ISTANBUL TECHNICAL UNIVERSITY ★ INFORMATICS INSTITUTE

HYPERSPECTRAL IMAGE COMPRESSION USING GRAPH SIGNAL PROCESSING AND WAVELET-BASED SPECTRAL DECORRELATION

M.Sc. THESIS

Indrit NALLBANI

Department of Applied Informatics

Applied Informatics Programme

JUNE 2019

ISTANBUL TECHNICAL UNIVERSITY ★ INFORMATICS INSTITUTE

HYPERSPECTRAL IMAGE COMPRESSION USING GRAPH SIGNAL PROCESSING AND WAVELET-BASED SPECTRAL DECORRELATION

M.Sc. THESIS

Indrit NALLBANI
(708171012)

Department of Applied Informatics

Applied Informatics Programme

Thesis Advisor: Doç.Dr. Behçet Uğur TÖREYİN

JUNE 2019

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ BİLİŞİM ENSTİTÜSÜ

ÇİZGE İŞARET İŞLEME VE DALGACIK TABANLI İZGE ILINTISIZLEŞTIRMEYE SPEKTRAL DEKORELASYON DAYALI HİPERSPEKTRAL İMGE SIKIŞTIRMA

YÜKSEK LİSANS TEZİ

Indrit NALLBANI
(708171012)

Bilişim Uygulamaları Anabilim Dalı

Bilişim Uygulamaları Programı

Tez Danışmanı: Doç.Dr. Behçet Uğur TÖREYİN

HAZİRAN 2019

Indrit NALLBANI, a M.Sc. student of ITU Informatics Institute with ID 708171012 successfully defended the thesis entitled "HYPERSPECTRAL IMAGE COMPRESSION USING GRAPH SIGNAL PROCESSING AND WAVELET-BASED SPECTRAL DECORRELATION", which he prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor : **Doç.Dr. Behçet Uğur TÖREYİN** ....
Istanbul Technical University

Jury Members : **Prof. Dr. Muhammed Oğuzhan KÜLEKCİ** ....
Istanbul Technical University

Doç.Dr. Gökhan BİLGİN ....
Yıldız Technical University

Date of Submission : 3 May 2019
Date of Defense : 13 June 2019

To my parents, who understood the importance of education and made the impossible to provide us with it. To my brother and sister, who have always been there in good and bad days.To my family whose sacrifices made me who I am. I dedicate this thesis to you.

FOREWORD

I would like to express my gratitude to Dr. Behçet Uğur Töreyin for the guidance provided throughout the completion of this thesis. I am very grateful for his teachings, motivation, and patience he has shown me. He has been nothing but kind and respectful and I can't thank him enough for what he has done for me.

I gratefully acknowledge the financial support of Vodafone Turkey for conducting this study under project no. ITUVF20180901P04 conducted jointly by Informatics Faculty, Istanbul Technical University (ITU) and Vodafone within the context of ITU Vodafone Future Lab R&D program.

13 June 2019 Indrit NALLBANI

TABLE OF CONTENTS

Page

FOREWORD .... **ix**
TABLE OF CONTENTS .... **xi**
ABBREVIATIONS .... **xiii**
LIST OF TABLES .... **xv**
LIST OF FIGURES .... **xvii**
SUMMARY .... **xix**
ÖZET .... **xxi**
1. INTRODUCTION **1**
  1.1 Literature Review 3
    1.1.1 Lossless compression 3
    1.1.2 Lossy compression 5
    1.1.3 Graph signal processing 6
  1.2 Purpose of Thesis 6
2. METHODOLOGY **9**
  2.1 Integer coefficient based Discrete Wavelet Transforms 9
    2.1.1 Spectral decorrelation, integer based forward transformations 10
  2.2 Discrete Wavelet Transform 10
    2.2.1 Four-level discrete wavelet transform 11
    2.2.2 Five-level discrete wavelet transform 12
  2.3 Graph Fourier Transform 13
  2.4 Hyperspectral Image Reconstruction 16
  2.5 JPEG-2000 Compression Scheme 18
  2.6 Peak Signal to Noise Ratio (PSNR) 18
3. DATASETS AND EXPERIMENTAL RESULTS **21**
  3.1 Jasper Ridge HSI 22
  3.2 Low Altitude HSI 27
  3.3 Lunar Lake HSI 30
  3.4 Pavia University HSI 34
  3.5 Botswana HSI 37
4. CONCLUSIONS **43**
REFERENCES **45**
CURRICULUM VITAE **49**

ABBREVIATIONS

HSI : Hyperspectral Image Compression
GSP : Graph Signal Processing
GFT : Graph Fourier Transform
DWT : Discrete Wavelet Transform
DW : Wavelet Transform
BPS : Bit Per Sample
PSNR : Peak-signal-to-noise-ratio
3-D : Three Dimensional
LZW : Lempel - Ziv - Welch
BIL : Band-interleaved-by-line
LUT : Lookup Table
KLT : Karhunen-Loeve-Transformation
MSE : Mean Square Error

LIST OF TABLES

Page

Table 2.1 : Transformation coefficients used for integer discrete wavelet transforms. **10**
Table 3.1 : Hyperspectral Image Datasets. Images are captured with different imaging systems and they have different band number. For our experiments, we will use only a portion of the images, more specifically, 256×256. .... **22**
Table 3.2 : Comparison of the two methods for BPS = 0.1 applied to the Jasper Ridge Hyperspecral Image.......... **23**
Table 3.3 : Comparison of the two methods for BPS = 0.3 applied to the Jasper Ridge Hyperspecral Image.......... **24**
Table 3.4 : Comparison of the two methods for BPS = 0.5 applied to the Jasper Ridge Hyperspectral Image.......... **24**
Table 3.5 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.1.......... **25**
Table 3.6 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.3.......... **25**
Table 3.7 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.5.......... **26**
Table 3.8 : PSNR (dB) vs BPS for Jasper Ridge Hyperspectral Image. **26**
Table 3.9 : Comparison of our compression scheme with other methods for Jasper Ridge HSI.......... **27**
Table 3.10 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.1.......... **28**
Table 3.11 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.3.......... **29**
Table 3.12 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.5.......... **29**
Table 3.13 : PSNR Vs BPS for Low Altitude Hyperspecral Image. **29**
Table 3.14 : Comparison of our compression scheme with other methods for Low Altitude HSI.......... **31**
Table 3.15 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.1.......... **32**
Table 3.16 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.3.......... **32**
Table 3.17 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.5.......... **33**
Table 3.18 : PSNR (dB) Vs BPS for Lunar Lake Hyperspecral Image. **33**

