-intensity images towards detecting the structural surface object. For this purpose, the results of M2(HYP_PCA) and M3(GL_HOG) are evaluated and compared.

ROC and PR curves for all the classes of model M2(HYP_PCA) and M3 (GL_HOG) for Test I are shown in Figure 13 and Figure 14 respectively. Both the ROC and PR curves for model M2(HYP_PCA) illustrate a significant discriminative ability when compared to the performance of model M3(GL_HOG).

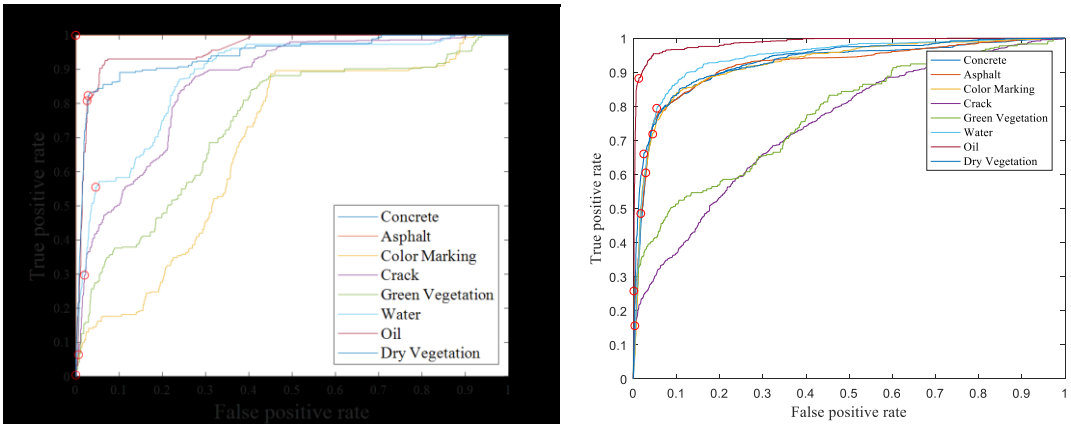


Figure 13: ROC curves for model M2(HYP_PCA) and M3(GL_HOG) (Test-I).

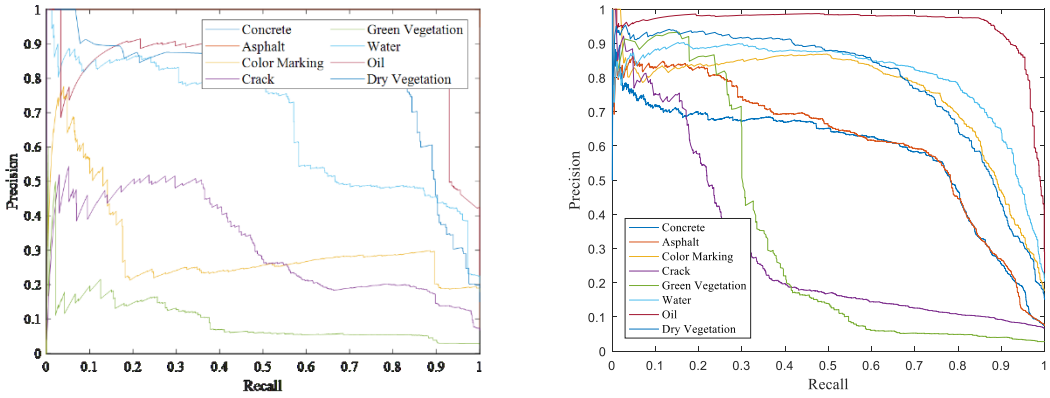


Figure 14: Precision-Recall curves for model M2(HYP_PCA) and M3(GL_HOG) (Test-I).

The confusion matrix both models M2(HYP_PCA) and M3(GL_HOG) are presented in Table 3 and Table 4 respectively.

Table 3: Confusion matrix for model M2(HYP_PCA) (Test-I)

Actual Class								
Concrete	1728	0	0	0	0	56	166	0
Asphalt	0	1775	40	29	20	0	0	86
Artificial Mark	0	27	4588	87	36	0	0	19
Crack	0	32	76	1641	8	0	0	25
Dry Vegetation	0	67	39	17	477	0	0	118
Green Vegetation	83	0	0	0	0	3901	0	0
Water	48	0	0	0	0	0	6051	0
Oil	0	69	50	22	2	0	0	5268
	Concrete	**Asphalt**	**Artificial Mark**	**Crack**	**Dry Vegetation**	**Green Vegetation**	**Water**	**Oil**
	Predicted Class							

Table 4: Confusion matrix for model M3(GL_HOG) (Test-I).

Actual Class								
Concrete	1090	499	98	13	0	63	31	156
Asphalt	495	1167	136	16	0	53	20	63
Artificial Mark	106	31	3714	114	6	313	111	362
Crack	39	84	566	348	5	297	170	273
Dry Vegetation	14	3	10	3	168	67	9	444
Green Vegetation	63	88	182	54	13	2913	315	356
Water	47	70	47	31	0	72	5752	80
Oil	92	71	384	26	40	157	71	4570
	Concrete	**Asphalt**	**Artificial Mark**	**Crack**	**Dry Vegetation**	**Green Vegetation**	**Water**	**Oil**
	Predicted Class							

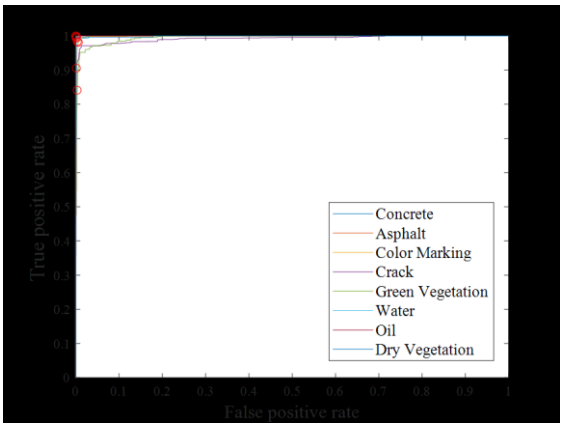
Similarly, the performance table model M3(GL_HOG) is presented in Table 5.

Table 5: Performance summary for model M3(GL_HOG).

		Concrete	Asphalt	Color	Crack	Dry Vegetation	Green Vegetation	Water	Oil
Test-I	Precision	0.560	0.580	0.723	0.575	0.724	0.740	0.888	0.725
	Recall	0.559	0.598	0.781	0.195	0.234	0.731	0.943	0.845
	F-Score	0.560	0.589	0.751	0.292	0.354	0.736	0.915	0.780
	AU-ROC	0.940	0.950	0.926	0.774	0.898	0.929	0.985	0.944
	AU-PR	0.549	0.615	0.724	0.295	0.365	0.750	0.932	0.764
Test-II	Precision	0.638	0.690	0.752	0.676	0.827	0.786	0.929	0.785
	Recall	0.669	0.637	0.827	0.273	0.290	0.801	0.964	0.903
	F-Score	0.653	0.662	0.788	0.389	0.430	0.793	0.946	0.840
	AU-ROC	0.963	0.966	0.948	0.833	0.931	0.958	0.991	0.962
	AU-PR	0.647	0.701	0.756	0.399	0.466	0.796	0.954	0.831
Test-III	Precision	0.665	0.690	0.772	0.644	0.886	0.801	0.882	0.802
	Recall	0.639	0.679	0.847	0.301	0.325	0.809	0.964	0.906
	F-Score	0.652	0.684	0.808	0.410	0.476	0.805	0.921	0.851
	AU-ROC	0.959	0.969	0.942	0.828	0.948	0.962	0.990	0.964
	AU-PR	0.666	0.737	0.739	0.424	0.529	0.816	0.921	0.850

Even if the evaluation is done based on the area under the ROC curve, which is usually the case in most of the classification work, the outcome suggests both the classifier performs satisfactorily. Yet other scores, derived from the confusion matrix provided more information particular for the faulty class of interest. As shown in Table 1, the AU-ROC curves for all classes of model M2(HYP_PCA) are greater than the value 0.95 with perfect result 1 for the asphalt and concrete. Similarly, the AU-PR curves follow the same trend for all the classes with an area above 0.9 except for the dry vegetation that has a value of 0.632. The highest AU-PR curve value is 1 which is for

both asphalt and concrete but the class artificial color, crack, and dry vegetation are underperforming with AU-PR value of 0.306, 0.316, and 0.092 respectively. Table 5 summarizes the values of AU-ROC and AU-PR curves for the model M3(GL_HOG). Unlike from Table 1, the AU- ROC curve for class water has the highest value of 0.986, and the rest of the classes have an AU-ROC curve value of around 0.90 except for the crack and dry vegetation with AUC-ROC 0.748 and 0.7758 respectively. The model M3(GL_HOG) gives AUC-PR of 0.9526 for the class water, however, for the rest of the classes, lower values are observed i.e. 0.2892, 0.3279, 0.5779 and 0.6152 for crack, dry vegetation, concrete, and asphalt respectively. This shows that with lower training instances, there are losses in classification accuracy when a classifier is trained either with gray features or principal components of hyperspectral features. Table 5 also illustrates that precision, recall, and, F score for the classifier trained with HOG features are not consistent throughout for all the classes compared to result obtained from the model M2. In Test-II where the equal number of training and testing instances are considered, it is expected to improve results for both the classifier. The ROC curve for model M2 presented in Figure 15 clearly reflects the increase in performance accuracy. The ROC and PR curves obtained for model M2 (HYP_PCA) and model M3(GL_HOG) for Test-II are presented in Figure 15 and Figure 16 respectively.



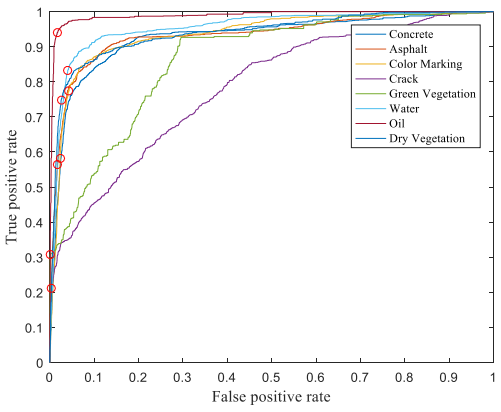


Figure 15: ROC curve for model M2 (HYP_PCA) and M3(GL_HOG) (Test-II).

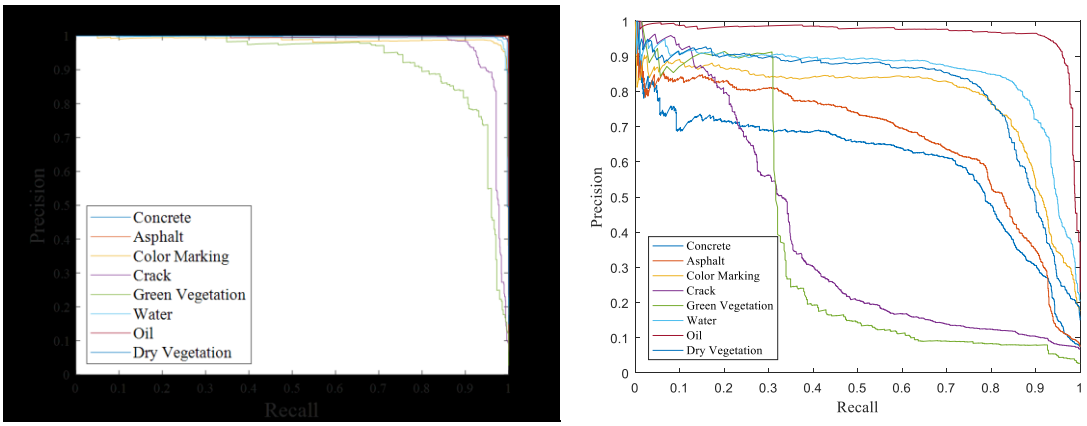


Figure 16: Precision-Recall curve for model M2 (HYP_PCA) and M3(GL_HOG) (Test-II).

An improved and optimum result for model M2(HYP_PCA) is reflected by the area under the precision-recall curve for all the classes. On the other hand, there aren't any significant improvement in the classifier accuracy using a higher number of features for the model M3(GL_HOG). The area under the ROC curve for each class classified using gray level features seems promising but is not backed by the area under the precision-recall curves as shown in Figure15. Table 6 and Table 7 presents the confusion matrix for each classification result.

Table 6: Confusion matrix for model M2(HYP_PCA) (Test-II).

Actual Class								
Concrete	1248	0	0	0	0	27	26	0
Asphalt	0	1230	8	12	15	0	0	36
Artificial Mark	0	10	3101	12	11	0	0	3
Crack	0	29	22	1132	3	0	0	3

Dry Vegetation	0	31	21	5	387	0	0	35
Green Vegetation	17	0	0	0	0	2639	0	0
Water	22	0	0	0	0	0	4044	0
Oil	0	14	36	3	14	0	0	3540
	Concrete	**Asphalt**	**Artificial Mark**	**Crack**	**Dry Vegetation**	**Green Vegetation**	**Water**	**Oil**
				Predicted Class				

Table 7: Confusion matrix for model M3(GL_HOG) (Test-II).

Actual Class								
Concrete	870	288	46	9	0	17	9	62
Asphalt	370	829	52	9	1	23	3	14
Artificial Mark	42	13	2594	66	1	218	31	172
Crack	21	27	429	325	2	183	69	133
Dry Vegetation	6	0	4	4	139	9	0	317
Green Vegetation	26	22	82	26	15	2127	185	173
Water	14	2	23	12	0	76	3920	19
Oil	15	21	219	30	10	53	2	3257
	Concrete	**Asphalt**	**Artificial Mark**	**Crack**	**Dry Vegetation**	**Green Vegetation**	**Water**	**Oil**
				Predicted Class				

Test-III with the most training instances among all three cases is expected to demonstrate an optimum classification efficiency among all the cases considered. The ROC and the PR curve are shown below in Figure 17 and Figure 18, follows the trend as

for the previous two cases and reflect a better discriminative ability for model M2(HYP_PCA) rather than model M3(GL_HOG).

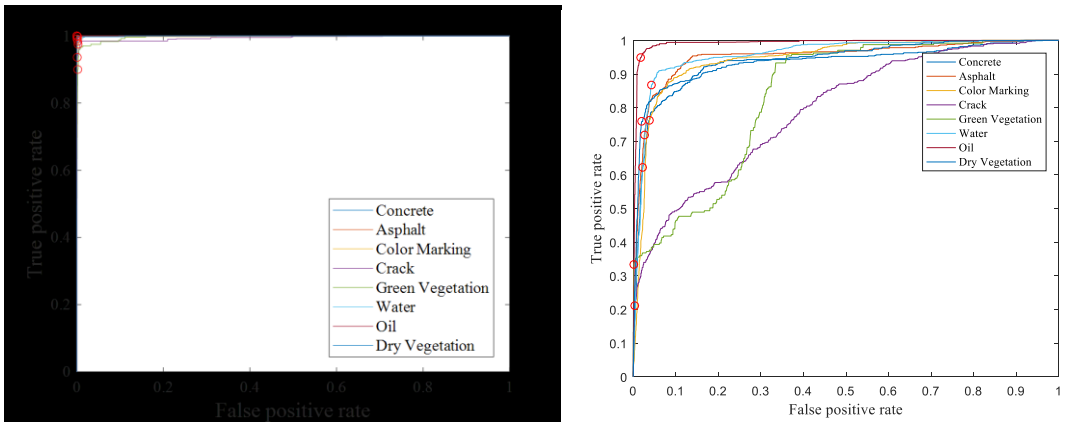


Figure 17: ROC curve for model M2 (HYP_PCA) and M3(GL_HOG) (Test-III).

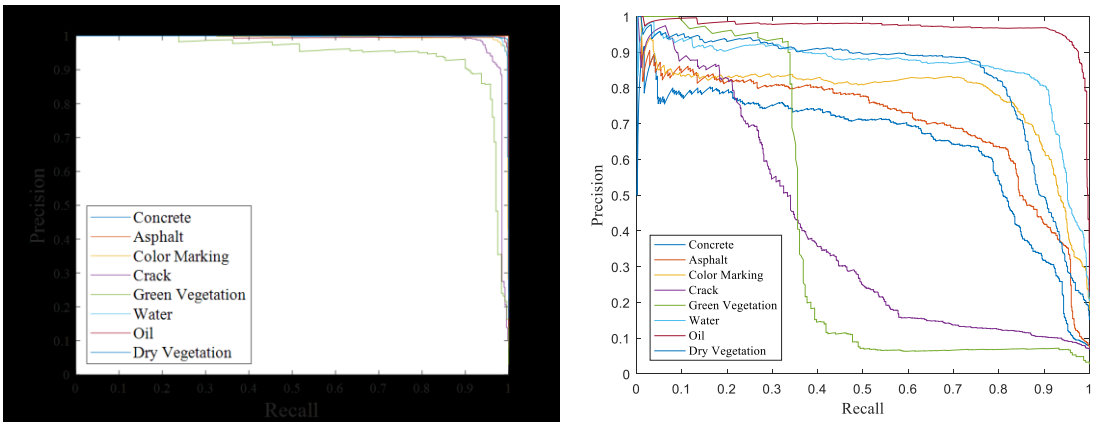


Figure 18: Precision recall curve for model M2 (HYP_PCA) and M3(GL_HOG) (Test-III).

Classification accuracy represented by the area under the ROC curve for each class in model M2(HYP_PCA) are all in the range of 0.99. For the class concrete and asphalt, the classifier performs to its fullest as reflected by the area under the ROC curve of 1 for both of the classes. The area under the precision-recall curve and the ROC curve for the classifier trained with hyperspectral data are uniforms. This reflects the exceptional performance model M2(HYP_PCA). Even though there a certain increment is evident for the classification accuracy of model M3(GL_HOG), the increase is not

significant and as well is not backed by the area under the precision-recall curve. The confusion matrix for the classifier result trained with spectral features and HOG features are presented below in Table 8 and Table 9 respectively. The overall summary for models M2(HYP_PCA) and model M3(GL_HOG) is presented in Table 1 and Table 5.

Table 8: Confusion matrix for model M2(HYP_PCA) (Test-III).

Actual Class								
Concrete	622	0	0	0	0	15	14	0
Asphalt	0	621	6	5	12	0	0	7
Artificial Mark	0	5	1519	4	5	0	0	0
Crack	0	20	7	565	3	0	0	0
Dry Vegetation	0	12	6	6	208	0	0	8
Green Vegetation	4	0	0	0	0	1324	0	0
Water	4	0	0	0	0	0	1129	0
Oil	0	1	5	2	1	0	0	1794
	Concrete	**Asphalt**	**Artificial Mark**	**Crack**	**Dry Vegetation**	**Green Vegetation**	**Water**	**Oil**
				Predicted Class				

Table 9: Confusion matrix for model M3(GL_HOG) (Test-III).