Table 3.19 : Comparison of our compression scheme with other methods for Lunar Lake HSI **34**
Table 3.20 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.1 ..... **35**
Table 3.21 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.3 ..... **35**
Table 3.22 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.5 ..... **36**
Table 3.23 : PSNR Vs BPS for Pavia University. **36**
Table 3.24 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.1 ..... **38**
Table 3.25 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.3 ..... **38**
Table 3.26 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.5 ..... **39**
Table 3.27 : Results of our coding scheme over different bitrates **40**
Table 3.28 : Comparison of results for different hyperspectral Images **40**

LIST OF FIGURES

Page

Figure 1.1 : Hyperspectral imaging sensor. Wavelength detectors capture a continuous range of wavelength reflected from the remote area/ **1**
Figure 1.2 : 3D Hyperspectral Image. Each pixel contains a continuous bands of wavelengths. **2**
Figure 1.3 : General method of compressing/decompressing a hyperspectral image.. **3**
Figure 2.1 : Hyperspectral Image decomposition using Discrete Wavelet Transform.. **9**
Figure 2.2 : A four-level Discrete Wavelet Transform of AVIRIS hyperspectral images. Boxes in red show that the images are compressed using JPEG-2000 compression scheme and the last fourteen bands are reproduced using Graph Fourier Transform................................. **11**
Figure 2.3 : A four-level Discrete Wavelet Transform of AVIRIS hyperspectral images. Boxes in red show that the images are compressed using JPEG-2000 compression scheme and the last fourteen bands are reproduced using Graph Fourier Transform................................. **12**
Figure 2.4 : A five-level Discrete Wavelet Transform decomposes the image five times thus creating six cubes from the original image. The bands from H, LH, LLH, LLLH, LLLLL cubes are compressed in lossy mode with low and high compression ratios and Graph Fourier Transform is applied to LLLLH bands Boxes in red show that the images are compressed using JPEG-2000 compression scheme and the last seven bands are reproduced using Graph Fourier Transform. **13**
Figure 2.5 : A five-level Discrete Wavelet Transform decompose the image five times thus creating six cubes from the original image. H, LH, LLH, LLLH, LLLLH. We apply Graph Fourier Transform onto the LLLLL bands Boxes in red show that the images are compressed using JPEG-2000 compression scheme and the last seven bands are reproduced using Graph Fourier Transform................................. **14**
Figure 2.6 : A graph representation of a 7 image.. **15**
Figure 2.7 : Hyperspectral Image reconstruction using using inverse discrete wavelet transform... **17**
Figure 2.8 : A five-level Inverse Discrete Wavelet Transform of the hyperspectral images. **17**
Figure 2.9 : A five-level Inverse Discrete Wavelet Transform of the hyperspectral images. ....
18
Figure 3.1 : Jasper Ridge band 10, 50, 100... **23**

Figure 3.2 : A comparion of different filter for Jasper Ridge hyperspectral image. 26
Figure 3.3 : Low Altitude band 10, 50, 100. .... **28**
Figure 3.4 : A comparion of different filter for Low Altitude hyperspectral image. 30
Figure 3.5 : Lunar Lake band 10, 50, 100. .... **31**
Figure 3.6 : A comparison of different filter for Lunar Lake hyperspectral image. We see that 5/3 outputs the best PSNR value compared to the other filters. **33**
Figure 3.7 : Pavia University band 10, 50, 100. .... **35**
Figure 3.8 : A comparion of different filter for Pavia University hyperspectral image. **37**
Figure 3.9 : Botswana HSI band 10, 50, 100. .... **37**
Figure 3.10 : A comparion of different filter for Pavia University hyperspectral image. 39

HYPERSPECTRAL IMAGE COMPRESSION USING GRAPH SIGNAL PROCESSING AND WAVELET-BASED SPECTRAL DECORRELATION

SUMMARY

In recent years, remote sensing is playing a very important role in the military and agriculture areas of research. Using sensors placed onboard in a remote object, we can analyze the soil contents or finding mineral deposits in a particular area of the world. In order to do this, large volumetric data are captured in the form of photography using hyperspectral imaging. Due to their large size, there exists a hardware bottleneck that doesn't allow us to record a lot of data. Therefore, it is very important that we compress the hyperspectral images while preserving the quality in order to capture more images onboard.

In this thesis, we introduce a novel method of compressing hyperspectral image(HSI) while preserving the quality. In order to achieve an efficient compression, the HSI is spectrally and spatially decorrelated. We apply a five-level Discrete Wavelet Transform (DWT) to spectrally decorrelate the image and losslessly compressed the detail coefficients using JPEG 2000.

We apply graph Fourier transform to the last seven approximation coefficients and save only a few coefficients with the highest values that will enable us to reconstruct the image back. The novelty of this method relies on transforming band coefficients of a signal on the graph . We reconstruct the lowest bands by taking the Inverse Graph Fourier Transform and reconstruct the HSI using Inverse Discrete Fourier Transform .

The quality of the HSI is measured using PSNR metrics because it is the most commonly used metrics for comparing losslessly compressed images. Two types of Hyperspectral Image datasets are used to implement our coding scheme: Airborne Visible Imaging Spectrometer (AVIRIS) dataset and Hyperion dataset.

ÇİZGE İŞARET İŞLEME VE DALGACIK TABANLI İZGE ILINTISIZLEŞTIRMEYE SPEKTRAL DEKORELASYON DAYALI HİPERSPEKTRAL İMGE SIKIŞTIRMA

ÖZET

Son yıllarda, uzaktan algılama askeri ve tarım araştırma alanlarında çok önemli bir rol oynamaktadır. Uzak bir nesneye yerleştirilmiş sensörleri kullanarak, toprak içeriklerini analiz edebilir veya dünyanın belirli bir bölgesinde mineral tortuları bulabiliriz. Bunu yapmak için büyük hacimler veriler, hiperspektral görüntüleme kullanılarak fotoğraf şeklinde yakalanır. Donanım kısıtlamalarından ötürü bu büyük boyutlu fotoğraflardan oluşan veri setlerini depolayamiyoruz.Bu yüzden, hiperspektral imgeleri kalitelerinden odun vermeden şıkıştırabilmek çok önemlidir.

Uzaktan algılama, temas etmeden hedef alandan yansıtılan elektromanyetik radyasyonu kaydederek bir alanın özelliklerini ölçme işlemi olarak tanımlanır. Ticari uzaktan algılama hiperspektral görüntüleme sistemlerinin iyileştirilmesi ile, çok sayıda alanların görüntülenmesi araştırmacılar için çok kolay hale gelmiştir ve tarım, çevre ,askeri uygulamalar, dünya minerallerini keşfetme, orman izleme gibi alanlarda kullanılmasına olanak sağlamıştır.