Actual Class								
Concrete	416	174	16	8	0	14	2	21
Asphalt	166	442	19	8	0	4	2	10
Artificial Mark	17	6	1329	45	0	109	10	53
Crack	11	11	194	179	1	93	28	78

Dry Vegetation	0	1	3	0	78	6	0	152
Green Vegetation	2	4	34	16	9	1075	104	84
Water	1	3	6	3	0	23	1092	5
Oil	13	0	120	19	0	18	0	1633
	Concrete	**Asphalt**	**Artificial Mark**	**Crack**	**Dry Vegetation**	**Green Vegetation**	**Water**	**Oil**
	Predicted Class							

From the F1 scores and the AU-ROC measurements shown in Figures 13, 15, and 17, respectively, a straightforward observation is clear that with the prediction of all class labels, M2(HYP_PCA) supersedes M3(GL_HOG). Even considering the fourth model M4 (HYP_PCA+GL_HOG) to be evaluated, the select model picked from all the models is M2(HYP_PCA) at Test 3 This model provides an outstanding performance: the smallest F1 score is 0.84 and the smallest AU_ROC measurement is 0.995, both at predicting dry vegetation; and F1 = 0.93, AU_ROC = 0.99 at the crack prediction. The control model at this data case, M3(GL_HOG), gives rise to F1 = 0.49, AU_ROC = 0.84 for dry vegetation; and AU_ROC = 0.79 F1 = 0.37, AU_ROC = 0.79 for cracks. The performance of both the model appears to be comparable in ROC space, however, in PR space model M2 (HYP_PCA) has a clear advantage over the other. One can observe that for the model M2(HYP_PCA), their ROC and PR curves reflect that the resulting models have not only superb classification capability but also stronger stability, the latter of which is seen from the smoothness of the curves as the underlying threshold varies. In the case of the M3(GL_HOG), the classification capability (and accuracy) are overall much moderate; in addition, the stability as well is worsened.

Test 3: Performance of model M4(HYP_PCA+GL_HOG)

Finally, in Test 3 the performance of the model trained with combined features from PC features of spectral data and high-resolution grayscale HOG feature is tested. A summary of the results for the test is presented in Table 10. As the observed tendency from the previous two tests, the classifier reflects an exceptional result for the area under the ROC curve. The class dry vegetation with the area under the ROC curve of 0.683 is the least among all the considered classes. Similarly, the values obtained from the area under the precision-recall curve reflects a supporting conclusion. Similar to the result for the AU-ROC, the result obtained for the area under the PR curve area also above 0.95 for most of the classes. This shows a promising and consistent inconsistent result for the classifier with a multi class problem. Table 10 also presents results for precision, recall, and F1-score for the classifier trained with PCA-Hyperspectral and HOG-gray features.

Table 10: Performance summary for model M4(HYP_PCA+GL_HOG)

		Concrete	**Asphalt**	**Color**	**Crack**	**Dry Vegetation**	**Green Vegetation**	**Water**	**Oil**
Test-I	Precision	0.520	0.842	0.904	0.785	0.489	0.905	0.956	0.890
	Recall	0.965	0.881	0.921	0.848	0.341	0.927	0.679	0.873
	F-Score	0.676	0.861	0.913	0.815	0.402	0.916	0.794	0.881
	AU-ROC	0.978	0.992	0.993	0.985	0.938	0.993	0.974	0.986
	AU-PR	0.695	0.933	0.963	0.898	0.391	0.964	0.930	0.930
Test-Ii	Precision	0.450	0.829	0.922	0.841	0.670	0.875	0.930	0.903
	Recall	0.963	0.888	0.931	0.876	0.436	0.910	0.567	0.902
	F-Score	0.613	0.857	0.927	0.858	0.528	0.892	0.704	0.902
	AU-ROC	0.969	0.993	0.994	0.990	0.957	0.990	0.961	0.988
	AU-PR	0.592	0.928	0.966	0.931	0.583	0.947	0.888	0.938
Test-III	Precision	0.779	0.761	0.931	0.895	0.779	0.947	0.969	0.925
	Recall	0.939	0.962	0.935	0.835	0.529	0.951	0.851	0.894
	F-Score	0.852	0.849	0.933	0.864	0.630	0.949	0.906	0.909
	AU-ROC	0.995	0.995	0.994	0.989	0.962	0.998	0.997	0.989
	AU-PR	0.946	0.957	0.970	0.924	0.683	0.985	0.981	0.944

Computational Cost

Training time and testing are important factors for assessing the classification models as well as for the selection of the best model for a specific classification task. For this thesis, the training and testing times for the four types of models in consideration with the three data schemes has been noted and presented in Figure 19. It is evident from the plots that as the number of data points increasing from Test-1 to 3, the training tends to consume more computation time and similarly with decrease in testing instances from Test-1 to 3 the classification process tends to consume less computational time. Second, the dimensionality and complexity of features significantly affects the training time. As evident form figure 19, the feature type GL_HOG demands more training time than any other models, whereas the differences in training M1, M3, and M4 models are relatively insignificant. Similarly, comparing the testing time for all models, it is evident that with better training of the model, testing time can be significantly reduced. This implies that when M2, M3, and M4 are candidates for choosing an optimal model, the prediction performance may be the dominant factor without weighing the training cost in the balance for model selection.

Figure 19: Training time comparison plot.

CHAPTER 6. DISCUSSION

Two arguments are discussed herein. Over the observed superior performance of M2(HYP_PCA) compared to M3(GL_HOG), it implies that a single hyperspectral pixel is much more effective than a 20 × 20 neighborhood of gray values in discriminating the complex structural surface objects. The authors state that hyperspectral pixels with reflectance features at both visible and infrared bands, once preprocessed (e.g., a PCA based feature selection step), outperform the gray values or the embedded texture/shape features corresponding to the hyperspectral pixel, even though the gray images are captured at a much higher resolution (20 times high). Although improved gray-values based feature extraction techniques can be used, it is safe to state that hyperspectral pixels (being captured in the visible to near-infrared spectral bands with a high dimensionality) have undoubtful promise in the detection of complex structural surface objects. Regarding the performance of M4 using the simple data fusion method in this thesis, the authors assert that an improved future fusion approach should be used rather than the concatenation method. A promising approach is to extract spectral-spatial features using a more integral approach; as mentioned earlier, through a pan-sharpening technique, one can generate a high spatial-resolution hyperspectral cube. In this paper, this may give out a cube data as $h(u, v, s)$ defined in a 1000 × 1000 grid with a spectral dimension number of 139, namely in a 1000 × 1000 × 139 cube. With this 3D cube, and the mask image of the same size, advanced spatial-spectral features may be extracted (Fang, He, Li, Plaza, & Plaza, 2018; Hang, Liu, Song, & Sun, 2015; Q. Zhang, Tian, Yang, & Pan, 2014), then a classifier (e.g., a kernel SVM as used in this paper) can be adopted. On the other hand, the latest deep learning architectures may be exploited as reviewed earlier. These promising directions are beyond the scope of this work

Conclusion

A mobile hyperspectral imaging system is developed in this paper ready for ground-based and aerial data collection. One of its primary application is to inspect structural surfaces of concrete or asphalt materials that are commonly used transportation structures. The innovation lies in its capability of detecting structural surface damage and other surface artifacts at the material levels thanks to its high-dimensional pixels with reflectance at both visible and near-infrared bands. This paper hence primarily aims to prove its effectiveness compared to regular gray-level images that are much high-resolution and commonly used in practice. Towards this goal, four different class cation models that are characterized by different feature extraction processes are trained and tested in this paper. With a total of 34,748 labeled features of different types, three data splitting schemes are used to evaluate the effects of data sizes. A multi-class support vector machine with a Gaussian kernel is adopted in all models. While testing the models, state-of-the-art measures are adopted and the issue of data unbalancing is considered. The F1 measure is employed as the primary accuracy measure, and the ROC-derived measure, AU-ROC, is considered as a primary model capacity measure. With a comprehensive evaluation, two major conclusions are formulated.

1) Hyperspectral pixels of reflectance in the VNIR domain as features are very effective in recognizing all of the eight structural surface objects. Nonetheless, dimensionality reduction is essential for this effectiveness. The linear PCA approach is adopted; through using only the first three-component scores, the resulting classification models are highly accurate, high-capacity, and stable.
2) When compared to the model based on gray-level features (GL_HOG, based on a popular variant of HOG descriptor), the PCA adapted hyperspectral features manifest

a much more compelling detection performance than GL_HOG. The author state that a single hyperspectral pixel with high-dimension spectral reflectance is evidentially competitive compared to the corresponding high-resolution gray intensities that express the shape and texture of the underlying objects. The data fusion technique in this paper has a less desirable performance, however. This is attributed to the simple concatenation technique that is used to combine the GL_HOG and HYP_PCA features.

With this experimental and machine learning-based evaluation results in this paper, the authors further envision the dawn of computational hyperspectral imaging or hyperspectral machine vision for structural damage detection in civil engineering and their promise in dealing with complex structural scenes in practice.

REFERENCE LIST

Abdel-Qader, I., Abudayyeh, O., & Kelly Michael, E. (2003). Analysis of edge-detection techniques for crack identification in bridges. *Journal of Computing in Civil Engineering, 17*(4), 255-263. doi:10.1061/(ASCE)0887-3801(2003)17:4(255)

Adhikari, R. S., Bagchi, A., & Moselhi, O. (2014). Automated condition assessment of concrete bridges with digital imaging. *Smart Structures and Systems, 13*, 901-925. doi:10.12989/sss.2014.13.6.901

Adler-Golden, S., Berk, A., Bernstein, L., Richtsmeier, S., Acharya, P., Matthew, M., . . . Chetwynd, J. (1998). *FLAASH, a MODTRAN4 atmospheric correction package for hyperspectral data retrievals and simulations.* Paper presented at the Summaries of the Seventh JPL Airborne Earth Science Workshop.

Alipour, M., Harris, D. K., & Miller, G. R. (2019). Robust pixel-level crack detection using deep fully convolutional neural networks. *Journal of Computing in Civil Engineering, 33*(6), 04019040.

ASCE. (2017). Infrastructure report card. Retrieved from https://www.infrastructurereportcard.org/

Benediktsson, J. A., Swain, P. H., & Ersoy, O. K. (1990). Neural network approaches versus statistical methods in classification of multisource remote sensing data. *IEEE Transactions on Geoscience and Remote Sensing, 28*(4), 540-552. doi:10.1109/TGRS.1990.572944

Bischof, H., Schneider, W., & Pinz, A. J. (1992). Multispectral classification of Landsat-images using neural networks. *IEEE Transactions on Geoscience and Remote Sensing, 30*(3), 482-490. doi:10.1109/36.142926

Bodkin, A., Sheinis, A., Norton, A., Daly, J., Beaven, S., & Weinheimer, J. (2009a). Snapshot hyperspectral imaging: The hyperpixel array camera. *Proceedings of SPIE - The International Society for Optical Engineering*. doi:10.1117/12.818929

Bodkin, A., Sheinis, A., Norton, A., Daly, J., Beaven, S., & Weinheimer, J. (2009b). *Snapshot hyperspectral imaging: The hyperpixel array camera.* Paper presented at the Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery XV.

Cawley, P. (2018). Structural health monitoring: Closing the gap between research and industrial deployment. *Structural Health Monitoring, 17*(5), 1225-1244. doi:10.1177/1475921717750047

Cha, Y.-J., Choi, W., & Büyüköztürk, O. (2017). Deep learning-based crack damage detection using convolutional neural networks. *Computer-Aided Civil and Infrastructure Engineering, 32*(5), 361-378. doi:10.1111/mice.12263

Chen, Z., Chen, J., Shen, F., & Lee, Y. (2015). Collaborative mobile-cloud computing for civil infrastructure condition inspection. *Journal of Computing in Civil Engineering, 29*(5), 04014066.

Chen, Z., Derakhshani, R., Halmen, C., & Kevern, J. T. (2011). *A texture-based method for classifying cracked concrete surfaces from digital images using neural networks.* Paper presented at the The 2011 International Joint Conference on Neural Networks, San Jose, CA, USA.

Chen, Z., & Hutchinson, T. C. (2010). Image-based framework for concrete surface crack monitoring and quantification. *Advances in Civil Engineering, 2010*, 18. doi:10.1155/2010/215295

Cheng, H. D., & Miyojim, M. (1998). Automatic pavement distress detection system. *Information Sciences, 108*(1), 219-240. doi:https://doi.org/10.1016/S0020-0255(97)10062-7

Cheng, H. D., Shi, X. J., & Glazier, C. (2003). Real-time image thresholding based on sample space reduction and interpolation approach. *Journal of Computing in Civil Engineering, 17*(4), 264-272. doi:10.1061/(ASCE)0887-3801(2003)17:4(264)

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning, 20*(3), 273-297. doi:10.1007/BF00994018

Cubert Gmbh. (2018). Real-time hyperspectral imaging. Retrieved from https://cubert-gmbh.com/

Dalal, N. (2006). Finding people iniImages and videos.

Dalal, N., & Triggs, B. (2005, 20-25 June 2005). *Histograms of oriented gradients for human detection.* Paper presented at the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA.

Duan, K.-B., Rajapakse, J. C., & Nguyen, M. N. (2007, 2007). *One-versus-one and one-versus-all multiclass SVM-RFE for gene selection in cancer classification.* Paper presented at the Evolutionary Computation,Machine Learning and Data Mining in Bioinformatics, Berlin, Heidelberg.

Fang, L., He, N., Li, S., Plaza, A. J., & Plaza, J. (2018). A new spatial–spectral feature extraction method for hyperspectral images using local covariance matrix representation. *IEEE Transactions on Geoscience and Remote Sensing, 56*(6), 3534-3546.

Felzenszwalb, P. F., Girshick, R. B., McAllester, D., & Ramanan, D. (2010). Object detection with discriminatively trained part based models. *IEEE Transactions on*

Pattern Analysis and Machine Intelligence, 32(9), 1627-1645. doi:10.1109/TPAMI.2009.167

Gavilán, M., Balcones, D., Marcos, O., Llorca, D. F., Sotelo, M. A., Parra, I., . . . Amírola, A. (2011). Adaptive road crack detection system by pavement classification. *Sensors, 11*(10), 9628-9657.

Goetz, A. F. H., Vane, G., Solomon, J. E., & Rock, B. N. (1985). Imaging spectrometry for earth remote sensing. *Science, 228*(4704), 1147. doi:10.1126/science.228.4704.1147

Graybeal, B. A., Phares, B. M., Rolander, D. D., Moore, M., & Washer, G. (2002). Visual inspection of highway bridges. *Journal of Nondestructive Evaluation, 21*(3), 67-83. doi:10.1023/A:1022508121821

Green, D. M., & Swets, J. A. (1966). *Signal detection theory and psychophysics*. New York: Wiley.

Green, R. O., Eastwood, M. L., Sarture, C. M., Chrien, T. G., Aronsson, M., Chippendale, B. J., . . . Williams, O. (1998). Imaging spectroscopy and the airborne visible/infrared imaging spectrometer (AVIRIS). *Remote Sensing of Environment, 65*(3), 227-248. doi:https://doi.org/10.1016/S0034-4257(98)00064-9

Gualtieri, J. A., & Chettri, S. (2000, 24-28 July 2000). *Support vector machines for classification of hyperspectral data.* Paper presented at the IGARSS 2000. IEEE 2000 International Geoscience and Remote Sensing Symposium. Taking the Pulse of the Planet: The Role of Remote Sensing in Managing the Environment. Proceedings (Cat. No.00CH37120).

Gualtieri, J. A., & Cromp, R. (1999). *Support vector machines for hyperspectral remote sensing classification* (Vol. 3584): SPIE(Society of Photographic Instrumentation Engineers).

Hang, R., Liu, Q., Song, H., & Sun, Y. (2015). Matrix-based discriminant subspace ensemble for hyperspectral image spatial–spectral feature fusion. *IEEE Transactions on Geoscience and Remote Sensing, 54*(2), 783-794.

Haralick, R. M., Shanmugam, K., & Dinstein, I. (1973). Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybernetics, SMC-3*(6), 610-621. doi:10.1109/TSMC.1973.4309314

Ho, H.-N., Kim, K.-D., Park, Y.-S., & Lee, J.-J. (2013). An efficient image-based damage detection for cable surface in cable-stayed bridges. *NDT & E International, 58*, 18-23. doi:https://doi.org/10.1016/j.ndteint.2013.04.006

Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (March 2013). *Applied logistic regression*: John Wiley & Sons, Inc.

Huo, G., Yang, S. X., Li, Q., & Zhou, Y. (2017). A robust and fast method for sidescan sonar image segmentation using nonlocal despeckling and active contour model. *IEEE Transactions on Cybernetics, 47*(4), 855-872. doi:10.1109/TCYB.2016.2530786

Isawa, K., Nakayama, S., Ikeda, M., Takagi, S., Tosaka, S., & Kasai, N. (2005). Robotic 3D SQUID imaging system for practical nondestructive evaluation applications. *Physica C: Superconductivity, 432*(3-4), 182-192.

Johnson, W. R., Wilson, D. W., Bearman, G. H., & Backlund, J. (2004). *An all-reflective computed tomography imaging spectrometer.* Paper presented at the Instruments, Science, and Methods for Geospace and Planetary Remote Sensing.

Kaiser, P. K., & Boynton, R. M. (1996). *Human Color Vision* (2 ed.). Washington DC: *Optical Society of America*.

Kaseko, M. S., & Ritchie, S. G. (1993). A neural network-based methodology for pavement crack detection and classification. *Transportation Research Part C: Emerging Technologies, 1*(4), 275-291.

Kim, H., Sim, S., & Cho, S. (2015). *Unmanned aerial vehicle (UAV)-powered concrete crack detection based on digital image processing*. Paper presented at the 6th International Conference on Advances in Experimental Structural Engineering 11th International Workshop on Advanced Smart Materials and Smart Structures Technology IL, USA.

Kimball, S., & Mattis, P. (2019). GIMP (Version 2.10.12): GIMP Development Team. Retrieved from https://www.gimp.org

Lattanzi, D. A., & Miller, G. (2013). *A prototype imaging and visualization system for robotic infrastructure inspection.* Paper presented at the Structures Congress 2013: Bridging Your Passion with Your Profession, Pittsburgh, Pennsylvania, United States.

Lee, J., Weger, R. C., Sengupta, S. K., & Welch, R. M. (1990). A neural network approach to cloud classification. *IEEE Transactions on Geoscience and Remote Sensing, 28*(5), 846-855. doi:10.1109/36.58972

Liu, Z., Suandi, S. A., Ohashi, T., & Ejima, T. (2002). *Tunnel crack detection and classification system based on image processing.* Paper presented at the Machine Vision Applications in Industrial Inspection X.

Loncan, L., De Almeida, L. B., Bioucas-Dias, J. M., Briottet, X., Chanussot, J., Dobigeon, N., . . . Simoes, M. (2015). Hyperspectral pansharpening: A review. *IEEE Geoscience and remote sensing magazine, 3*(3), 27-46.

Lu, G., & Fei, B. (2014). Medical hyperspectral imaging: a review. *Journal of biomedical optics, 19*(1), 010901.

Mondal, A., Kundu, S., Chandniha, S., Shukla, R., & Mishra, P. (2012). Comparison of support vector machine and maximum likelihood classification technique using satellite imagery. *International Journal of Remote Sensing and GIS, 1*, 116-123.

Morris, T. (2004). *Computer vision and image processing*: Basingstoke : Palgrave Macmillan.

Ni, F., Zhang, J., & Chen, Z. (2019). Pixel‐level crack delineation in images with convolutional feature fusion. *Structural Control and Health Monitoring, 26*(1), e2286.

Nouri, D., Lucas, Y., & Treuillet, S. (2013). *Calibration and test of a hyperspectral imaging prototype for intra-operative surgical assistance* (Vol. 8676): Progress in Biomedical Optics and Imaging - Proceedings of SPIE.

Oliveira, H., & Correia, P. L. (2009, 24-28 Aug. 2009). *Automatic road crack segmentation using entropy and image dynamic thresholding.* Paper presented at the 2009 17th European Signal Processing Conference.

Olsen, M., Chen, Z., Hutchinson, T., & Kuester, F. (2012). Optical techniques for multiscale damage assessment. *Geomatics, Natural Hazards and Risk, 4*, 1-22. doi:10.1080/19475705.2012.670668

Olsen, M. J., Barbosa, A., Burns, P., Kashani, A., Wang, H., Veletzos, M., . . . Federal Highway, A. (2016). *Assessing, coding, and marking of highway structures in*

emergency situations : volume 3 : coding and marking guidelines (Vol. 3). Washington, D.C.: Transportation Research Board.