Spektral yansıma hiperspektral görüntüleme sistemi tarafından yakalanır ve dispersiyon elemanları yansıyan ışığı farklı bitişik dalga boylarına ayırmak için kullanılır. Her dalga boyu belirli bir dalga boyu dedektörü tarafından yakalanır. Spektral yansıma, çalisilan nesne veya alanlar icin parmak izi olarak davranir, cunku farklı malzemeler isigi . Spektral yansıma şekli her nesne veya malzeme, böylece nesneleri farklı anlatmak kolay inceleniyor için farklıdır.

Hiperspektral görüntüler, görüntüyü sürekli bir elektromanyetik spektrumla kaydedilmiş görüntüler şeklinde saklayarak büyük hacimler verilerdir. Bu, her pikselin 16 veya 12 bit olarak depolandığı bir 3B küpte biçiminde sonuçlanir. Büyük boyutu, bircok veriyi ayni anda kaydedebilmeyi olanaksiz hale getiren bir darboğaz yaratiyor. iste bu nedenle, daha fazla goruntuyu saklayabilmek icin hiperspektral sikistirmanin onemi ortaya cikiyor.

Uzlamsal ve spektral artiklik gibi iki turu olan, Hiperspektral Goruntu (HSI) icindeki gereksiz bilgileri kaldirarak olusturulur. Bunlardan ilki yedek piksel arasindaki uzlamsal etki alaninda olusurken ikincisi ve en buyuk artiklik kaynagi olani ise bitisik bandlarda yakalanan dalga boyunda olusur.

Spektral etki alanında, biz ayrık dalgacık dönüşümler kullanarak hiperspektral görüntü dekorelasyon yapiyoruz. Biz birden fazla filtre dekorelasyon yapılmış katsayılarını tamsayı katsayıları içine kayan sayılar kaynaklanan kaybı en aza indirmek için kaldırmak için kullandık. Görüntünün decorrelated olduğu iki etki alanı vardır. Spektral etki alanında, piksel arasındaki korelasyon azaltmak için görüntülerin decorrelated beş kez ayrı Wavelet dönüşümler kullandık. Görüntü ters ayrık Wavelet Transform kullanılarak yeniden oluşturulur. Uzamsal etki alanında, sadece birkaç

katsayılar ile sinyali yeniden oluşturmak için Graph Fourier Transform kullanıyoruz. Biz Laplacian eigenvektörler Fourier temeli olarak hareket grafik Laplacian inşa. Grafik sinyali ters grafik Fourier dönüşümü alarak kullanılarak yeniden oluşturulur. Katsayılar sayısına bağlı olarak, biz atmak sinyal katsayıları ters Fourier eigenbasis kullanılarak tahmin edilir.

Bu tezde, kaliteyi korurken Hiperspektral görüntüyü (HSI) şıkıştırmak için yeni bir yöntem tanıtıyoruz. Verimli bir sıkıştırma elde etmek için, HSI ışıksal ve yüzeysel olarak ayrıştırılır. Imgeleri spektral olarak ayrıştırmak icin beş aşamalı Discrete Wavelet Transform (DWT) uygulandi ve katsayılar JPEG 2000 kullanilarak kayipsiz bir sekilde sikistirildi.

En küçük yedi katsayiya graf Fourier transformu uygulandi ve imgeleri geri dönüstürmek için en yuksek degere sahip birkaç katsayı kaydedildi. Bu metodun yeniligi graftaki bir sinyalin frekans katsayılarınıin dönşturulmesiyle ilgilidir. Frekans bandındaki düşük katsayılar kullanilarak Inverse Discrete Wavelet Transform (IDWT) ile imgeler tekrar elde edildi. Hiperspektral imgeler de Inverse Discrete Wavelet Transform geri elde edildi.

Biz beş seviyeli ayrık dalgacık dönüşümü uyguladıktan sonra kodlama şeması kullanıyoruz. Bizim kodu dekorelasyondan cikan en son yedi bantlarıda uyguluyoruz.

Biz son yedi yaklaşma katsayıları için grafik Fourier dönüşümü uygulamak ve bize görüntüyü geri yeniden oluşturmasını sağlayacak en yüksek değerlerle yalnızca birkaç katsayıları kaydediyoruz. Bu yöntem, yenilenen grafik üzerinde bir sinyal bant katsayıları dönüştürmesine dayanır.

Grafikler, karmaşık veri yapılarını modellemek ve veri içeriği arasındaki etkileşimlerden yararlanmak için bize izin verir. Grafik Sinyal Işleme araçlarını kullanarak, grafik düğümlerini bir grafiğe bağlı sinyaller olarak modelleyerek ayrıca sinyal filtreleme ve Fourier Transform gibi klasik sinyal işleme araçlarını grafik üzerine uygulayabiliriz.

Grafik sinyal işleme yöntemleri ayrıca piksellerin normal 2B ızgaralar oluşturacağı resimler ve videolar gibi düzenli grafiklere dağıtılır. Görüntüler, piksellerin grafiğin düğümlerini oluşturabileceği ve yakınlığı kenarları olarak kullanılabileceği normal grafikler olarak görüntülenebilir.

İki adet hiperspektral görüntü seti kullandık: AVIRIS ve HYPERION veri kümeleri, genel olarak kullanılabilen veriler. Hiperspektral görüntüler, hesaplamalı karmaşıklığı azaltmak için daha küçük bir boyuta kırpılmıştır. Kodlama şemamiz ile elde edilen sonuçlar, yontemimizin JPEG 2000 sıkıştırma şeması ile karşılaştırıldığında, görüntü kalitesinde diğer bazı yöntemlerle birlikle bir gelişme gösterir. Hiperspektral görüntüleri sıkıştırmak için tercih ettiğimiz yönteme göre çok iyi PSNR değerleri elde etmeyi başardık. HSI kalitesi, kayıpsız sıkıştırılmış görüntüleri karşılaştırmak için en sık kullanılan ölçümlerden olan PSNR metrikleri ile ölçüldü.

Kodlama düzenimizin sonuçlarını daha da iyileştirmek ve diğer yöntemlerle karşılaştırmak için, aynı küp boyutuna sahip aynı Hiperspektral görüntülerin standart bir veri kümesi olarak kullanılması gerektiğini düşünüyoruz. Araştırmacılar çoğu küpün hangi kısmını ve hangi uçuş numarasını kullandılar belirtmiyor beri.

Gelecekteki çalışmanıza göre, yeniden oluşturulmuş Hiperspektral görüntünün kalitesini artırmak için grafik filtreleme, grafik dalga boyları vb. gibi daha fazla grafik sinyal Işleme aracı kullanırız. Daha fazla araştırma hiperspektral görüntüleri sıkıştırmak için başka yollar bulma yapılacaktır.

1. INTRODUCTION

Remote sensing is defined as the process of measuring the characteristics of an area by recording the electromagnetic radiation that is reflected from the target area without being in contact with it. With the improvement of commercial remote sensing hyperspectral imaging systems, capturing a large number of areas has become very easy for researchers to study and the application spread in fields, such as agriculture, environmental and forest monitoring, military applications and the exploration of Earth minerals.