Osowski, S., Siwek, K., & Markiewicz, T. (2004, 11-11 June 2004). *MLP and SVM networks - a comparative study.* Paper presented at the Proceedings of the 6th Nordic Signal Processing Symposium, 2004. NORSIG 2004.

Ozer, E., Feng, M. Q., & Feng, D. (2015). Citizen sensors for SHM: Towards a crowdsourcing platform. *Sensors, 15*(6), 14591-14614.

Palhang, M. b. K. A. M. E. M. (2009). Generalization performance of support vector machine and neural network in runoff modeing. *Expert System with Applications, 36*, 7624-4174. doi:10.1016/j.eswa.2008.09.053

Paola, J. D., & Schowengerdt, R. A. (1995). A detailed comparison of backpropagation neural network and maximum-likelihood classifiers for urban land use classification. *IEEE Transactions on Geoscience and Remote Sensing, 33*(4), 981-996. doi:10.1109/36.406684

Plaza, A., Benediktsson, J. A., Boardman, J. W., Brazile, J., Bruzzone, L., Camps-Valls, G., . . . Trianni, G. (2009). Recent advances in techniques for hyperspectral image processing. *Remote Sensing of Environment, 113*, S110-S122. doi:https://doi.org/10.1016/j.rse.2007.07.028

Prasanna, P., Dana, K. J., Gucunski, N., Basily, B. B., La, H. M., Lim, R. S., & Parvardeh, H. (2014). Automated crack detection on concrete bridges. *IEEE Transactions on automation science and engineering, 13*(2), 591-599.

Richards, J. A. (2013). *Remote sensing digital image analysis: An introduction* (Vol. 5;): Springer Publishing Company, Incorporated.

Richards, J. A., & Jia, X. (1999). *Remote sensing digital image analysis: An introduction*: Springer-Verlag.

Ritchie, S. G. (1987). Expert systems in pavement management. *Transportation Research Part A: General, 21*(2), 145-152. doi:https://doi.org/10.1016/0191-2607(87)90007-0

Ritchie Stephen, G. (1990). Digital imaging concepts and applications in pavement management. *Journal of Transportation Engineering, 116*(3), 287-298. doi:10.1061/(ASCE)0733-947X(1990)116:3(287)

Roh, Y., Heo, G., & Whang, S. E. (2019). A survey on data collection for machine learning: a big data-ai integration perspective. *IEEE Transactions on Knowledge and Data Engineering*.

Sakagami, T. (2015). Remote nondestructive evaluation technique using infrared thermography for fatigue cracks in steel bridges. *Fatigue & Fracture of Engineering Materials & Structures, 38*(7), 755-779. doi:10.1111/ffe.12302

Shimin Tang, & Chen, Z. (2017). *Detection of complex concrete damage – A deep learning framework and performance evaluation*. Paper presented at the International Workshop on Computing for Civil Engineering (IWCCE), Seattle, WA.

Shrivakshan, G. T., & Chandrasekar, C. (2012). A comparison of various edge detection techniques used in image processing. *International Journal of Computer Science Issues, 9*, 269-276.

Siesler, H. W., Ozaki, Y., Kawata, S., & Heise, H. M. (2008). *Near-infrared spectroscopy: principles, instruments, applications*: John Wiley & Sons.

Song, J., & Lyu, M. R. (2005). A Hough transform based line recognition method utilizing both parameter space and image space. *Pattern Recognition, 38*(4), 539-552. doi:https://doi.org/10.1016/j.patcog.2004.09.003

Stabile, T. A., Giocoli, A., Perrone, A., Palombo, A., Pascucci, S., & Pignatti, S. (2012). A new joint application of non-invasive remote sensing techniques for structural health monitoring. *Journal of Geophysics and Engineering, 9*(4), S53-S63. doi:10.1088/1742-2132/9/4/s53

Tomoyuki, Y., Shingo, N., & Shuji, H. (2008, 3-5 June 2008). *An efficient crack detection method using percolation-based image processing.* Paper presented at the 2008 3rd IEEE Conference on Industrial Electronics and Applications.

Tung, P.-C., Hwang, Y.-R., & Wu, M.-C. (2002). The development of a mobile manipulator imaging system for bridge crack inspection. *Automation in construction, 11*(6), 717-729.

Vaghefi, K., Ahlborn Theresa, M., Harris Devin, K., & Brooks Colin, N. (2015). Combined imaging technologies for concrete bridge deck condition assessment. *Journal of Performance of Constructed Facilities, 29*(4), 04014102. doi:10.1061/(ASCE)CF.1943-5509.0000465

Valença, J., Dias-da-Costa, D., Gonçalves, L., Júlio, E., & Araújo, H. (2014). Automatic concrete health monitoring: assessment and monitoring of concrete surfaces. *Structure and Infrastructure Engineering, 10*(12), 1547-1554. doi:10.1080/15732479.2013.835326

Vapnik, V. N. (1998). *Statistical learning theory*. *Wiley-Interscience*.

Waldbjørn, J., Høgh, J., Wittrup-Schmidt, J., Nielsen, M. W., Branner, K., Stang, H., & Berggreen, C. (2014). Strain and displacement controls by fibre bragg grating and digital image correlation. *Strain, 50*(3), 262-273. doi:10.1111/str.12089

Yeum, C. M., & Dyke, S. J. (2015). Vision-based automated crack detection for bridge inspection. *Computer-Aided Civil and Infrastructure Engineering, 30*(10), 759-770. doi:10.1111/mice.12141

Zakeri, H., Nejad, F. M., & Fahimifar, A. (2017). Image based techniques for crack detection, classification and quantification in asphalt pavement: A review. *Archives of Computational Methods in Engineering, 24*(4), 935-977.

Zhang, C., & Elaksher, A. (2012). An unmanned aerial vehicle‐based imaging system for 3D measurement of unpaved road surface distresses 1. *Computer‐Aided Civil and Infrastructure Engineering, 27*(2), 118-129.

Zhang, J., & Modestino, J. W. (1989). *A markov random field model-based approach to image interpretation* (Vol. 1199): SPIE.

Zhang, L., Yang, F., Zhang, Y. D., & Zhu, Y. J. (2016). *Road crack detection using deep convolutional neural network.* Paper presented at the 2016 IEEE international conference on image processing (ICIP).

Zhang, Q., Tian, Y., Yang, Y., & Pan, C. (2014). Automatic spatial–spectral feature selection for hyperspectral image via discriminative sparse multimodal learning. *IEEE Transactions on Geoscience and Remote Sensing, 53*(1), 261-279.

VITA

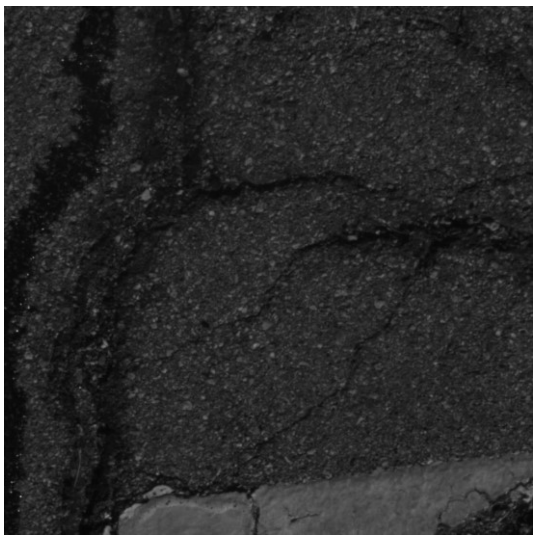
Sameer Aryal was born in Chitwan, Nepal on April 13th, 1992. He joined the Himalaya College of Engineering to pursue his Civil Engineering Studies. He was awarded a Bachelor of Engineering degree from Tribhuwan University in April 2016.

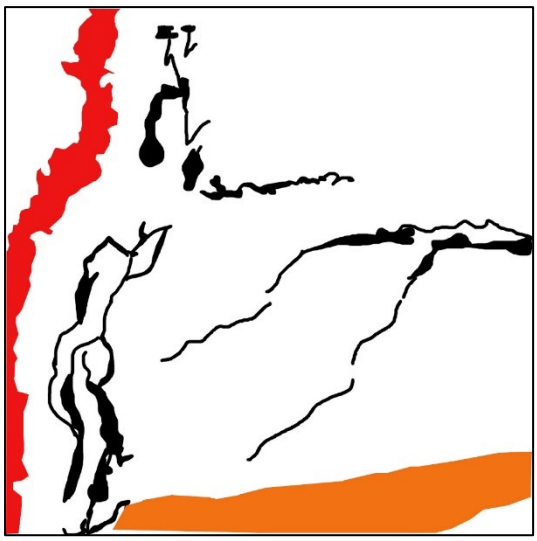
Sameer joined the University of Missouri – Kansas City (UMKC) in the spring term of 2018 to pursue his Master's degree in Civil Engineering. He started working as a Graduate Research Assistant in the Civil and Mechanical Engineering Department under the supervision of Dr. ZhiQiang Chen. He pursued his research in the field of structural health assessment using regular RGB images and Hyperspectral images.

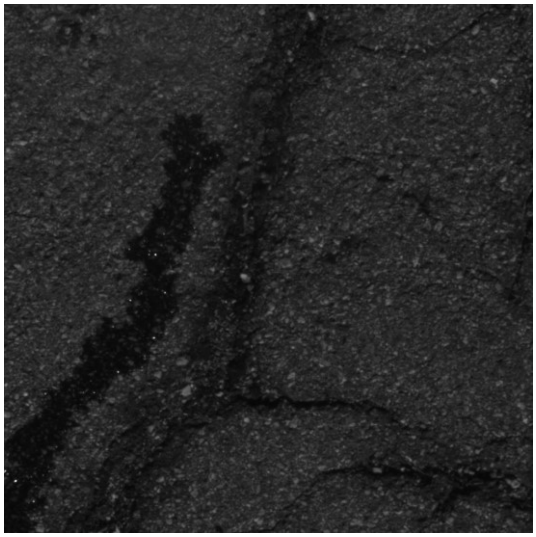
Appendix-1

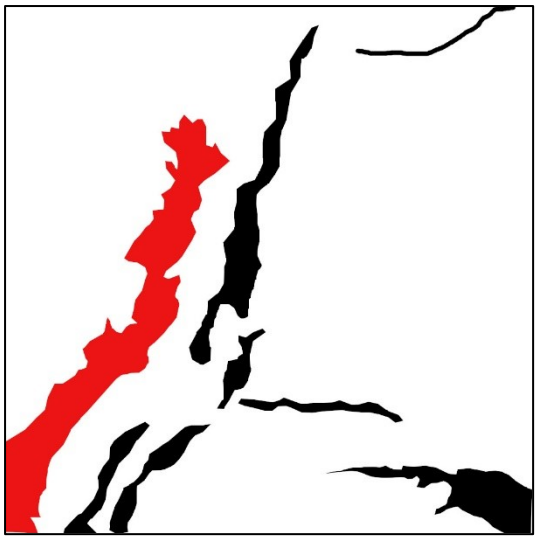
Here within appendix-1, all the images that were used in this research along with their respective mask have been presented. In the first section, 16 images on different artifacts on asphalts surface are presented followed by 34 images of the concrete surface with different artifacts on it.