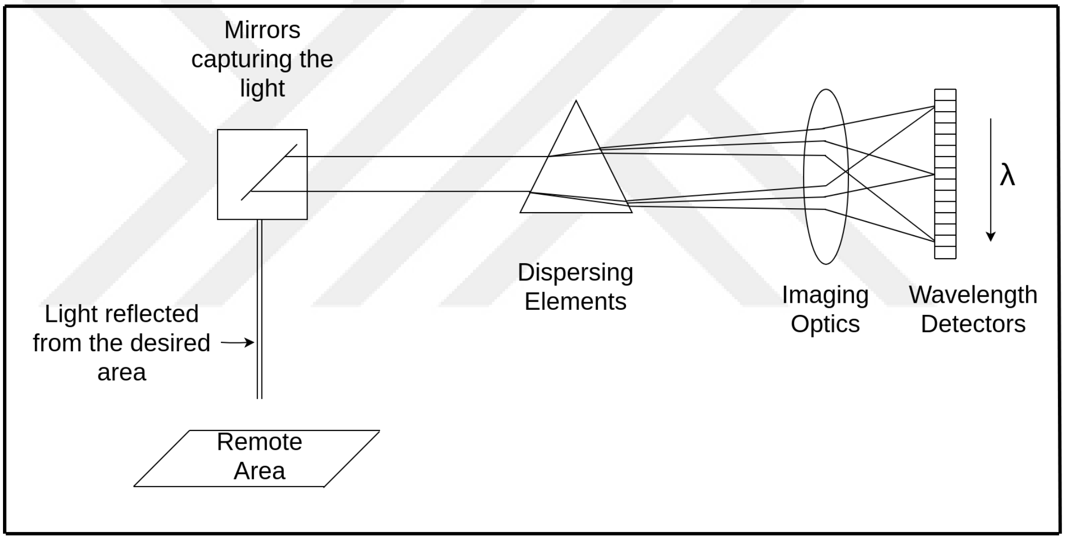


Figure 1.1 : Hyperspectral imaging sensor. Wavelength detectors capture a continuous range of wavelength reflected from the remote area/

In Fig 1.1, the spectral reflectance is captured by the hyperspectral imaging system and dispersing elements are used to separate the reflected light into different adjacent wavelengths. Each wavelength is captured by a specific wavelength detector. The spectral reflectance acts as a fingerprint for the object or the area to be studied because different materials scatter or absorb the light in a different manner from other objects. The shape of the spectral reflectance is different for every objects or material that is being studied so it is easy to tell objects apart.

Teke *et al.*, point out the importance of using hyperspectral imaging in crop yield estimation, plant monitoring, etc [1], The HSI increases the precision and accuracy of the knowledge related to crop yields and plant disease detection. The content of the soil minerals can be detrimental to the crop yield causing small harvest and famine. Also, changes in leaf colors may suggest an infection occurring in plant and leaves. Obtaining this important information early can help farmers taking action before being too late to do something. Govender *et al.* has shown that remote sensing can reduce labor time and cost of mapping the vegetation fields [2].

Briottet *et al.* have pointed out the important use of hyperspectral images in military applications. Hyperspectral images are able to capture objects that can hide from normal cameras due to the continuous band of wavelengths that are able to differentiate object with similar wavelengths. The experiment conducted bt Briottet and his colleagues were able to detect hidden targets that were not captured by the human eye.

Hyperspectral images are large volumetric data stored in form of images whose image is saved with a continuous electromagnetic spectrum. This results in a 3D cube where each pixel is stored in 16 or 12 bits. In Fig 1.2, we have the spectral response of a pixel obtained by the Airborne Visible Infra Red Imaging Spectrometer (AVIRIS). The response of the hyperspectral images obtained by AVIRIS contains 224 wavelengths whereas Hyperion hyperspectral images contain 220 wavelengths.

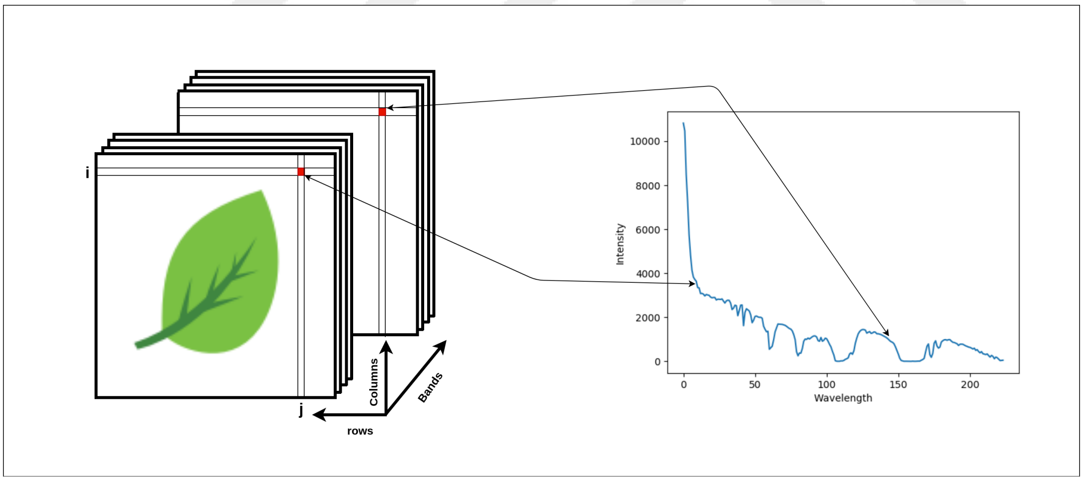


Figure 1.2 : 3D Hyperspectral Image. Each pixel contains a continuous bands of wavelengths.

1.1 Literature Review

Hyperspectral image compression is achieved by removing redundant information in the hyperspectral image. There are two types of redundancy in an HSI: spatial and spectral redundancy. Spectral redundancy, which is the biggest source of redundancy in a hyperspectral image, occurs in the wavelength captured in the adjacent bands. Spatial redundancy occurs in the spatial domain between pixels.

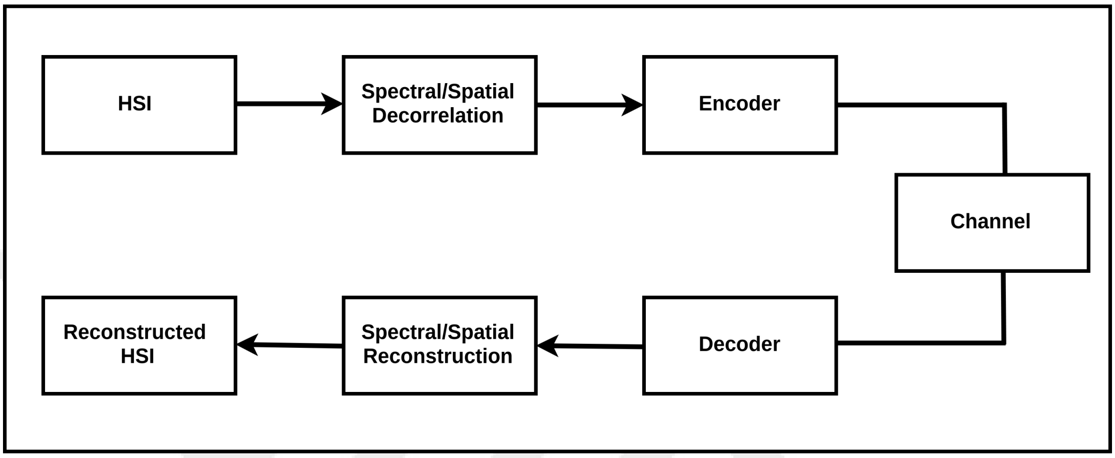

HSI
Spectral/Spatial Decorrelation
Encoder
Channel
Decoder
Spectral/Spatial Reconstruction
Reconstructed HSI

Figure 1.3 : General method of compressing/decompressing a hyperspectral image.

There have been several studies done comparing hyperspectral image methods. Bilgin *et al.* have compared several methods for compressing hyperspectral images as 3D cubes as well as treating the whole image as a wave. They found out that the method that produces the best result is SPIHT [3].

Most compression techniques are categorized into two categories: lossy and lossless [4]. Lossless compression schemes are used in very important fields such as the military and medicine where image quality and accuracy are very important. The reconstructed image is identical to the original image but the compression ratio is very low compared to lossy schemes. In lossy schemes, we have a loss of image quality and only an approximation of the original image is obtained. This results in a higher compression ratio. This scheme is generally used in audio and video files where the degradation of the original signal not very noticeable.

1.1.1 Lossless compression

There are several algorithms that are used to compress an image in a losses manner. Lempel-Ziv-Welch (LZW) algorithm compresses the files by reading every token separately and combines them to strings. The strings are then coded into shorter codes and the repeated string is compressed using the assigned code in the code table. [5]

Zhang *et al.*, deploy a statistical-based algorithm for compressing the hyperspectral image in a lossless manner. Huffman coding encodes tokens or characters of the input files by assigning them a variable-length code. The length of these codes depends on the frequency of the characters in the data. The most frequent characters get a small code where the least frequent characters are assigned a large code [6].

J.Mielikainen *et al.* have used arithmetic coding to losslessly compress their hyperspectral image. The method consists of three steps: clustering, prediction, and coding [7]. In the clustering stage, clusters are formed in such a way that the entropy is minimized. The prediction step is achieved a linear predictor and the result between original and reconstructed pixel values are entropy encoded with the help of a range coder.

Lookup Tables (LUT) are another example of lossless compression schemes. In this scheme, LP Tables are used to search faster previous bands faster for the pixel with an equal value of the pixel to be coded. This method has outperformed other schemes of losslessly compressing images in the band interleaved by line [8].

Töreyin *et al.*, in their paper, have studied the compression ratio gains of integer coefficient discrete wavelet transforms. The hyperspectral images are spectrally decorrelated using integer-based discrete wavelet transforms in order to reduce the correlation between the bands. They have shown that decorrelating the hyperspectral images in the spectral domain with integer coefficients discrete wavelet transforms their compression rations [9].

In another paper, Töreyin uses the fractional wavelet transform instead of discrete wavelet transforms. Fractional wavelet transforms decomposes the signal into signals with different lengths. To achieve their high compression ratio by spectrally decorrelating the hyperspectral image using an unbalanced lazy lifting structure [10].

A.Karaca *et al.* have proposed a bimodal conventional recursive least squares based predictive method for compressing the hyperspectral images in lossless mode. Their method consist on predicting each pixel value by adding the dot product of input and weight vectors. Their method achieves good compression ratios and it is achieved with a lower computation time [11].

Lossless compression of the hyperspectral images is very important in the field of medicine where the accuracy and the perfect reconstruction is of high importance. However, other fields such as graphics, audio, and video domains can tolerate the loss of signal. The signal can be reconstructed with part of the original signal. This results in a higher compression ratio and achieves the desired result. Below are shown some of the lossy compression schemes.

1.1.2 Lossy compression

Lossy compressions schemes are based on transform coding. It allows for more image compression and hyperspectral image on-board capturing. Transform coding techniques are based on spectral and spatial decorrelation since the correlation occurs in the spectrum axis of the hyperspectral image.

Mihaela *et al.* uses the Karhunen-Loeve Transformation (KLT) to spectrally decorrelate the hyperspectral image and Discrete Wavelet Transform (DWT) to spatially decorrelate the hyperspectral image and encode it using JPEG 2000. This method can compress the HSI with a high compression ratio while preserving quality but it suffers from computational complexity [12].

S.Lim *et al.* have constructed a 3D set partition embedded block to spatially and spectrally decorrelate the HSI using a 3D Discrete Wavelet Transform. This method has proved to compress with high compression ration while preserving the rate distortion. [13].

Z.Gundogar *et al.* have developed a compression technique named tridiagonal folded matrix enhanced multivariance products representation (TFEMPR). The authors treat the image as a matrix and use the concept of a folded matrix to binary decompose a matrix resulting in low computational complexity and low distortion ratio [14].

Julide *et al.*, have used Haar filter to spectrally decorrelate the hyperspectral images and compressing it with sparse coding. The hyperspectral image is decomposed using the Haar filter and is compressed using online dictionary learning [15].

H.Jawdhari *et al.*, in his thesis, has proposed a method of compressing hyperspectral images in lossy mode. The hyperspectral image is spectrally decorrelated using discrete wavelet transform and the low bands are compressed using sparse representatins whereas high bands are compressed using JPEG 2000. [16].

1.1.3 Graph signal processing

Graphs allow us to model complex data structures and exploit the interactions among the data content. Using Graph Signal Processing tools, we can model graph nodes as signals relying on a graph and apply classical signal processing tools, such as signal filtering and Fourier Transform onto graphs.

In machine learning tasks, Zhu *et al.* uses Graph Signal Processing to estimate labels in unsupervised learning problems [17]. Sandryhaila *et al.*, design graph filters applied to sensor malfunction and data classification of partially labeled data [18]. Graph Signal Processing is also used in recommendation systems. By effectively exploiting the connection between the attributes, Huang *et al.*, have increased the accuracy of predicting similar attributes [19].

Graph Signal Processing has also been deployed into regular graphs such as images and videos where pixels form regular 2D grids. Images can be viewed as regular graphs where pixels form the nodes of the graph and their proximity can be used as edges. G.Cheung *et al.* have used Graph Signal Processing tools, such as Graph Fourier Transform and graph spectral filtering for image compression, image filtering, etc [20]. W.Hu *et al.*, use GSP to compress piecewise smooth images by minimizing the representation cost for each pixel block [21].