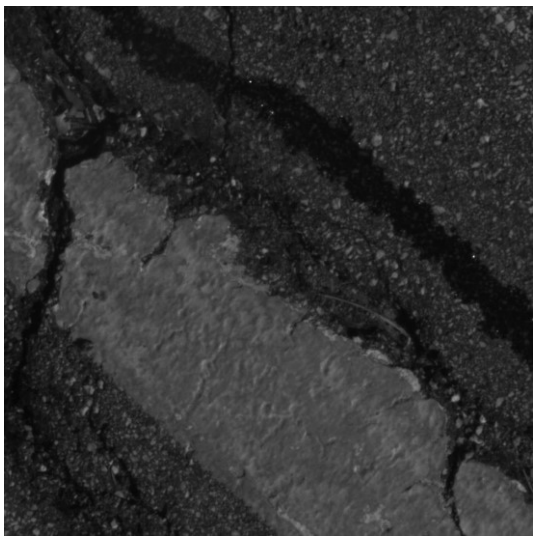
Asphalt Surface

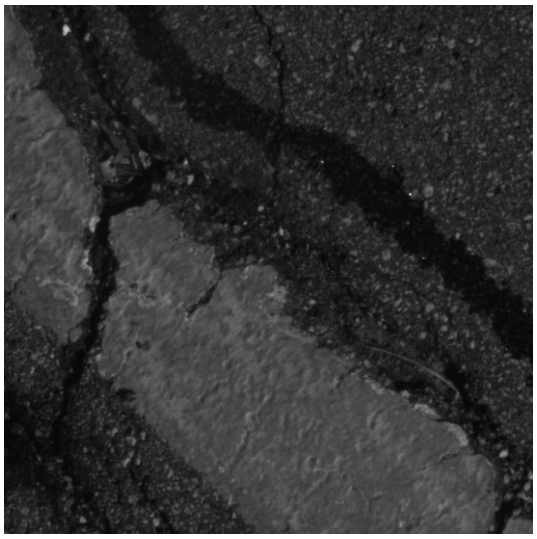


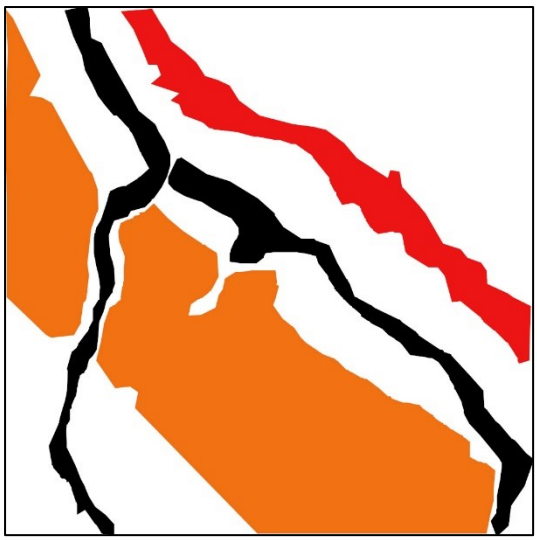


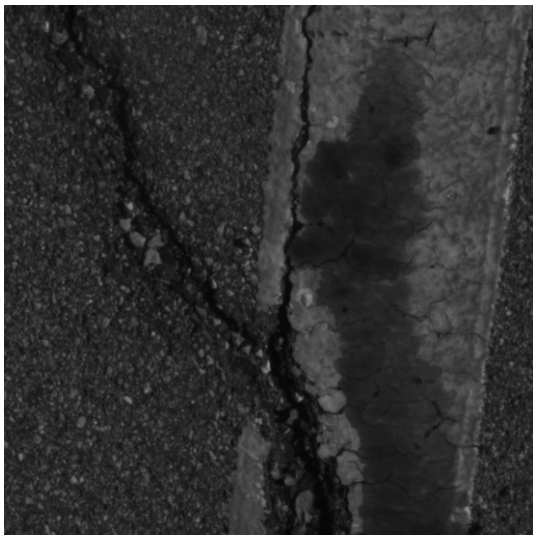


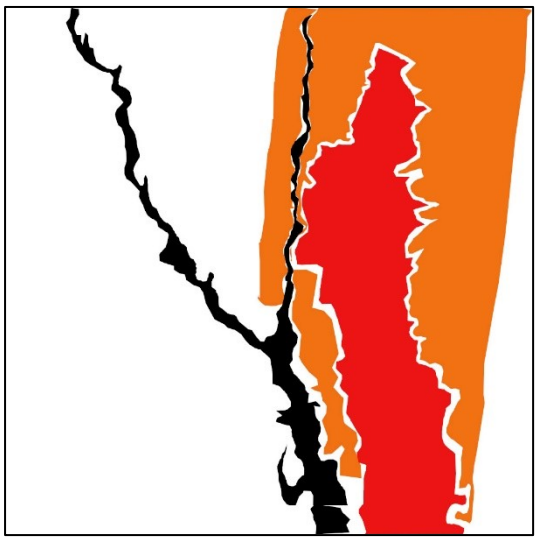


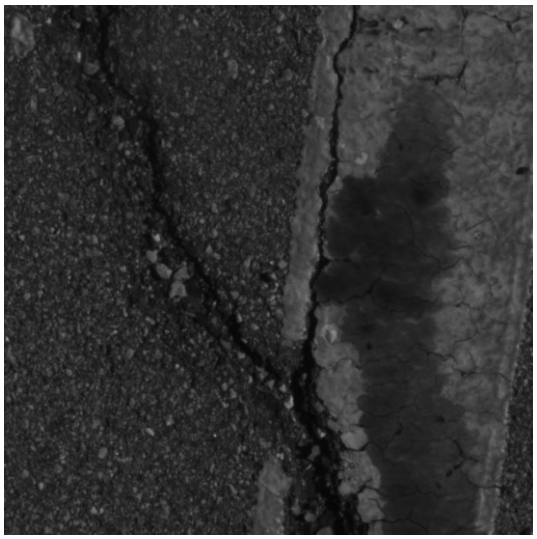


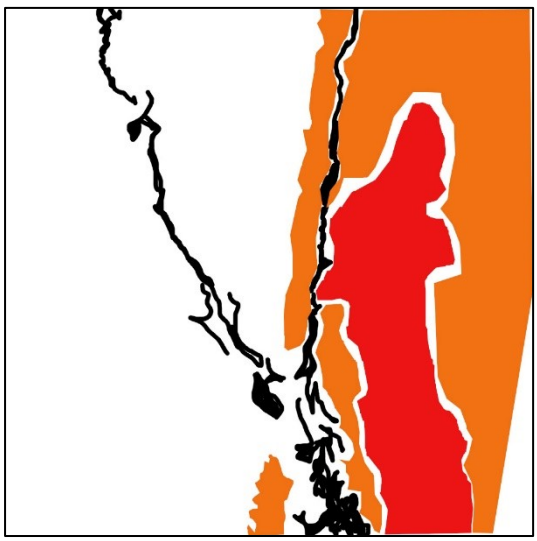


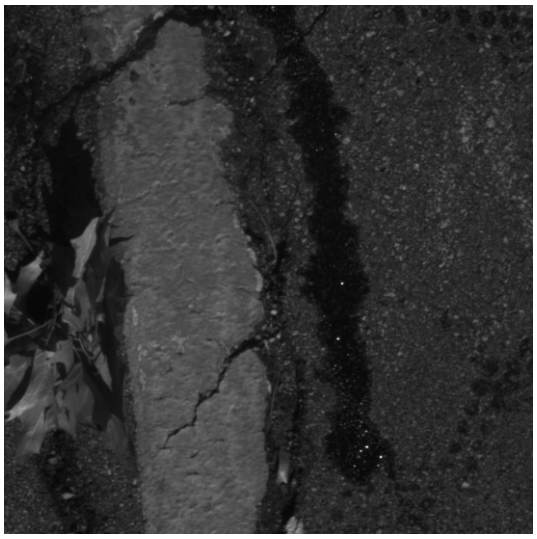


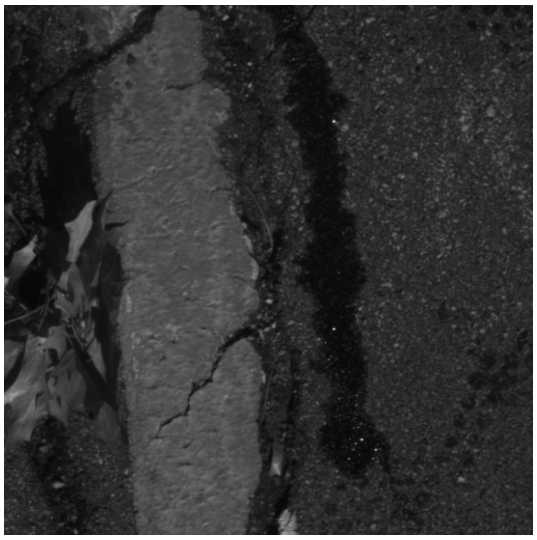


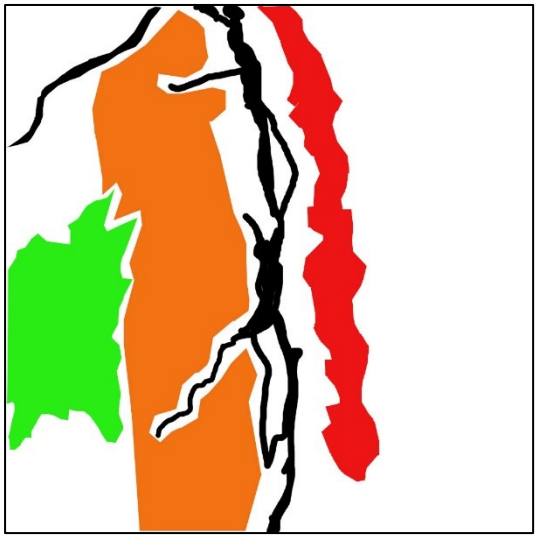


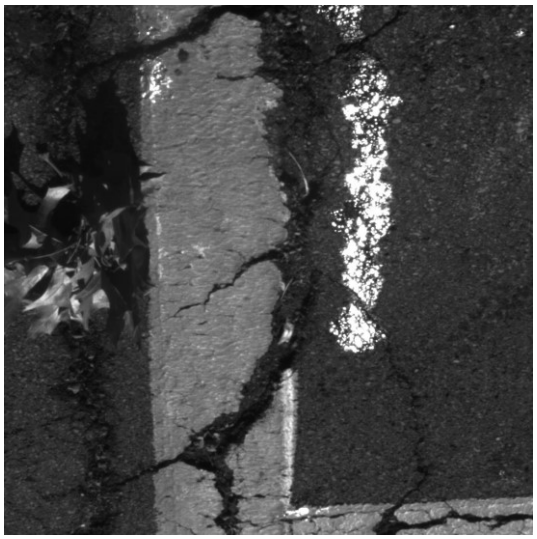


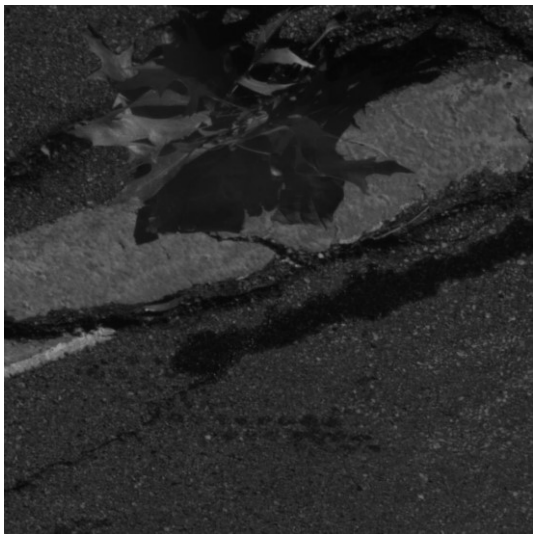


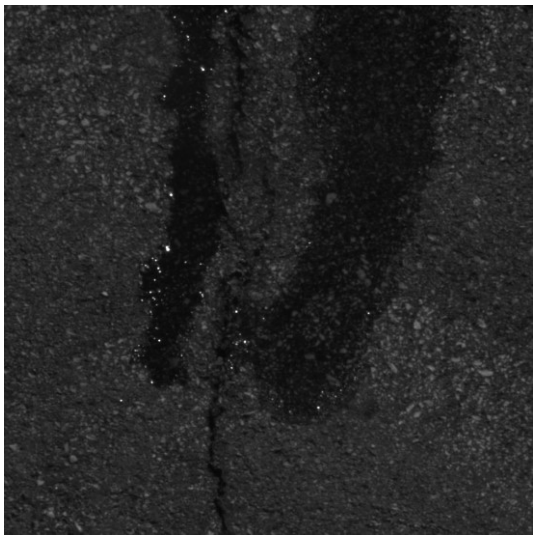


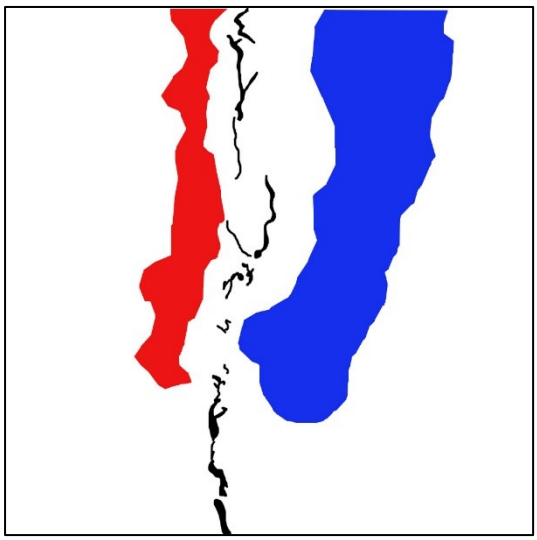


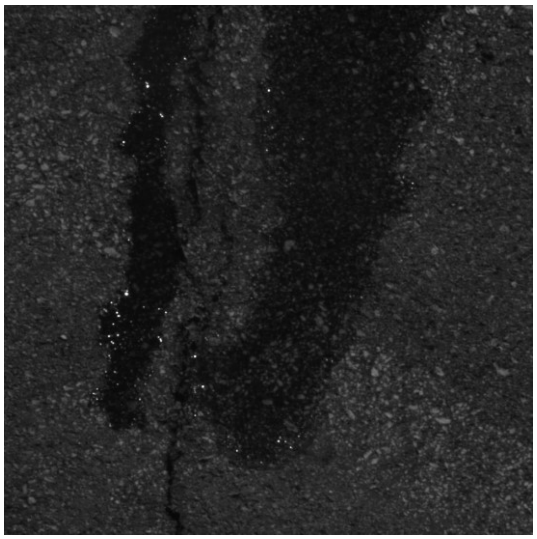


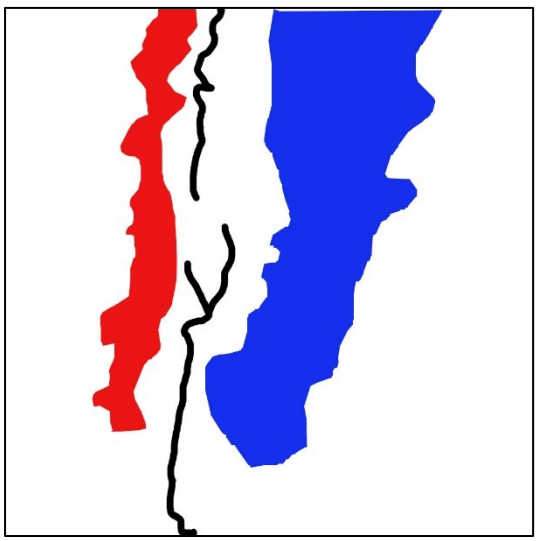


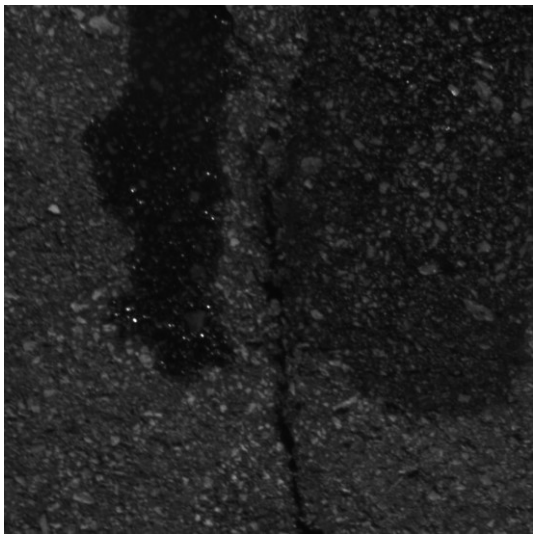


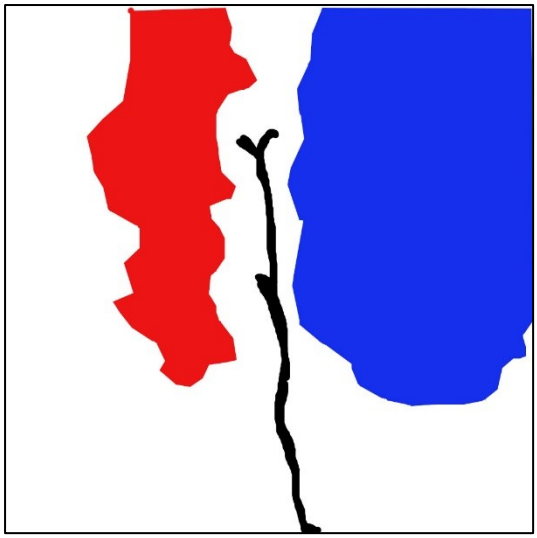


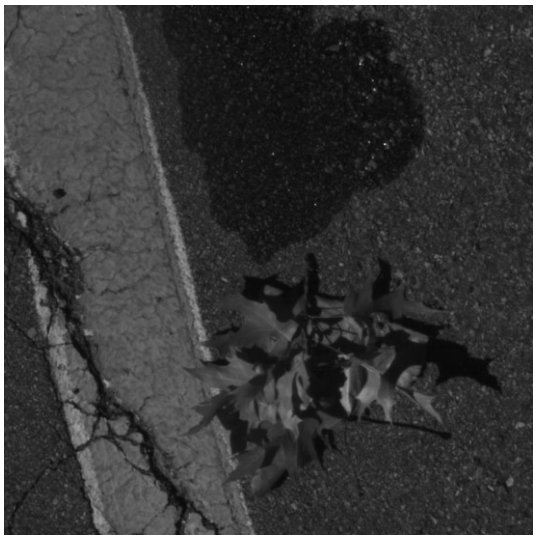




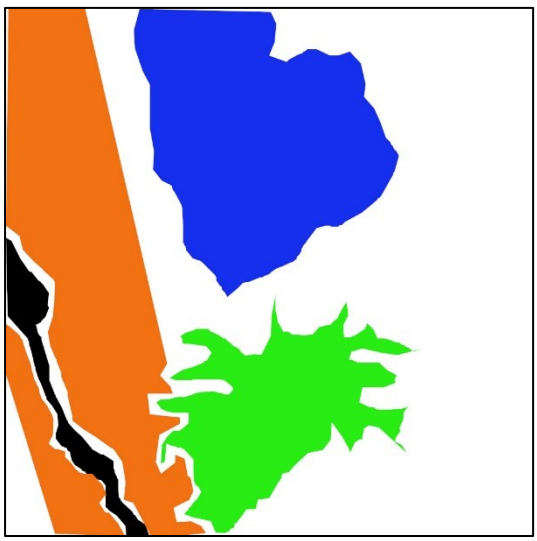


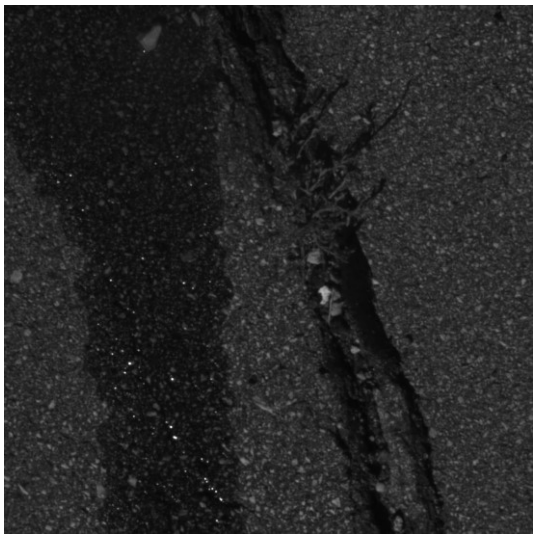