There are several works done on using Graph Signal Processing to compress Hyperspectral Images. In [22], A.Ortega *et al.*, use graph wavelets to compress hyperspectral images by decorrelating the hyperspectral images in the spatial and spectral domain. The images are divided into groups of several bands with a similar spectral response and encode it.

1.2 Purpose of Thesis

In this thesis, we are introducing a novel lossy compression scheme that uses Discrete Wavelet Transform and Graph Fourier Transform to compress hyperspectral images. In the spectral domain, we decorrelate the hyperspectral image using Discrete Wavelet Transforms. We have used several filters to lift the decorrelated coefficients into integer coefficients in order to minimize the loss arising from floating numbers. There are two domains in which the image is decorrelated. In the spectral domain, we have used Discrete Wavelet Transforms where the images are decorrelated five times in order to reduce the correlation among the pixels. The image is reconstructed using Inverse Discrete Wavelet Transform. In the spatial domain, we use Graph Fourier Transform to reconstruct the signal with only a few coefficients. We construct the graph Laplacian where the Laplacian eigenvectors act as Fourier basis. The graph signal is reconstructed using by taking the inverse graph Fourier transform. Depending on the number of coefficients, the signal coefficients that we discard are estimated using the inverse Fourier eigenbasis. The coding scheme has low computational complexity and it can be applied onto onboard systems. Our aim was to research new methods of compressing hyperspectral images.

2. METHODOLOGY

In this chapter, we will explain the method used for compressing the hyperspectral images. We will start by decorrelating the images using integer-to-integer discrete wavelet transform. We have used several filters such as Haar, 9/7-M, 2/6, 2/10 and 5/3. The approximation details of the five-level DWT are approximated using Graph Fourier Transform. In the end, we reconstruct the images using the inverse integer discrete wavelet transform.

2.1 Integer coefficient based Discrete Wavelet Transforms

In order to increase the compression ratio while preserving the image quality, we must decorrelate the hyperspectral image in the spatial and the spectral domains. In the spectral domain, we apply integer based discrete wavelet transform and we split the original image into two cubes of detail and approximation coefficients.

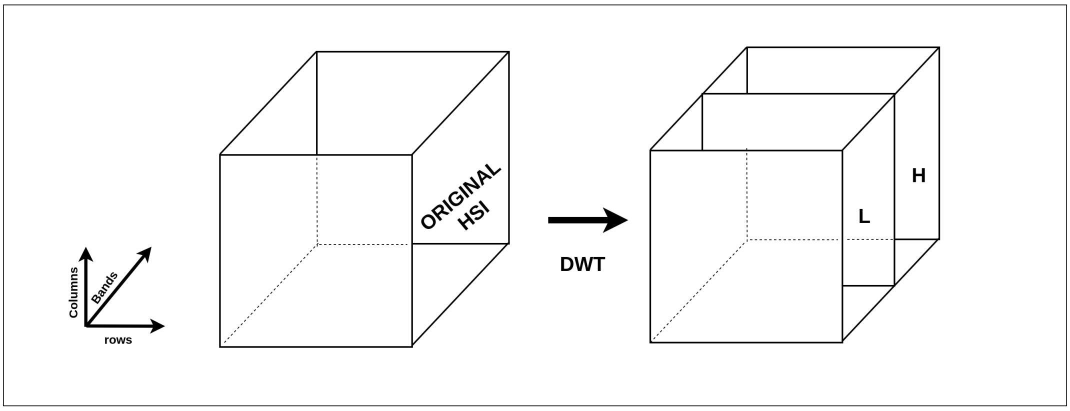


Figure 2.1 : Hyperspectral Image decomposition using Discrete Wavelet Transform.

The discrete wavelet transform decomposition of the image is shown in figure 2.1. We have applied high pass and low pass filters and we have decorrelated the image in the spectral domain. There are several integers to integer wavelet filters used for this step of the method such as Haar, 2/6, 2/10, 5/3, CDF 9/7-m.

David Adams, in his doctorate thesis, offers several integer wavelets transform based on the lifting framework [23]. Given an input signal $x[i]$, we split the signal into highpass and lowpass subands and denote them as $d_0[i] = x[2i+1]$ and $s_0[i] = x[2i]$.

2.1.1 Spectral decorrelation, integer based forward transformations

Haar filter transformation is one of the easiest and the most used for compression purposes. Its computational complexity is very low since it only needs two operations to transform the input signal.

Table 2.1 : Transformation coefficients used for integer discrete wavelet transforms.

Filter	Formula
Haar	$d[i] = d_0[i] - s_0[n]$ $s[i] = s_0[i] + \lfloor \frac{1}{2} d[i] \rfloor$
5/3	$d[i] = d_0[i] - \lfloor \frac{1}{2} (s_0[i+1] + s_0[i]) \rfloor$ $s[i] = s_0[i] + \lfloor \frac{1}{4} (d[i] + d[i-1]) + \frac{1}{2}$
9/7-m	$d[i] = d_0[i] + \lfloor \frac{1}{16} ((s_0[i+2] + s_0[i-1]) - 9 (s_0[i+1] + s_0[i])) + \frac{1}{2} \rfloor$ $s[i] = s_0[i] + \left[\frac{1}{4} (d[i] + d[i-1]) + \frac{1}{2} \right\rfloor$
2/6	$d_1[i] = d_0[i] - s_0[i]$ $s[i] = s_0[i] + \lfloor \frac{1}{2} d_1[i] \rfloor$ $d[i] = d_1[i] + \lfloor \frac{1}{4} (-s[i+1] + s[i-1]) + \frac{1}{2} \rfloor$
2/10	$d_1[i] = d_0[i] - s_0[i]$ $s[i] = s_0[i] + \lfloor \frac{1}{2} d_1[n] \rfloor$ $d[i] = d_1[i] + \lfloor \frac{1}{64} (22(s[i-1] - s[i+1]) + 3(s[i+2] - s[i-2])) + \frac{1}{2} \rfloor$

The forward transformation makes sure that the approximation and detail coefficients are integers so that the losses in the rounding minimizes. Haar filter and 5/3 filter are the fastest and the least computationally complex compared to the 2/6 and 2/10 wavelet filters. This arises due to the extra forward transform that should be taken to decompose the coefficients. The inverse transforms can be deduced using the same forward transform formulas in reverse order.

2.2 Discrete Wavelet Transform

Images can be decomposed into several levels. In our case, since the coefficients compressed using graph Fourier transform are saved using 16 bits. For this reason, we need to decompose more than three levels. For this reason, we will be experimenting using four-level and five-level discrete wavelet transforms.