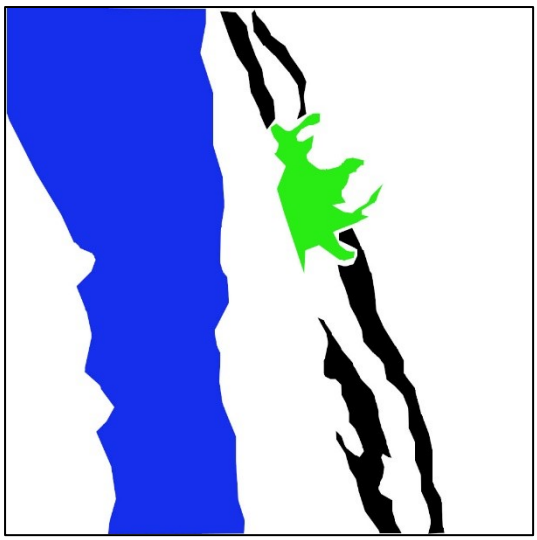




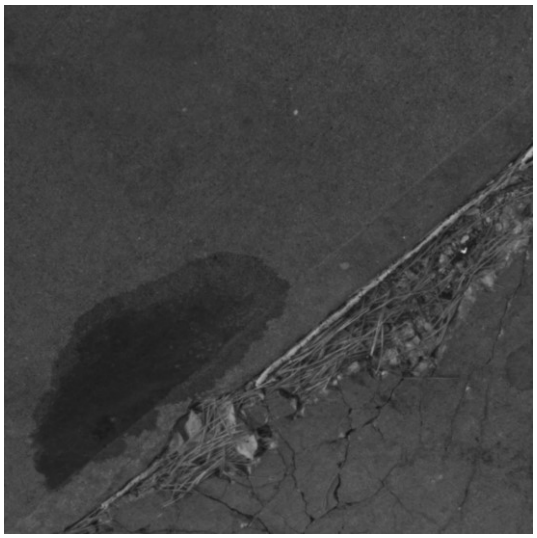
f

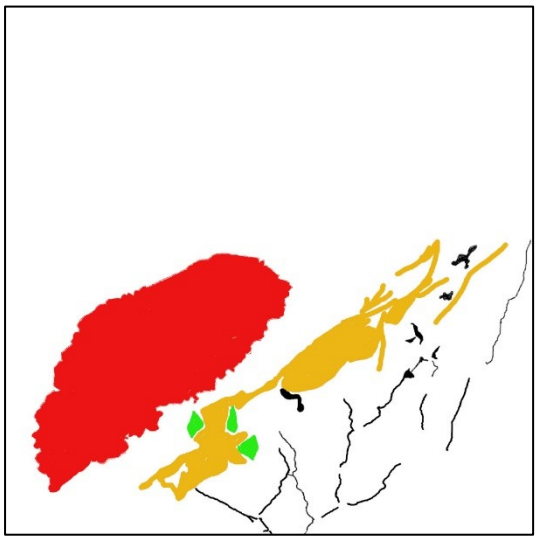


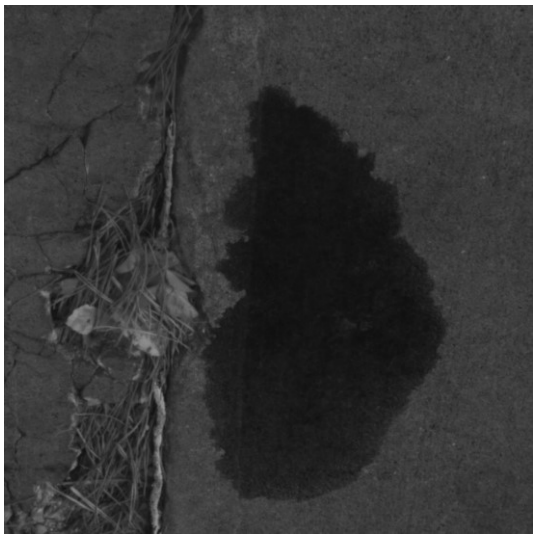


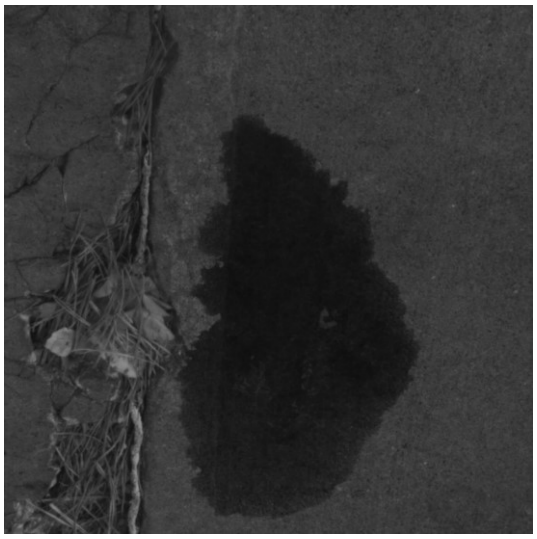


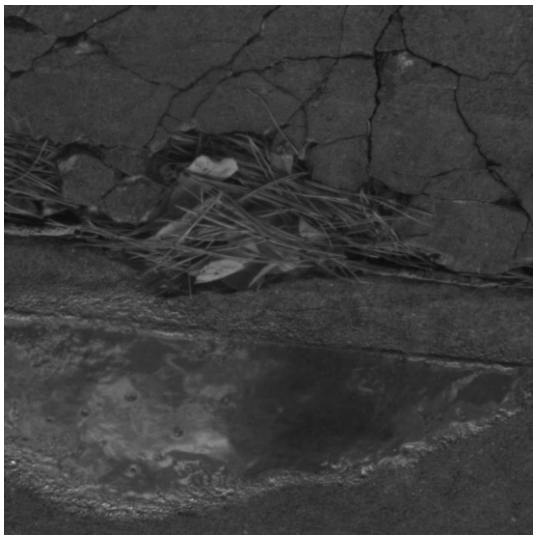
Concrete surface

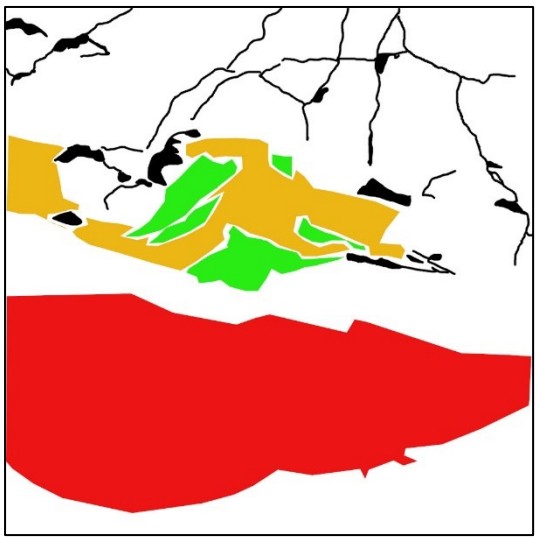


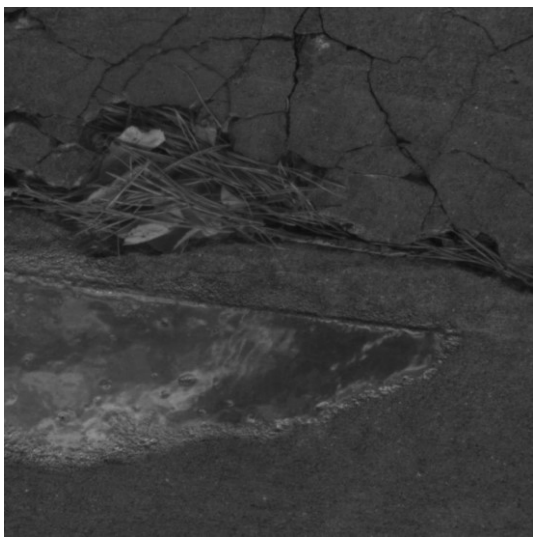


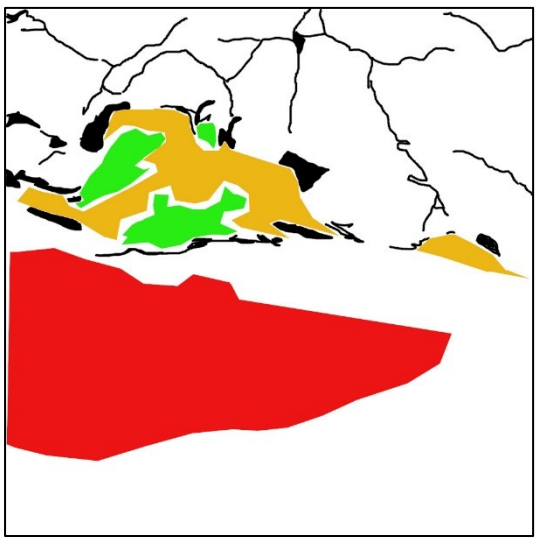


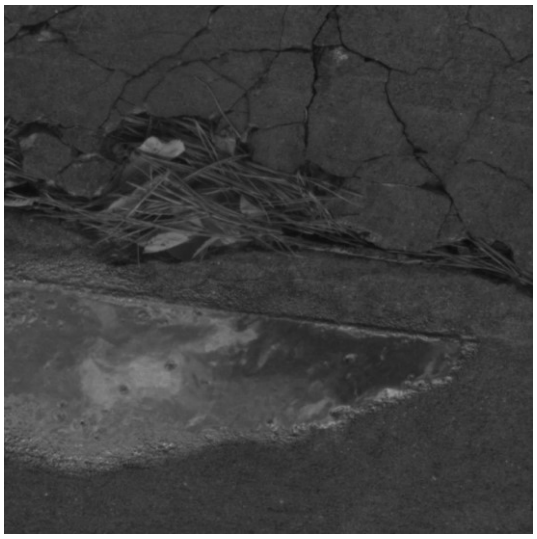


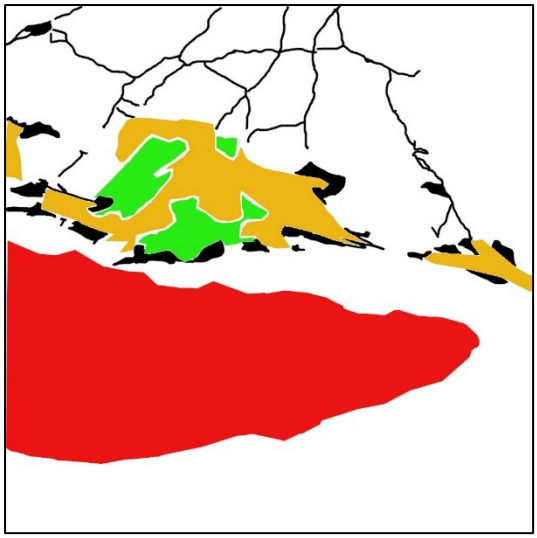


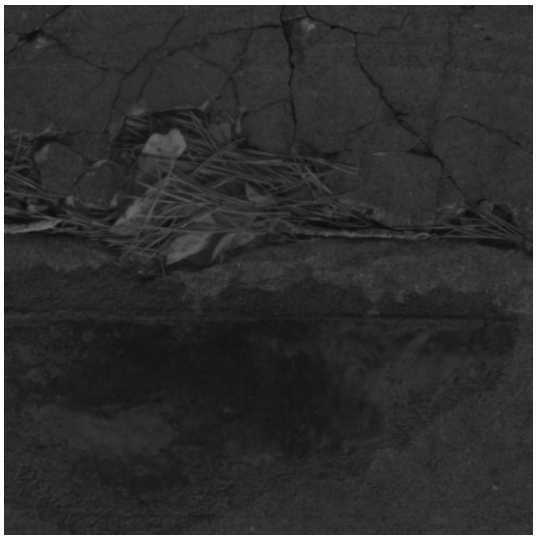


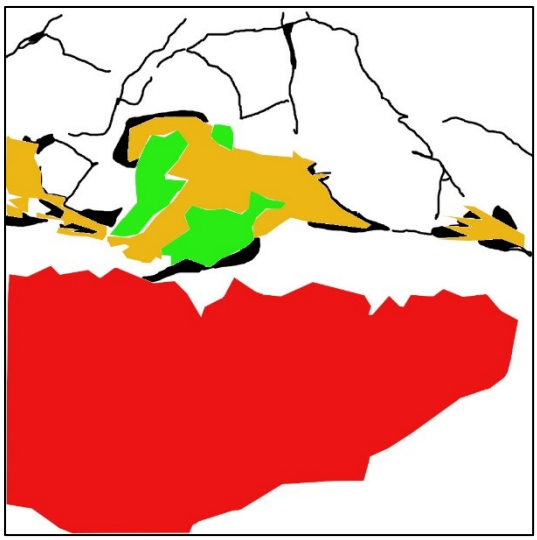


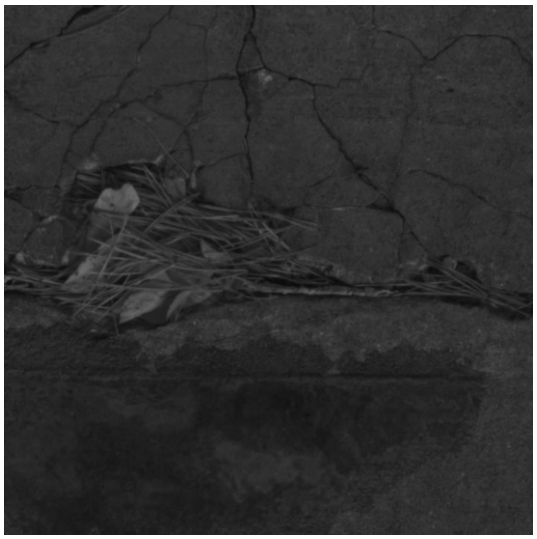


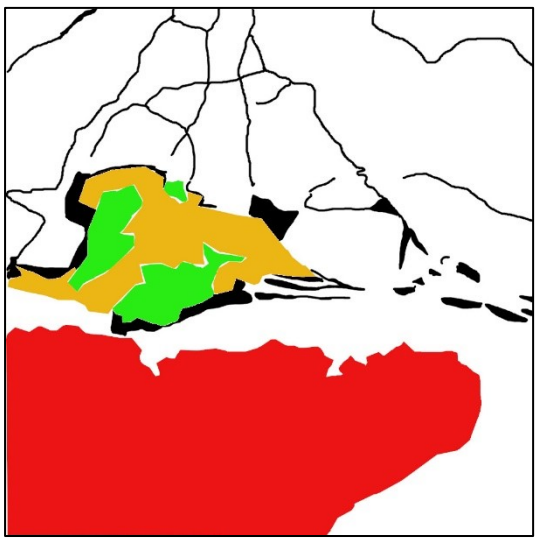


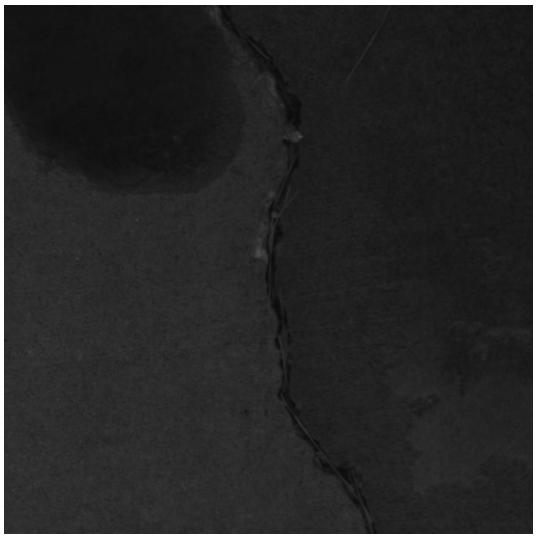
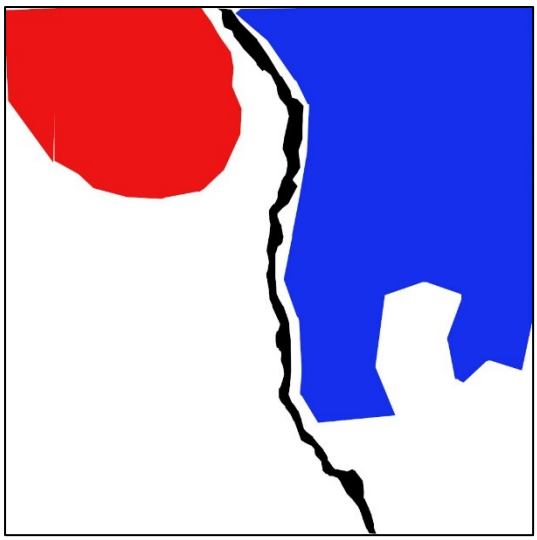
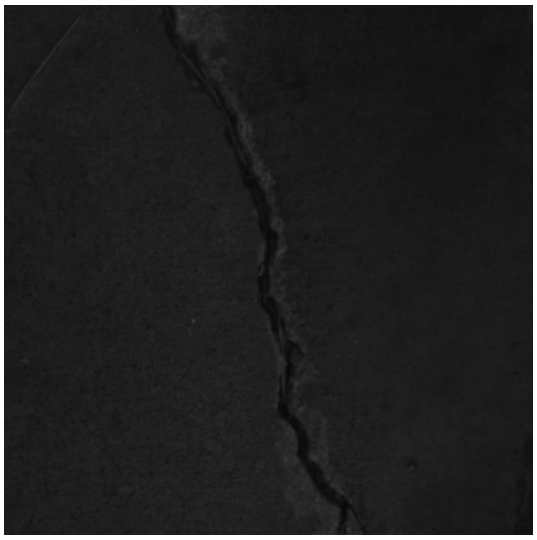
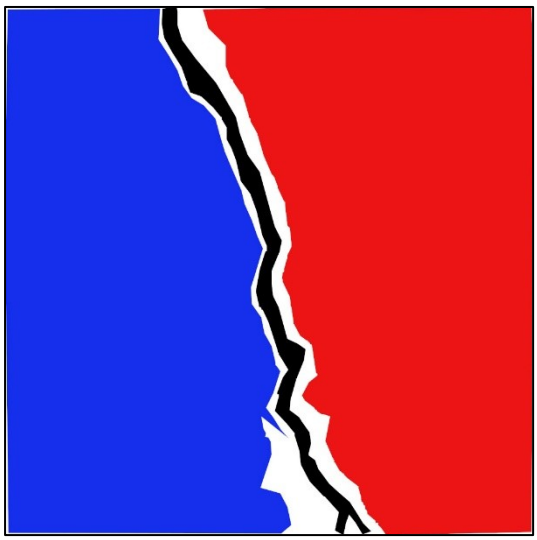
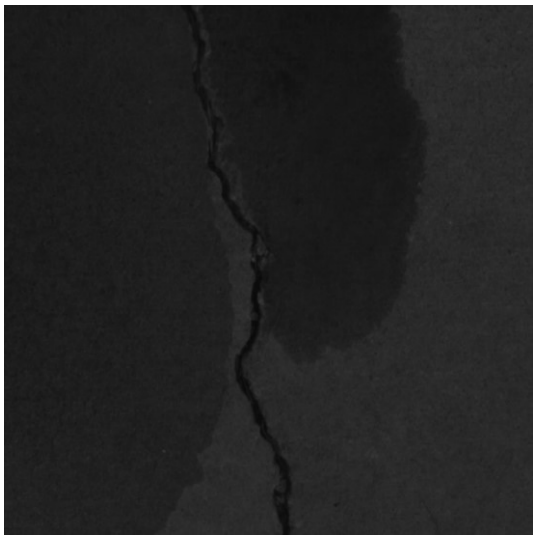
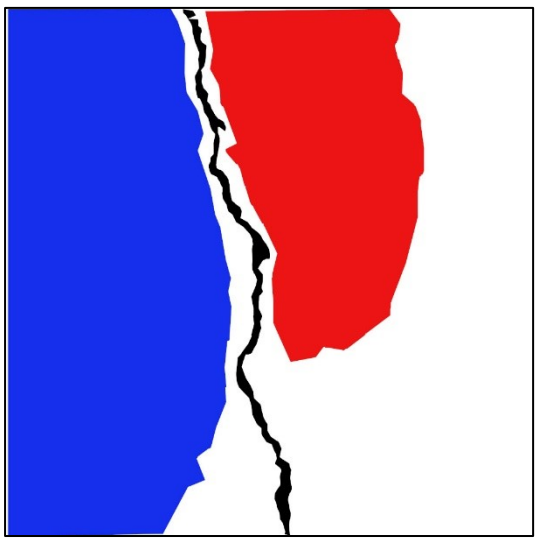