2.2.1 Four-level discrete wavelet transform

Our method consists of compressing most of the image and using Graph Fourier Transform only on a few bands of the image. We start by doing a four-level Discrete Wavelet Transform decomposition and using Graph Fourier Transform on the LLLL bands. Using Graph Fourier Transform has some drawbacks since every pixel is saved either with 16 bits or 12 bits.

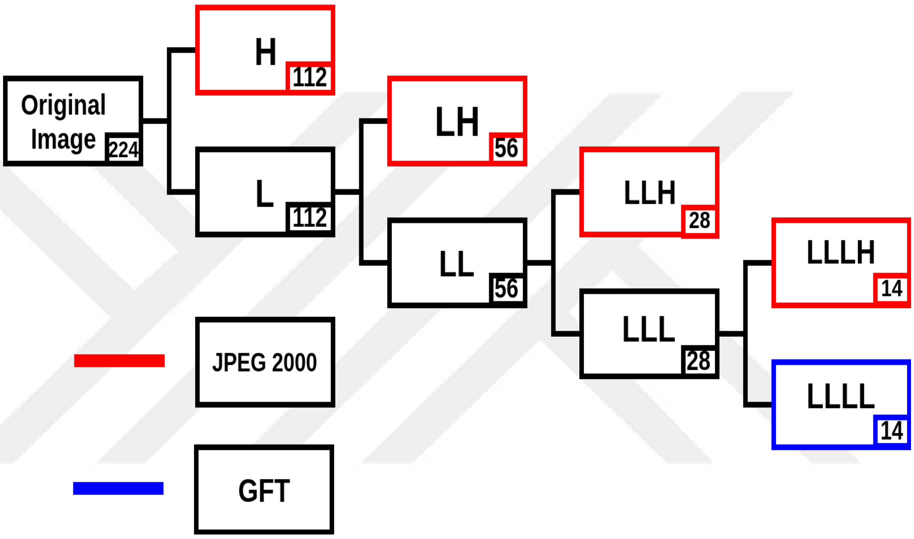


Figure 2.2 : A four-level Discrete Wavelet Transform of AVIRIS hyperspectral images. Boxes in red show that the images are compressed using JPEG-2000 compression scheme and the last fourteen bands are reproduced using Graph Fourier Transform.

This increases the memory allocation needed to save the desired coefficients of the Graph Fourier Transform. Using GFT on the 14 will take a lot of memory so the coding scheme will be inefficient and will lose its purpose.

The results of doing a four-level Discrete Wavelet Transform will be discussed in the next chapter. We deduce that the four-level Discrete Wavelet Transform will not yield good results since pixels of the last 14 bands will be saved with 16 or 12 bits.

Therefore, we need to do one more Discrete Wavelet Transform decomposition so we can use only 7 out of 14 bands. By doing this, we will decorrelate the hyperspectral image, even more, this will improve the compression result even higher.

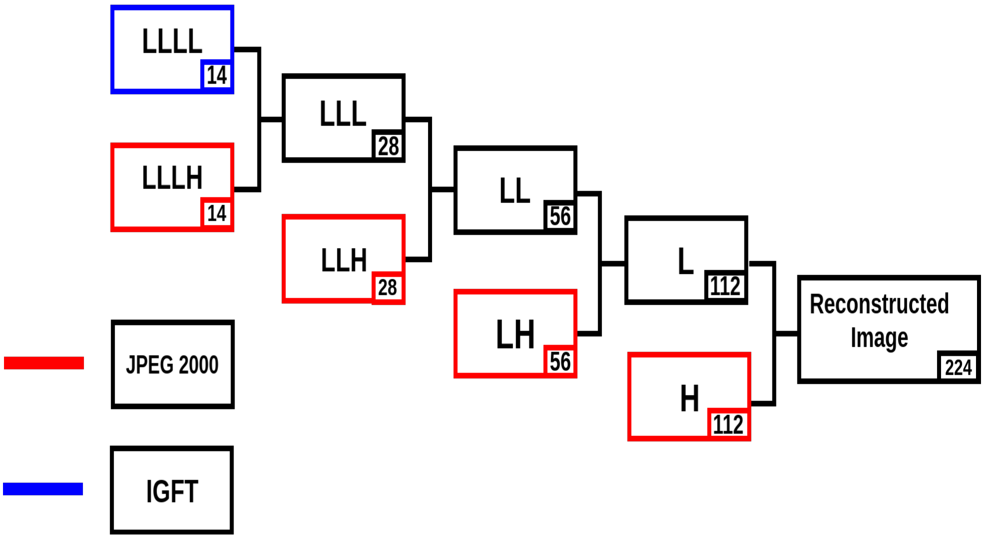


Figure 2.3 : A four-level Discrete Wavelet Transform of AVIRIS hyperspectral images. Boxes in red show that the images are compressed using JPEG-2000 compression scheme and the last fourteen bands are reproduced using Graph Fourier Transform.

2.2.2 Five-level discrete wavelet transform

Having decided that the five-level Discrete Wavelet Transform will produce us the best result we will apply our method, Graph Fourier Transform, on the last bands of the whole DWT process. Firstly we will use GFT on LLLLL bands and then we will use GFT on LLLLH bands. They have the same number of bands so they occupy the same memory size. below we have shown the process for both methods and in the results section, we will obtain and discuss obtained from the hyperspectral images.

As we can see from Figure 2.4, we firstly decorrelate the hyperspectral image using five-level Discrete Wavelet Transform and decompose the image five times thus creating six cubes from the original image. H, LH, LLH, LLLH, LLLLL cubes are compressed in lossy mode with low and high compression ratios. We apply Graph Fourier transform on the detail coefficients or LLLLH bands. We compress the other bands using JPEG-2000 in lossy mode with different ratios.

The second method is very similar to the previous one. As we can see from Figure 2.5, we apply a five-level Discrete Wavelet Transform and decompose the image five times thus creating six cubes from the original image. H, LH, LLH, LLLH, LLLLH cubes are compressed in lossy mode with low and high compression ratios. We apply Graph Fourier Transform on the approximation coefficients or LLLLL bands. We compress the other bands using JPEG-2000 in lossy mode with different ratios.