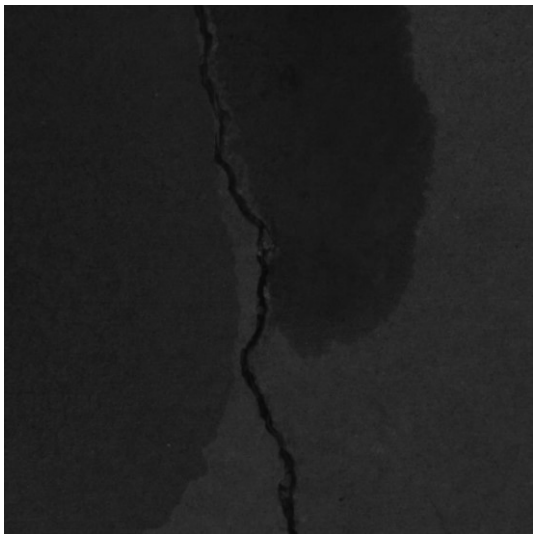


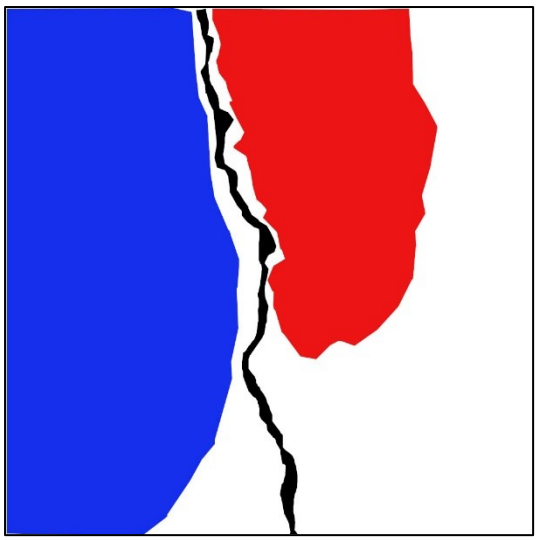


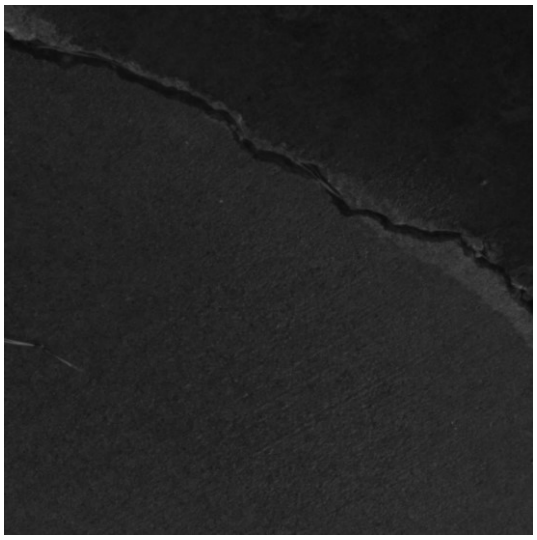


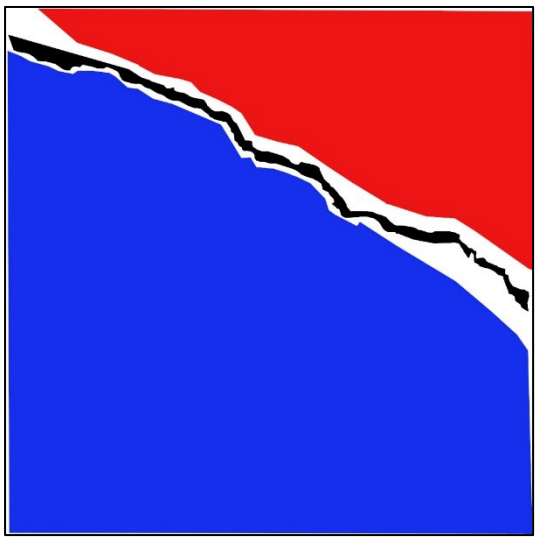


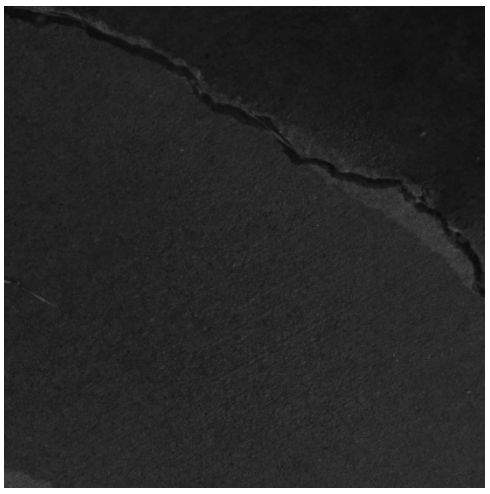







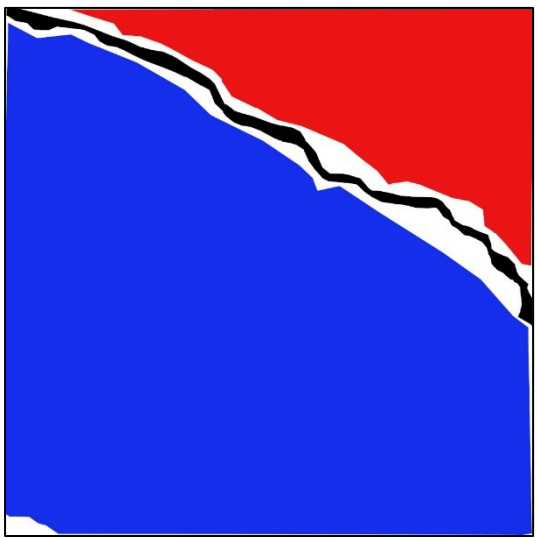


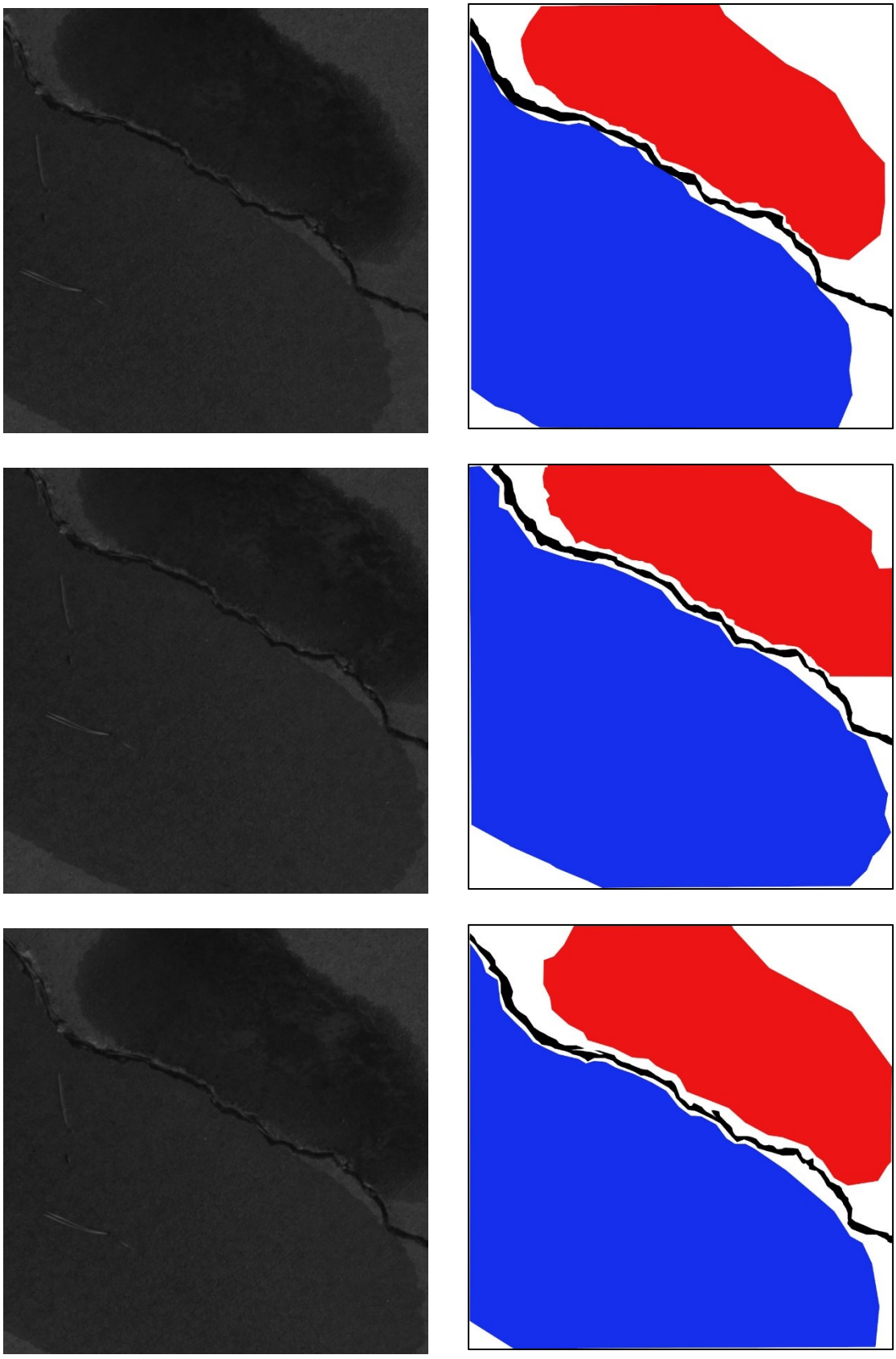


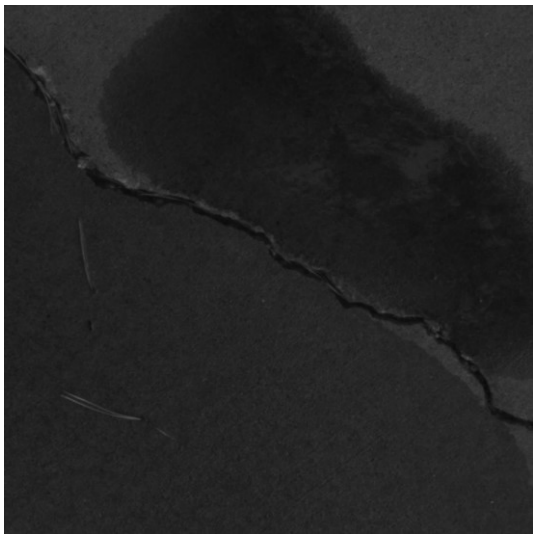


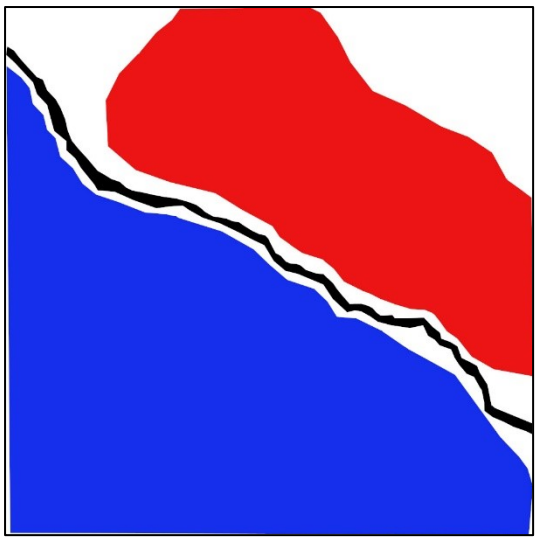


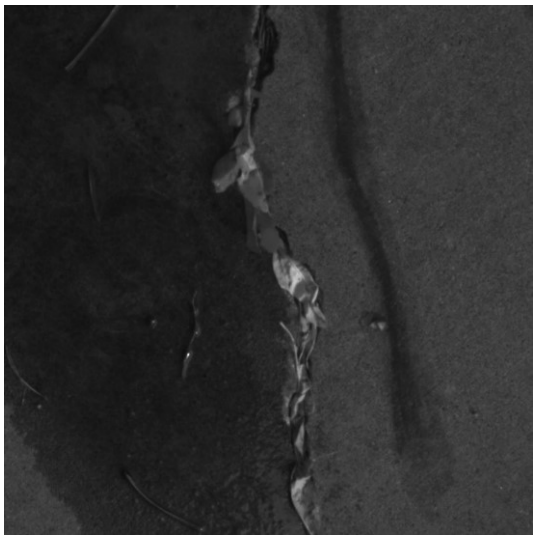


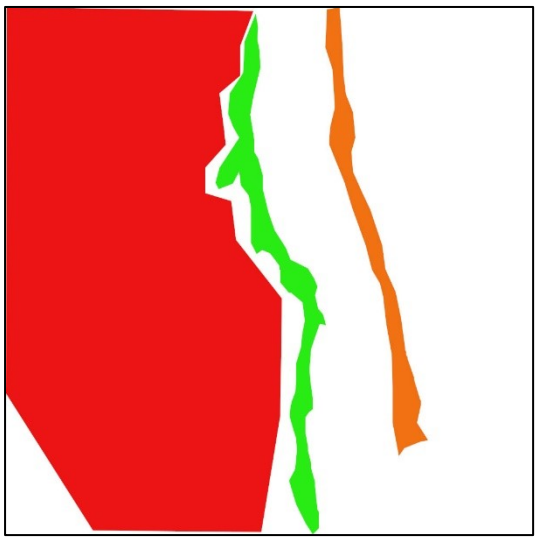


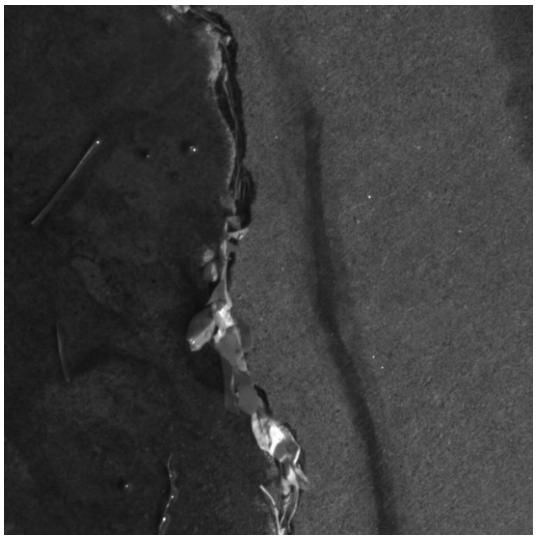


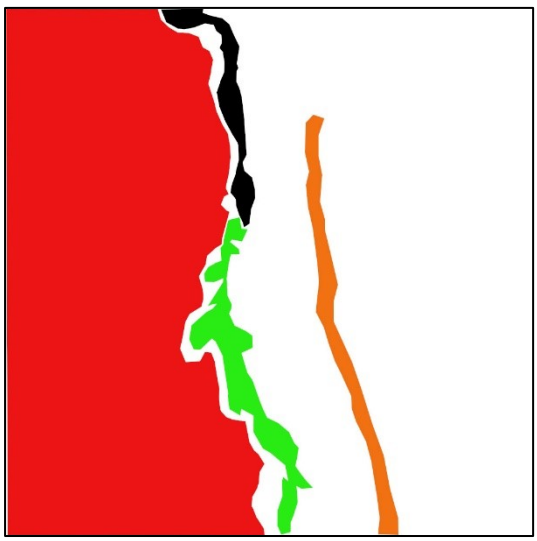


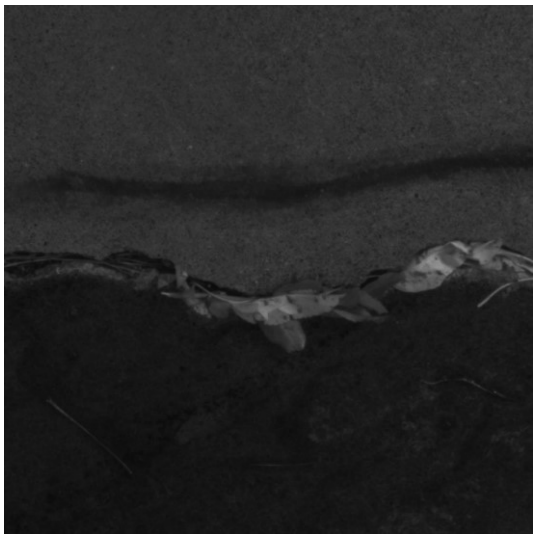


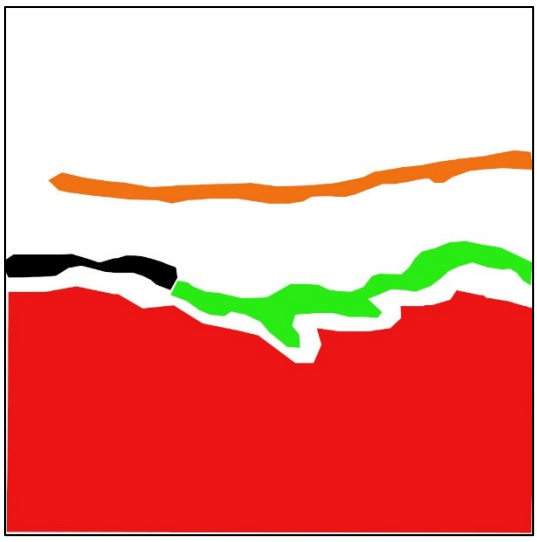


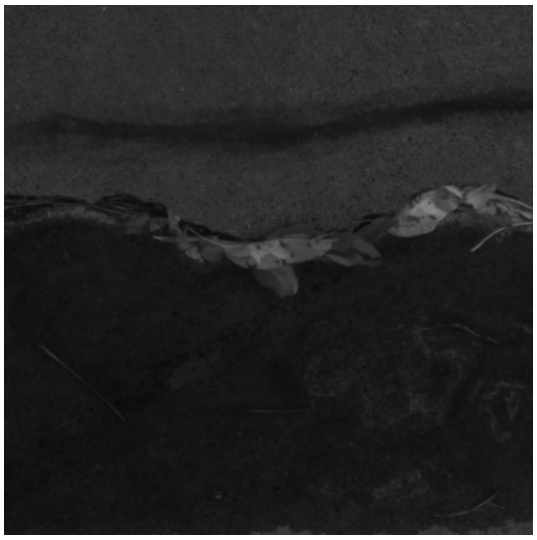


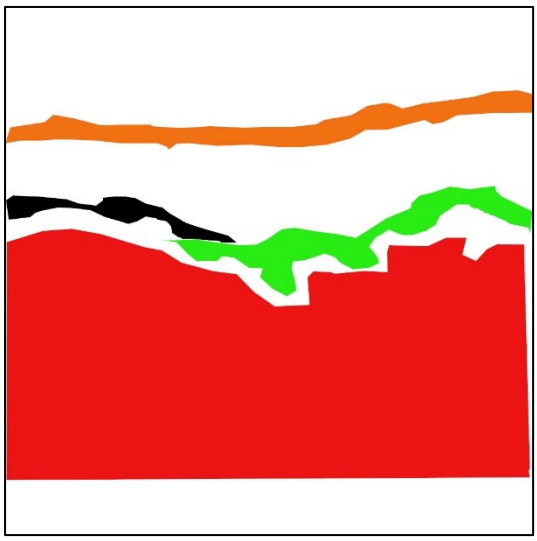


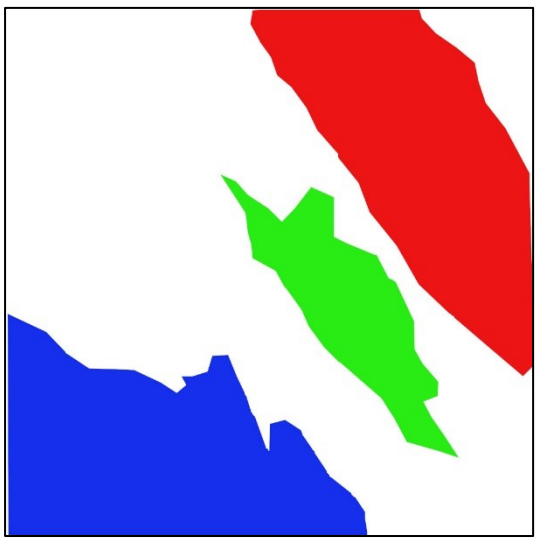


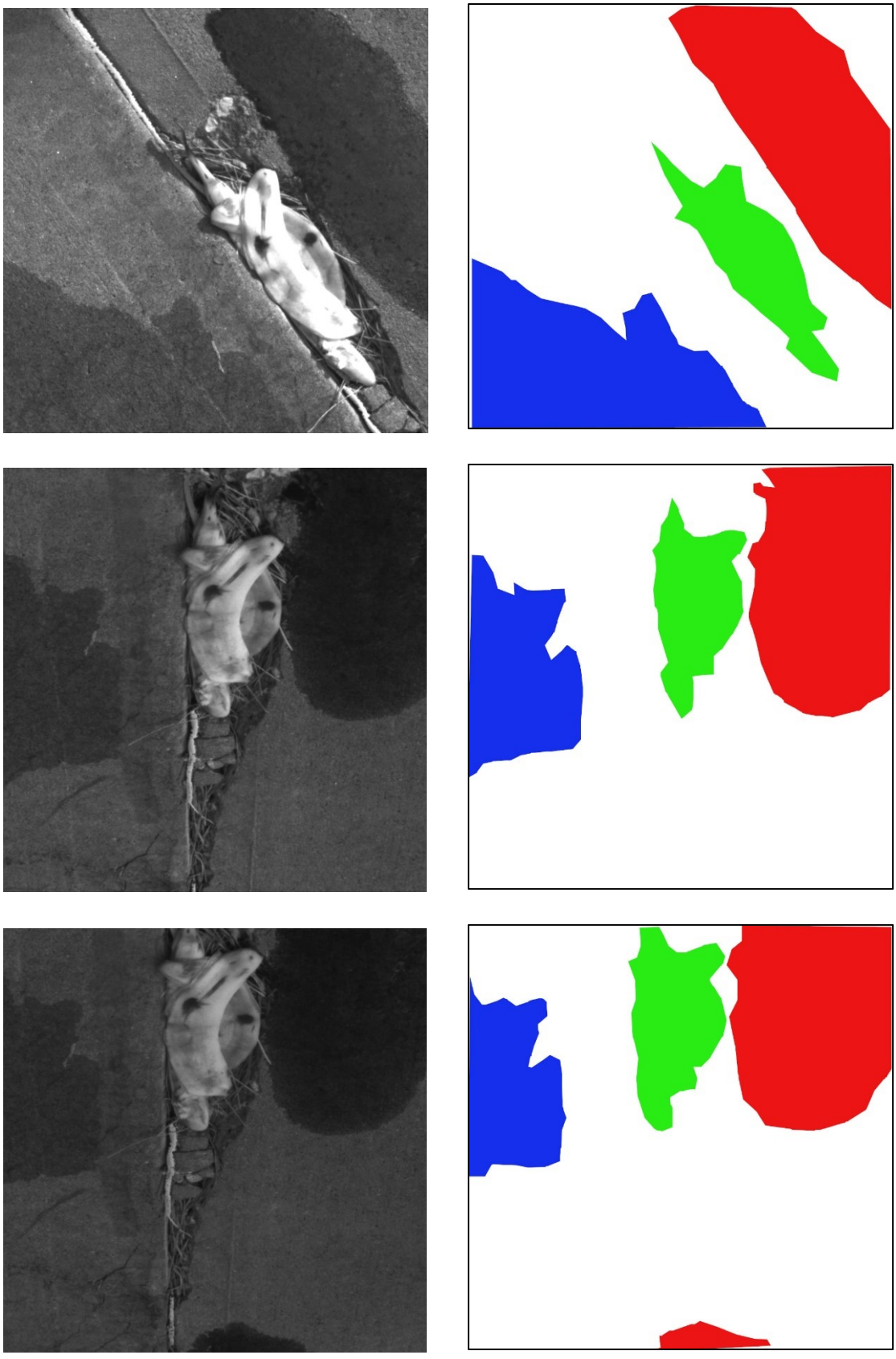


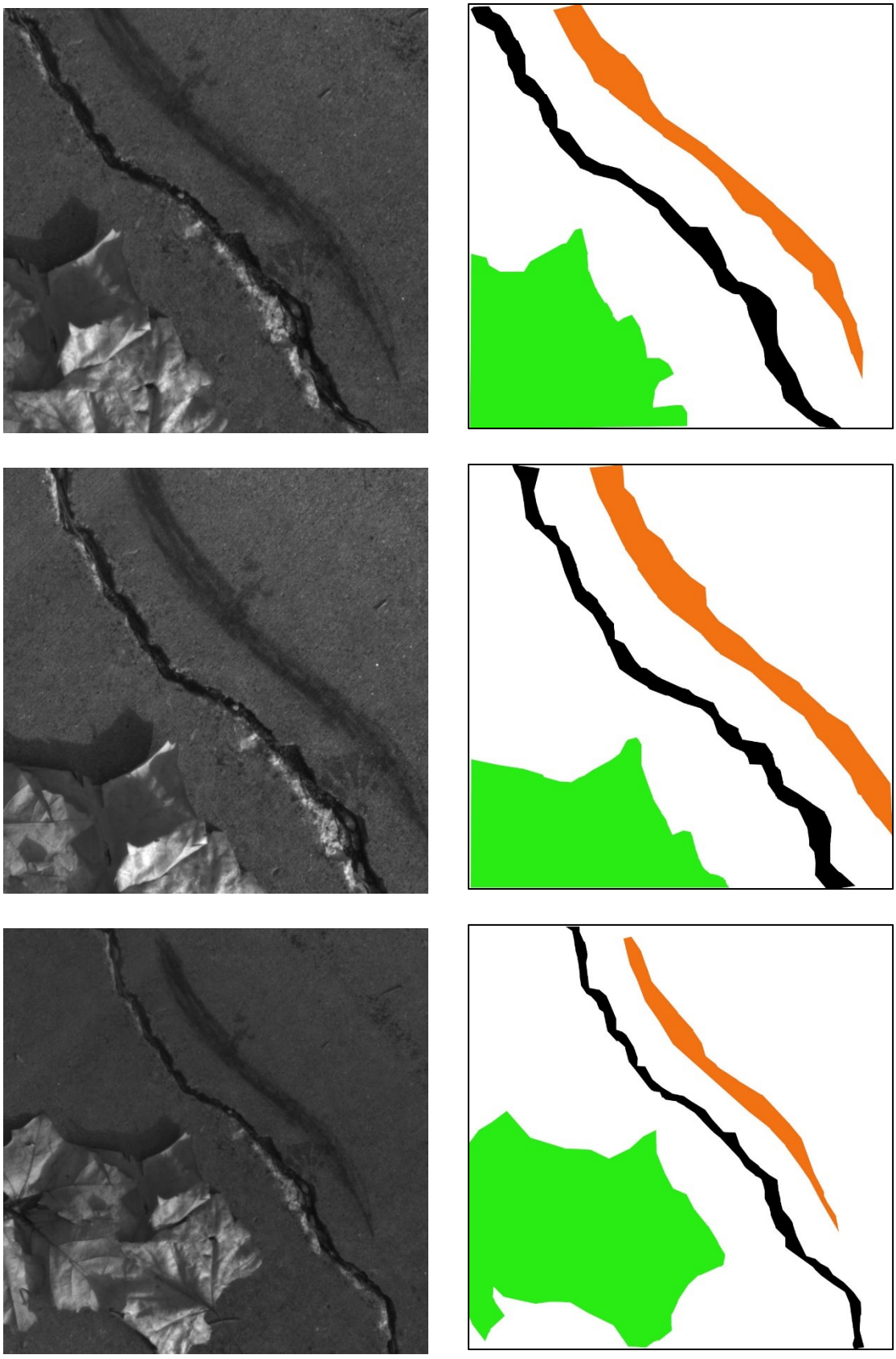


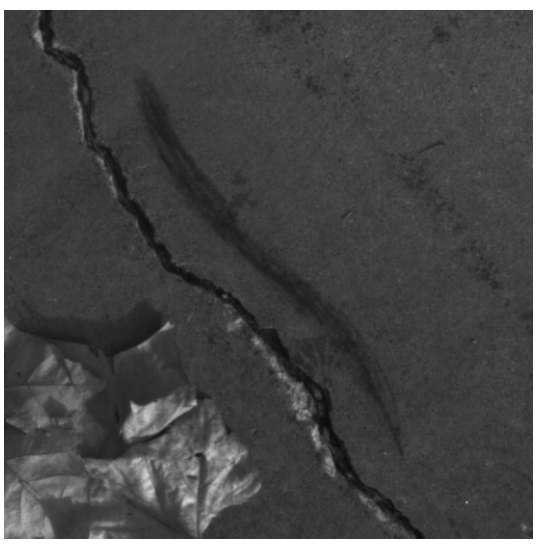


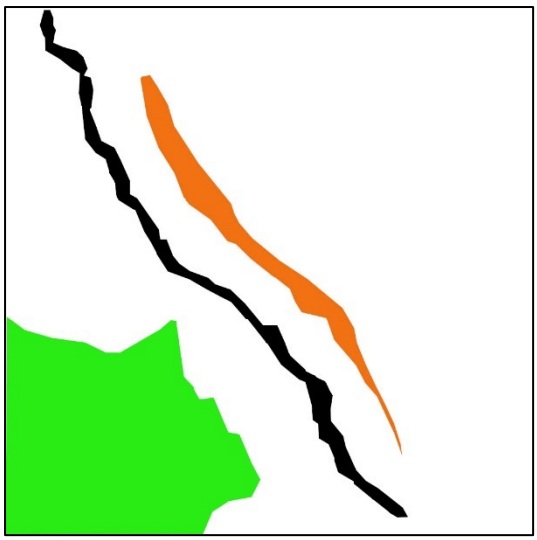


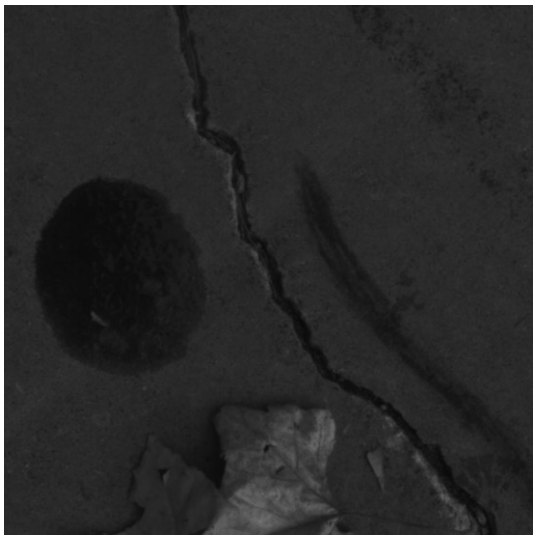


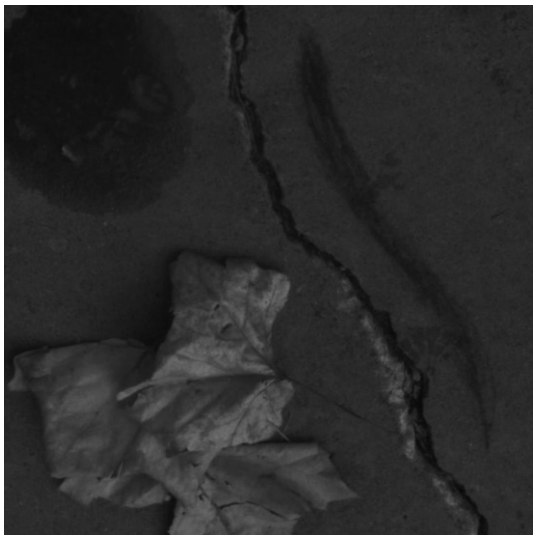


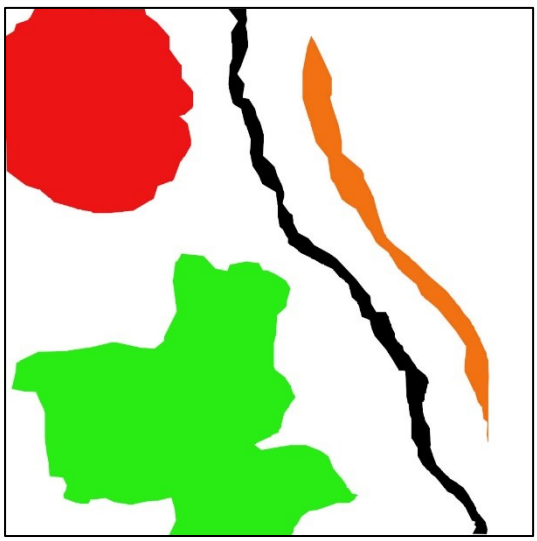


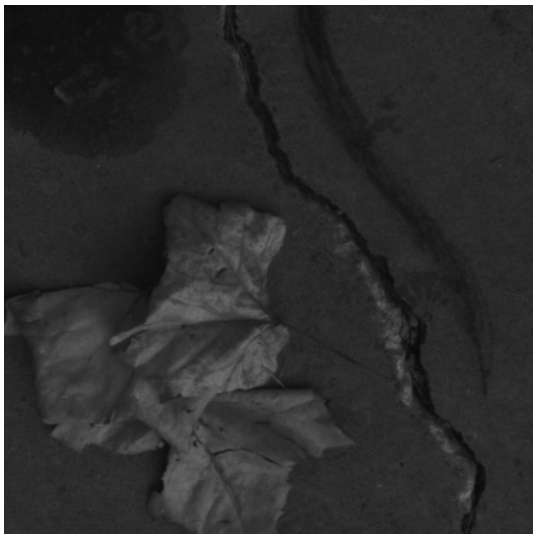


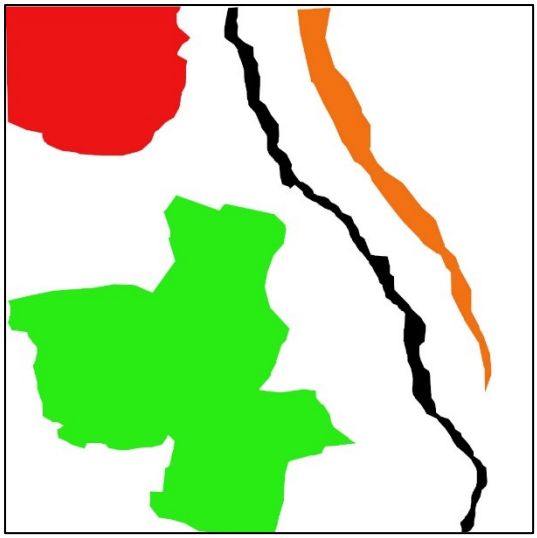












Appendix 2

In appendix 2, the MatLab codes written for the different sections of the total workflow of this research has been presented. To begin with, two functions for reading the hypercube and high-resolution gray image of the respective hyperspectral images is presented. Spectral feature extraction of the hyperspectral image with the assistance of the mask image, dimensionality reduction using Principal Component Analysis (PCA), grayscale feature extraction using Histogram of Oriented Gradient (HOG) and dataset preparation for training and testing the SVM classifer respectively has been presented next. Finally, this section is concluded with detailed code for extraction of results for classification like the confusion matrix, ROC and PR- curves and summary statistics of the classifier.

Function to read gray scale image

```
function [ img] = ReadGray(count, Dir_Mask,
FullDir_Mask)
        I =
fullfile(Dir_Mask,FullDir_Mask(count).name);
        img = imread(I);
       img = img(:,:,2);
end
```

Function to read hyperspectral image

```
function [ imgHyper] = ReadHyper(count,
Dir_Hyperspectral, FullDir_Hyperspectral)

        IH =
fullfile(Dir_Hyperspectral,FullDir_Hyperspectral(count)
.name);
        imgHyper = double(imread(IH));
end
```

Extracting the spectral profile and HOG features of different classes on concrete and asphalt surface.

```
% Defining directory with hyperspectral cube
Dir_Hyperspectral =
'C:\Users\sayd8\Documents\DataSet_Complex\Concrete\50_5
0';
FullDir_Hyperspectral =
dir(fullfile(Dir_Hyperspectral,'Auto*.tiff'));
% Defining directory with respective mask
Dir_Mask          =
'C:\Users\sayd8\Documents\DataSet_Complex\Concrete\Mask
';
FullDir_Mask = dir(fullfile(Dir_Mask,'Auto*.jpg'));
% Defining directory for respective gray scale image
Dir_Gray =
'C:\Users\sayd8\Documents\DataSet_Complex\Concrete\Gray
_Scale';
FullDir_Gray =   dir(fullfile(Dir_Gray,'Auto*.jpg'));
% Assign counter values
R = 1; C = 1;
```

```
counter_row = 1; counter_col = 1;
cellSize = 20;
Cz = 0; Wz = 0; Oz = 0; Dz = 0;Gz = 0;Az = 0;
Cx = 1; Wx = 1; Ox = 1; Dx = 1;Gx = 1;Ax = 1;


for count = 1:size(FullDir_Mask, 1)
        % Reading hyperspectral cube
        hyperspectral_image = ReadHyper(count,
Dir_Hyperspectral, FullDir_Hyperspectral);
        % Reading respective mask
        Mask = ReadGray(count, Dir_Mask, FullDir_Mask);
        % Reading grayscale image
        GrayImage = ReadGray(count, Dir_Gray,
FullDir_Gray);
        im = im2single(GrayImage); % Converting gray
image to single format
        % Extracting HOG feature from gray image
        hog = vl_hog(im, cellSize, 'Verbose');

     for row_block = 1 : 20 : 1000  %Working on 20*20
block of mask image
       for col_block = 1 : 20 : 1000

              for ct_row = R : R+19
                 for ct_column = C : C+19;
     % Count number of considered pixel of respective
class in each block.
                     if Mask(ct_row, ct_column)  == 0 ;
                         Cz = Cz + 1; %Crack pixel
counter
                 else if Mask(ct_row, ct_column) == 47;
                         Wz = Wz + 1; % Water pixel
counter
                 else if Mask(ct_row, ct_column) == 22;
                         Oz = Oz + 1; % Oil pixel
counter
                 else if Mask(ct_row, ct_column) == 182;
                         Dz = Dz + 1; % Dry vegetation
pixel counter
                 else if Mask(ct_row, ct_column)  ==
113;
                         Az = Az + 1; % Artificial color
pixel counter
```

```
                else if  Mask(ct_row, ct_column) ==
236;
                        Gz = Gz + 1; % Green vegetation
pixel counter
                    end
                    end
                    end
                    end
                    end
                    end
                end
            end
% Extracting spectrum profile from the hyperspectral
cube and HOG features
% From grayscale image for the class with more than 200
pixels in each block
% If the number of cracked pixels is greater than 2000
% Spectral profile for class crack
      if Cz > 200 ;          Crack_Spectrum1{Cx ,1} =
hyperspectral_image(counter_row,counter_col,:);
                              % HOG feature for class
crack
                             Crack_Hog1{Cx,1} =
hog(counter_row,counter_col,:);
                              Cx = Cx + 1;
                             % Spectral profile for
class water
      else if Wz > 200 ;     Water_Spectrum1{Wx,1} =
hyperspectral_image(counter_row,counter_col,:);
                             % HOG feature for class
water
                             Water_Hog1{Wx,1} =
hog(counter_row,counter_col,:);
                             Wx = Wx + 1;
                             % Spectral profile for
class Oil
      else if Oz > 200 ;     Oil_Spectrum1{Ox,1} =
hyperspectral_image(counter_row,counter_col,:);
                             % HOG feature for class Oil
                             Oil_Hog1{Ox,1} =
hog(counter_row,counter_col,:);
                             Ox = Ox + 1;
                             % Spectral profile for
class dry vegetation
```

```
      else if Dz > 200 ;    Dry_Spectrum1{Dx,1} =
hyperspectral_image(counter_row,counter_col,:);
                            % HOG feature for class dry
vegetation
                            Dry_Hog1{Dx,1} =
hog(counter_row,counter_col,:);
                            Dx = Dx + 1;
                            % Spectral profile for
class artificial color
      else if Az > 200 ;    Color_Spectrum1{Ax,1} =
hyperspectral_image(counter_row,counter_col,:);
                            % HOG feature for class
artificial color
                            Color_Hog1{Ax,1} =
hog(counter_row,counter_col,:);
                            Ax = Ax + 1;
                            %   Spectral profile for
class green vegetation
      else if Gz > 200 ;    Green_Spectrum1{Gx,1} =
hyperspectral_image(counter_row,counter_col,:);
                            %   HOG features for class
green vegetation
                            Green_Hog1{Gx,1} =
hog(counter_row,counter_col,:);
                            Gx = Gx + 1;
          end
          end
          end
          end
          end
      end
       Cz= 0; Wz=0; Oz=0; Dz=0; Gz=0; Az=0;% resetting
the pixel counter
             C = C +20; % Moving the block along the
row to nect set of 20 pixels
             counter_col = counter_col+1;
       end
       R = R + 20; % Moving the block to next row of 20
pixels.
       C = 1;
       counter_col = 1;
     end
     R = 1;
     C = 1;
end
```

```
% Apppending spectral and HOG features obtained from
concrete and asphalt surface
%Class Artificial Color
Color_Spectrum = [Color_Spectrum1 ; Color_Spectrum2];%
Spectral Feature
Color_Hog = [Color_Hog1; Color_Hog2];% HOG Feature
%Class Crack
Crack_Spectrum = [Crack_Spectrum1;
Crack_Spectrum2];%Spectral Features
Crack_Hog = [Crack_Hog1; Crack_Hog2];% HOG Features
%Class Dry Vegetation
Dry_Spectrum = [Dry_Spectrum1]; % Spectral Feature
Dry_Hog = [Dry_Hog1];% HOG Feature
% Class Green vegetation
Green_Spectrum = [Green_Spectrum1;
Green_Spectrum2]; %Spectral Feature
Green_Hog = [Green_Hog1 ; Green_Hog2];% HOG Feature
%Class Water
Water_Spectrum = [Water_Spectrum1(1:5000);
Water_Spectrum2];% Spectral Feature
Water_Hog = [Water_Hog1(1:5000) ; Water_Hog2];% HOG
Feature
% Class Oil
Oil_Spectrum = [Oil_Spectrum1(1:5000);
Oil_Spectrum2];%SpectralFeature
Oil_Hog = [Oil_Hog1(1:5000) ; Oil_Hog2];%HOG Feature


%Re-shuffling the obtained feature vector for each
class
Color_Train = cell2mat(Color_Spectrum);
Color_Train =
reshape(Color_Train,size(Color_Train,1)*1,139);
Color_HoGTrain = cell2mat(Color_Hog);
Color_HoGTrain =
reshape(Color_HoGTrain,size(Color_HoGTrain,1)*1,31);

Crack_train = cell2mat(Crack_Spectrum);
Crack_train =
reshape(Crack_train,size(Crack_train,1)*1,139);
Crack_HoGTrain = cell2mat(Crack_Hog);
Crack_HoGTrain =
reshape(Crack_HoGTrain,size(Crack_HoGTrain,1)*1,31);
```

```
Dry_Train = cell2mat(Dry_Spectrum);
Dry_Train = reshape(Dry_Train,size(Dry_Train,1)*1,139);
Dry_HoGTrain = cell2mat(Dry_Hog);
Dry_HoGTrain =
reshape(Dry_HoGTrain,size(Dry_HoGTrain,1)*1,31);

Green_Train = cell2mat(Green_Spectrum);
Green_Train =
reshape(Green_Train,size(Green_Train,1)*1,139);
Green_HoGTrain = cell2mat(Green_Hog);
Green_HoGTrain =
reshape(Green_HoGTrain,size(Green_HoGTrain,1)*1,31);

Water_Train = cell2mat(Water_Spectrum);
Water_Train =
reshape(Water_Train,size(Water_Train,1)*1,139);
Water_HoGTrain = cell2mat(Water_Hog);
Water_HoGTrain =
reshape(Water_HoGTrain,size(Water_HoGTrain,1)*1,31);

Oil_Train = cell2mat(Oil_Spectrum);
Oil_Train = reshape(Oil_Train,size(Oil_Train,1)*1,139);
Oil_HoGTrain = cell2mat(Oil_Hog);
Oil_HoGTrain =
reshape(Oil_HoGTrain,size(Oil_HoGTrain,1)*1,31);


% Re-shuffling the obtained spectral feature for
concrete and asphalt surfaces respectively.
Concrete_Train = cell2mat(Concrete_Spectrum);
Concrete_Train =
reshape(Concrete_Train,size(Concrete_Train,1)*1,139);
Asphalt_Train = cell2mat(Asphalt_Spectrum);
Asphalt_Train =
reshape(Asphalt_Train,size(Asphalt_Train,1)*1,139);


% HOG feature extraction for Concrete and asphalt
surface
Dir_Concrete =
'C:\Users\sayd8\Documents\DataSet_Complex\Plain';
FullDir_Concrete =
dir(fullfile(Dir_Concrete,'Auto*.jpg'));
```

```
Dir_Asphalt =
'C:\Users\sayd8\Documents\DataSet_Complex\Plain\Asphalt
';
FullDir_Asphalt =
dir(fullfile(Dir_Asphalt,'Auto*.jpg'));
Cx = 1;
for count = 1 : 9
    Asphalt_img = ReadGray(count, Dir_Asphalt,
FullDir_Asphalt);
    Concrete_img = ReadGray(count, Dir_Concrete,
FullDir_Concrete);
    cellSize = 20; %HOG feature are extracted in 20*20
block of 1000*1000 gray image
         im1 = im2single(Concrete_img);
         im2 = im2single(Asphalt_img);
        hog1 = vl_hog(im1, cellSize, 'Verbose');
        hog2 = vl_hog(im2, cellSize, 'Verbose');
    % Extracting the HOG feature of every third pixel
in 50*50 dimensional space.
    for row = 1 : 3 : 50
       for col = 1 : 3 : 50
           Concrete_Hog{Cx ,1} = hog1(row,col,:);
           Asphalt_Hog{Cx ,1} = hog2(row,col,:);
    Cx = Cx + 1;
       end
     end

end

% Re-shuffling the extracted HOG feature for concrete
and asphalt surface respectively.
Concrete_HoGTrain = cell2mat(Concrete_Hog);
Concrete_HoGTrain =
reshape(Concrete_HoGTrain,size(Concrete_HoGTrain,1)*1,3
1);
Asphalt_HoGTrain = cell2mat(Asphalt_Hog);
Asphalt_HoGTrain =
reshape(Asphalt_HoGTrain,size(Asphalt_HoGTrain,1)*1,31)
;
```

Principal Component Analysis

```
% Principal Component Analysis for Spectral Dataset
% Division of respective dataset into training and
testing

%PCA for Class artificial Color
[x_row x_col] = size(Color_Train);
m = mean(Color_Train');
d = Color_Train - repmat(m',1,139);
X = PCA(d);
Color_Train =
Color_Train(randperm(size(Color_Train,1)),:); % PC data
reshuffling
Color_TrainingSet = Color_Train(1:1568,:);   %Training
set for PC of Class Artificial Color
Color_Test = Color_Train(1569:size(Color_Train),:); %
Testing set for PC of Class Artificial Color

%PCA for Class Crack
[x_row x_col] = size(Crack_train);
m = mean(Crack_train');
d = Crack_train - repmat(m',1,139);
X = PCA(d);
Crack_train =
Crack_train(randperm(size(Crack_train,1)),:);% PC data
re-shuffling
Crack_TrainingSet = Crack_train(1:595,:); % Training
set for PC of Class Crack
Crack_Test = Crack_train(596:size(Crack_train),:); %
Testing set for PC of class Crack

% PCA for Class Green Vegetation
[x_row x_col] = size(Green_Train);
m = mean(Green_Train');
d = Green_Train - repmat(m',1,139);
X = PCA(d);
Green_Train =
Green_Train(randperm(size(Green_Train,1)),:); %PC data
re-shuffling
Green_TrainingSet = Green_Train(1:1328,:); % Training
set for PC of Class Green Vegetation
Green_Test = Green_Train(1329:size(Green_Train),:);%
Testing set for PC of Green Vegetation
```

```
% PCA of Class Water
[x_row x_col] = size(Water_Train);
m = mean(Water_Train');
d = Water_Train - repmat(m',1,139);
X = PCA(d);
Water_Train =
Water_Train(randperm(size(Water_Train,1)),:);
 % PC data re shuffling
Water_TrainingSet = Water_Train(1:2033,:);
 % Training set for PC of Class Water
Water_Test = Water_Train(2034:size(Water_Train),:);
% Testing set for PC of class Water

%PCA of Class Concrete
[x_row x_col] = size(Concrete_Train);
m = mean(Concrete_Train');
d = Concrete_Train - repmat(m',1,139);
X = PCA(d);
Concrete_Train =
Concrete_Train(randperm(size(Concrete_Train,1)),:);
% PC data reshuffling
Concrete_TrainingSet = Concrete_Train(1:651,:);
 % Training set for PC of class concrete
Concrete_Test =
Concrete_Train(652:size(Concrete_Train),:);
 % Testing set for PC of class Concrete

%PCA of Class Asphalt
[x_row x_col] = size(Asphalt_Train);
m = mean(Asphalt_Train');
d = Asphalt_Train - repmat(m',1,139);
X = PCA(d);
Asphalt_Train =
Asphalt_Train(randperm(size(Asphalt_Train,1)),:); % PC
data reshuffling
Asphalt_TrainingSet = Asphalt_Train(1:651,:);  %
Trainig set for PC of class Asphalt
Asphalt_Test =
Asphalt_Train(652:size(Asphalt_Train),:);% Testing set
for PC of class Asphalt

%PCA of Class Dry Vegetation
[x_row x_col] = size(Dry_Train);
m = mean(Dry_Train');
d = Dry_Train - repmat(m',1,139);
```

```
X = PCA(d);
Dry_Train = Dry_Train(randperm(size(Dry_Train,1)),:);%
Re-shuffling of data
Dry_TrainingSet = Dry_Train(1:239,:);  % Training set
for PC of class dry Vegetation
Dry_Test = Dry_Train(240:size(Dry_Train),:); % Testing
set for PC of class dry vegetation


% Training and testing set formation for Hog Data Set
% Asphalt HoG dataset
Asphalt_Train =
Asphalt_HoGTrain(randperm(size(Asphalt_HoGTrain,1)),:);
Asphalt_TrainingHoG = Asphalt_Train(1:651,:);
% Training Set
Asphalt_TestingHog = Asphalt_Train(652:2601,:);
% Testing Set

%Color HoG Dataset
Color_Train =
Color_HoGTrain(randperm(size(Color_HoGTrain,1)),:);
Color_TrainingHoG = Color_Train(1:1568,:); % Training
Set
Color_TestingHoG = Color_Train(1569:6273,:); % Testing
Set

%Concrete HoG DataSet
Concrete_Train =
Concrete_HoGTrain(randperm(size(Concrete_HoGTrain,1)),:
);
Concrete_TrainingHoG = Concrete_Train(1:651,:); %
Training Set
Concrete_TestingHoG = Concrete_Train(652:2601,:); %
Testting Set

%Crack HoG DataSet
Crack_Train =
Crack_HoGTrain(randperm(size(Crack_HoGTrain,1)),:);
Crack_TrainingHoG = Crack_Train(1:595,:); %Training Set
Crack_TestingHoG = Crack_Train(596:2377,:); % Testing
Set

%Dry HoG DataSet
DRY_Train =
Dry_HoGTrain(randperm(size(Dry_HoGTrain,1)),:);
```

```
Dry_TrainingHoG = DRY_Train(1:240,:); % Training Set
Dry_TestingHoG = DRY_Train(241:957,:);% Testing Set

%Oil HoG DataSet
Oil_Train =
Oil_HoGTrain(randperm(size(Oil_HoGTrain,1)),:);
Oil_TrainingHoG = Oil_Train(1:1804,:); % Training Set
Oil_TestingHoG = Oil_Train(1805:7214,:);% Testing Set

%Water HoG DataSet
Water_Train =
Water_HoGTrain(randperm(size(Water_HoGTrain,1)),:);
Water_TrainingHoG = Water_Train(1:2033,:); % Training
Set
Water_TestingHoG = Water_Train(2034:8132,:); % Testing
Set

%Green HoG DataSet
Green_Train =
Green_HoGTrain(randperm(size(Green_HoGTrain,1)),:);
Green_TrainingHoG = Green_Train(1:1328,:); % Training
Set
Green_TestingHoG = Green_Train(1329:5312,:); % Testing
Set


Appending Training set from each class and creating the
label for the training set to feed into the classifier

Training_Set = [ Concrete_TrainingSet;
Asphalt_TrainingSet; Color_TrainingSet;
Crack_TrainingSet; Dry_TrainingSet; Green_TrainingSet;
Water_TrainingSet; Oil_TrainingSet];

Label = cell(size(Training_Set,1),1);
Label(1:size(Concrete_TrainingSet,1)) = {'Concrete'};
Label((size(Concrete_TrainingSet,1)+1):(size(Asphalt_Tr
ainingSet,1)+size(Concrete_TrainingSet,1))) =
{'Asphalt'};
Label((size(Concrete_TrainingSet,1)+size(Asphalt_Traini
ngSet,1)+1) :
size(Concrete_TrainingSet,1)+size(Asphalt_TrainingSet,1
)+ size(Color_TrainingSet,1)) = {'Artificial Color'};
Label((size(Concrete_TrainingSet,1)+size(Asphalt_Traini
ngSet,1)+ size(Color_TrainingSet,1)+1) :
size(Concrete_TrainingSet,1)+...
```

```
        size(Asphalt_TrainingSet,1)+
size(Color_TrainingSet,1)+size(Crack_TrainingSet,1)) =
{'Crack'};
Label((size(Concrete_TrainingSet,1)+
size(Asphalt_TrainingSet,1)+
size(Color_TrainingSet,1)+size(Crack_TrainingSet,1))+1
: size(Concrete_TrainingSet,1)...
        + size(Asphalt_TrainingSet,1)+
size(Color_TrainingSet,1)+size(Crack_TrainingSet,1)+siz
e(Dry_TrainingSet,1)) = {'Dry Vegetation'};
Label((size(Concrete_TrainingSet,1)+
size(Asphalt_TrainingSet,1)+
size(Color_TrainingSet,1)+size(Crack_TrainingSet,1))+
(size((Dry_TrainingSet),1))+ 1 :
size(Concrete_TrainingSet,1)+...
        size(Asphalt_TrainingSet,1)+
size(Color_TrainingSet,1)+size(Crack_TrainingSet,1)+siz
e((Dry_TrainingSet),1)+ size(Green_TrainingSet,1)) =
{'Green vegetation'};
Label((size(Concrete_TrainingSet,1)+size(Asphalt_Traini
ngSet,1)+
size(Color_TrainingSet,1)+size(Crack_TrainingSet,1)+
size(Dry_TrainingSet,1)+size(Green_TrainingSet,1)+1):
size(Concrete_TrainingSet,1)...
    +size(Asphalt_TrainingSet,1)+
size(Color_TrainingSet,1)+size(Crack_TrainingSet,1)+
size(Dry_TrainingSet,1)+
size(Green_TrainingSet,1)+size(Water_TrainingSet,1)) =
{'Water'};
Label((size(Concrete_TrainingSet,1)+size(Asphalt_Traini
ngSet,1)+
size(Color_TrainingSet,1)+size(Crack_TrainingSet,1)+siz
e(Dry_TrainingSet,1)+size(Green_TrainingSet,1)+size(Wat
er_TrainingSet,1)...
    +1) :
(size(Concrete_TrainingSet,1)+size(Asphalt_TrainingSet,
1)+
size(Color_TrainingSet,1)+size(Crack_TrainingSet,1)+siz
e(Dry_TrainingSet,1)+
size(Green_TrainingSet,1)+size(Water_TrainingSet,1)+siz
e(Oil_TrainingSet,1)))= {'Oil'};

% Training the SVM Classifier for the hyperspectral
data set
Tic  % record time span for training the classifier
```

```
X = Training_Set;
Y = Label;
temp =
templateSVM('KernelFunction','rbf','KernelScale','auto'
,'BoxConstrain',1);
options = statset('UseParallel',true);

 [PMdl] =
fitcecoc(X,Y,'coding','onevsone','Learners',temp,'kFold
',10,...
    'ClassName',{'Concrete','Asphalt','Artificial
Color','Crack','Dry Vegetation',...
    'Green
vegetation','Water','Oil'},'FitPosterior',true,'CrossVa
l','on');
 Mdl = PMdl.Trained{1};
 toc

 %Validating the Classifier
 ValiInds = test(PMdl.Partition);  % Extract the test
indices
XVali = Asphalt_Train(ValiInds,:);
YVali = Y(ValiInds,:);
[labels,~,~,Posterior] = predict(Mdl,XVali);

idx = randsample(sum(ValiInds),60); %Number of data to
cross validate the performance of classifer
table(YVali(idx),labels(idx),Posterior(idx,:),...

'VariableNames',{'TrueLabels','PredictedLabels','Poster
ior'})


% Appending the testing set of respective classes and
creating ground truth for validating the classifier.
TestingSet = [Concrete_Test; Asphalt_Test; Color_Test;
Crack_Test;Dry_Test; Green_Test; Water_Test; Oil_Test];

GroundTruth = cell(size(TestingSet,1),1);
GroundTruth(1:size(Concrete_Test,1)) = {'Concrete'};
GroundTruth((size(Concrete_Test,1)+1):(size(Asphalt_Tes
t,1)+size(Concrete_Test,1))) = {'Asphalt'};
GroundTruth((size(Concrete_Test,1)+size(Asphalt_Test,1)
+1) : size(Concrete_Test,1)+size(Asphalt_Test,1)+
size(Color_Test,1)) = {'Artificial Color'};
```

```
GroundTruth((size(Concrete_Test,1)+size(Asphalt_Test,1)
+ size(Color_Test,1)+1) : size(Concrete_Test,1)+...
        size(Asphalt_Test,1)+
size(Color_Test,1)+size(Crack_Test,1)) = {'Crack'};
GroundTruth((size(Concrete_Test,1)+
size(Asphalt_Test,1)+
size(Color_Test,1)+size(Crack_Test,1))+1 :
size(Concrete_Test,1)...
        + size(Asphalt_Test,1)+
size(Color_Test,1)+size(Crack_Test,1)+size(Dry_Test,1))
= {'Dry Vegetation'};
GroundTruth((size(Concrete_Test,1)+
size(Asphalt_Test,1)+
size(Color_Test,1)+size(Crack_Test,1))+size(Dry_Test,1)
+ 1 : size(Concrete_Test,1)+...
        size(Asphalt_Test,1)+
size(Color_Test,1)+size(Crack_Test,1)+size(Dry_Test,1)+
size(Green_Test,1)) = {'Green vegetation'};
GroundTruth((size(Concrete_Test,1)+size(Asphalt_Test,1)
+ size(Color_Test,1)+size(Crack_Test,1)+
size(Dry_Test,1)+ size(Green_Test,1)+1):
size(Concrete_Test,1)...
    +size(Asphalt_Test,1)+
size(Color_Test,1)+size(Crack_Test,1)+size(Dry_Test,1)+
size(Green_Test,1)+size(Water_Test,1)) = {'Water'};
GroundTruth((size(Concrete_Test,1)+size(Asphalt_Test,1)
+
size(Color_Test,1)+size(Crack_Test,1)+size(Dry_Test,1)+
size(Green_Test,1)+size(Water_Test,1)...
    +1) : (size(Concrete_Test,1)+size(Asphalt_Test,1)+
size(Color_Test,1)+size(Crack_Test,1)+ size(Dry_Test,1)
+size(Green_Test,1)+size(Water_Test,1)+size(Oil_Test,1)
))= {'Oil'};


[Class,~,~,Posterior] = predict(Mdl,TestingSet);
% Classification
C = confusionmat(GroundTruth,Class); % Confusion Matrix
% plotting the true and predicted class after
classification
table(GroundTruth,Class,'VariableNames',{'TrueLabels','
PredictedLabels'})

% Calculation of precision and recall for single data.
% Calculation of Recall for each class
```

```
for i = 1:size(C,1)
    recall(i) = C(i,i)/sum(C(i,:));
end
%Calculation of Precision for each Class
for i = 1:size(C,1)
    precision(i) = C(i,i)/sum(C(:,i));
end
%Calculation of F-measure
for i = 1:size(C,1)
    f_Score(i) =
(2*recall(i)*precision(i))/(precision(i) + recall(i));
end


%Calculating ROC Curve For EACH Individual class and
plotting it together.
GT_Conc =
double(strcmp(GroundTruth,'Concrete'));%Ground Truth
for Class Concrete
PredictedConc =double(strcmp(Class,'Concrete'));
GT_Asphalt =
double(strcmp(GroundTruth,'Asphalt'));%Ground Truth for
Class Asphalt
PredictedAsphalt =double(strcmp(Class,'Asphalt'));
GT_Color = double(strcmp(GroundTruth,'Artificial
Color'));%Ground Truth for Class Color
 PredictedColor =double(strcmp(Class,'Artificial
Color'));
GT_Crack = double(strcmp(GroundTruth,'Crack'));%Ground
Truth for Class Crack
PredictedCrack =double(strcmp(Class,'Crack'));
GT_Veg = double(strcmp(GroundTruth,'Green
vegetation'));%Ground Truth for Class Green vegetation
 PredictedVeg =double(strcmp(Class,'Green
vegetation'));
GT_Water = double(strcmp(GroundTruth,'Water'));%Ground
Truth for Class Water
 PredictedWater =double(strcmp(Class,'Water'));
GT_Oil = double(strcmp(GroundTruth,'Oil'));%Ground
Truth for Class Oil
PredictedOil =double(strcmp(Class,'Oil'));
GT_Dry = double(strcmp(GroundTruth,'Dry
Vegetation'));%Ground Truth for Class Dry Vegetation
PredictedDry =double(strcmp(Class,'Dry Vegetation'));
```

```
%Plotting of ROc Curve
[XConc,YConc,~,AUC_Conc,OPTROCPT_Conc] =
perfcurve(GT_Conc,Posterior(:,1),1); %Class Concrete
[XAsphalt,YAsphalt,~,AUC_Asphalt,OPTROCPT_Asphalt]    =
perfcurve(GT_Asphalt,Posterior(:,2),1);%Class Asphalt
[XColor,YColor,~,AUC_Color,OPTROCPT_Color]        =
perfcurve(GT_Color,Posterior(:,3),1);%Class Artificial
Color
[XCrack,YCrack,~,AUC_Crack,OPTROCPT_Crack]        =
perfcurve(GT_Crack,Posterior(:,4),1);%Class Crack
[XVeg,YVeg,~,AUC_Veg,OPTROCPT_Veg]        =
perfcurve(GT_Veg,Posterior(:,6),1);% Class
GreenVegetation
[XWater,YWater,~,AUC_Water,OPTROCPT_Water]        =
perfcurve(GT_Water,Posterior(:,7),1);%Class Water
[XOil,YOil,~,AUC_Oil,OPTROCPT_Oil]        =
perfcurve(GT_Oil,Posterior(:,8),1);%Classs Oil
[XDry,YDry,~,AUC_Dry,OPTROCPT_Dry]        =
perfcurve(GT_Dry,Posterior(:,5),1);% Class Dry
Vegetation

% Plotting Precision recall curve
[XConc1,YConc1,~, AUCConc1, OPTROCPT_Conc1] =
perfcurve(GT_Conc,Posterior(:,1),1,'xCrit',
'reca','yCrit','prec'); % Class Concrete
[XAsph1,YAsph1,~, AUCAsph1, OPTROCPT_Asph1] =
perfcurve(GT_Asphalt,Posterior(:,2),1,'xCrit',
'reca','yCrit','prec');% Class Asphalt
[XColor1,YColor1,~, AUCColor1, OPTROCPT_Color1] =
perfcurve(GT_Color,Posterior(:,3),1,'xCrit',
'reca','yCrit','prec');% Class Artificial Color
[XCrack1,YCrack1,~, AUCCrack1,OPTROCPT_Crack1] =
perfcurve(GT_Crack,Posterior(:,4),1,'xCrit',
'reca','yCrit','prec');% Class Crack
[XDry1,YDry1,~, AUCDry1, OPTROCPT_Dry1] =
perfcurve(GT_Dry,Posterior(:,5),1,'xCrit',
'reca','yCrit','prec');% Class Dry vegetation
[XGreen1,YGreen1,~, AUCGreen1,OPTROCPT_Green1] =
perfcurve(GT_Veg,Posterior(:,6),1,'xCrit',
'reca','yCrit','prec');% Class Green Vegetation
[XWater1,YWater1,~, AUCWater1,OPTROCPT_Water1] =
perfcurve(GT_Water,Posterior(:,7),1,'xCrit',
'reca','yCrit','prec');% Class Water
```

```
[XOil1,YOil1,~, AUCOil1,OPTROCPT_Oil1] =
perfcurve(GT_Oil,Posterior(:,8),1,'xCrit',
'reca','yCrit','prec');% Class Oil
```