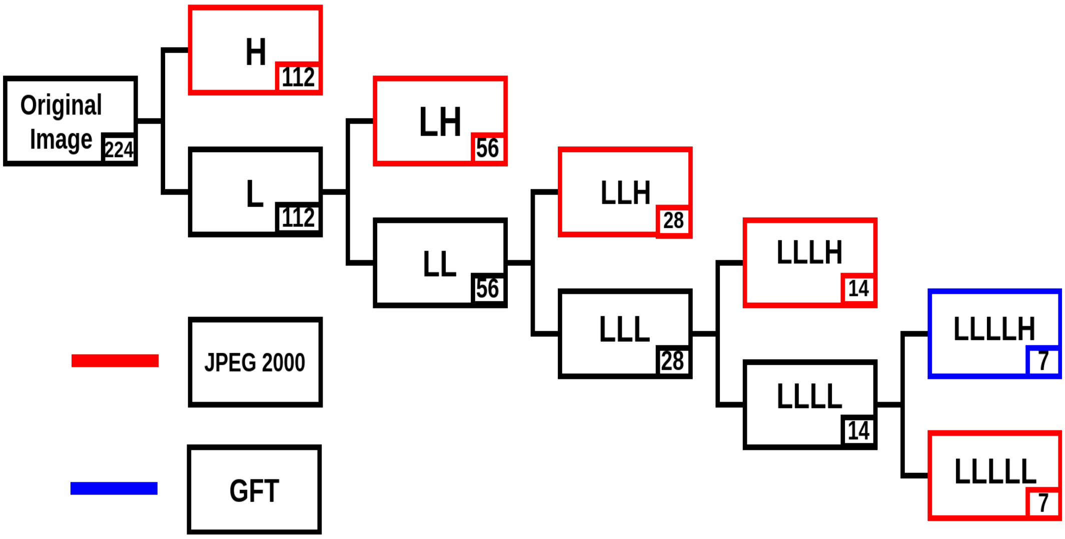


Figure 2.4 : A five-level Discrete Wavelet Transform decomposes the image five times thus creating six cubes from the original image. The bands from H, LH, LLH, LLLH, LLLLL cubes are compressed in lossy mode with low and high compression ratios and Graph Fourier Transform is applied to LLLLH bands Boxes in red show that the images are compressed using JPEG-2000 compression scheme and the last seven bands are reproduced using Graph Fourier Transform.

To achieve a BPS of 0.1 we compress the cubes 160 times their original size. In our case, the original cubes have a size of 28 MB compressing the image 160 reduces the HSI size to just 0.175 MB.

2.3 Graph Fourier Transform

The novelty of our coding scheme is using Graph Fourier Transform [20]. It is a new technique of treating graphs as signals and nodes as signal values. We apply graph signal processing tools to regular and irregular graphs. Signal processing on the graph has a lot of applications in the fields of social networks [24], World Wide Web [25], sensor malfunction [26], etc.

In our case, we treat each band as of the hyperspectral image as a graph where nodes are the pixels and each node value is equal to the pixel intensity. When converted to a graph, our image has a regular 2D shape. In order to continue, we must convert our graph into a matrix.

Given a graph G={V, E}, where *V* is the set of nodes and *E* is the set of edges, we can formulate the Graph Laplacian as the matrix so mathematical operations can be

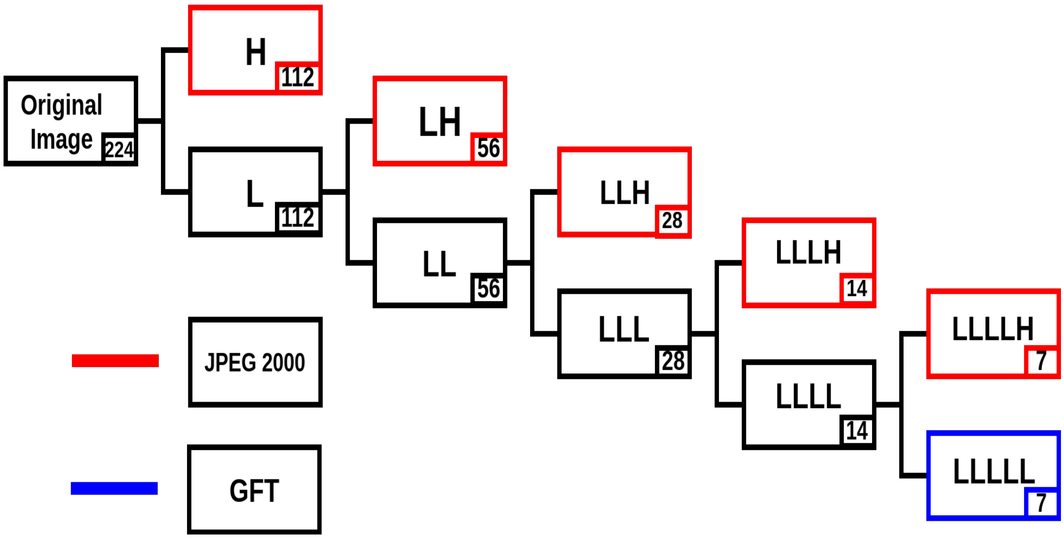


Figure 2.5 : A five-level Discrete Wavelet Transform decompose the image five times thus creating six cubes from the original image. H, LH, LLH, LLLH, LLLLH. We apply Graph Fourier Transform onto the LLLLL bands Boxes in red show that the images are compressed using JPEG-2000 compression scheme and the last seven bands are reproduced using Graph Fourier Transform.

applied. To formulate the Graph Laplacian we must find the adjacency matrix and the degree matrix.

Adjacency matrix, **A**, is the matrix representation of edge weights that occur among the nodes. The simplest weight is expressed as 1 or 0 where 1 means there is a connection between the nodes and 0 means there is no connection between the nodes. A graph with N nodes will give us an adjacency matrix of N × N shape. There are several metrics to find the edge weight between two nodes and it depends on the application. In our case, the weight matrix is expressed as below [27]:

$$w_{i,j} = \exp\left(-\frac{||I_i - I_j||_2^2}{\sigma^2}\right) \tag{2.1}$$

where i and j are the two nodes, I_i and I_j are the intensity of the pixels i and j and σ is a parameter. There are several other metrics to calculate the similarity between pixels and the readers can read the paper above for more information.

We continue by defining the Degree matrix, **D**, as an N × N matrix whose entries describe the number of connections each node has. It is a sparse matrix whose non-diagonal entries are zero.

Laplacian matrix is the translation of a graph from a geometric shape to a matrix formulation. It is defined as the degree matrix minus the adjacency matrix and contains every information related to the graph. Having defined **D** and **A** we compute the Graph Laplacian matrix as follows:

$$\mathbf{L} = \mathbf{D} - \mathbf{A} \tag{2.2}$$

where each element is defined as :

$$L_{ij} = \begin{cases} \deg(i), & \text{if } i = j \\ -w_{ij}, & \text{if } (i,j) \in E \\ 0, & \text{otherwise} \end{cases} \tag{2.3}$$

In the end, for a graph with N nodes, we end up with a Graph Laplacian matrix of shape N × N where each node and its edges are numerically defined for processing.

Below, in Figure 2.6, we have shown a graph representation of a 7 × 7 image. We can see that the adjacency matrix has a shape of 49 × 49 from the weight of the last node. Graph signal Processing helps us capture valuable information that is recorded in the edges.

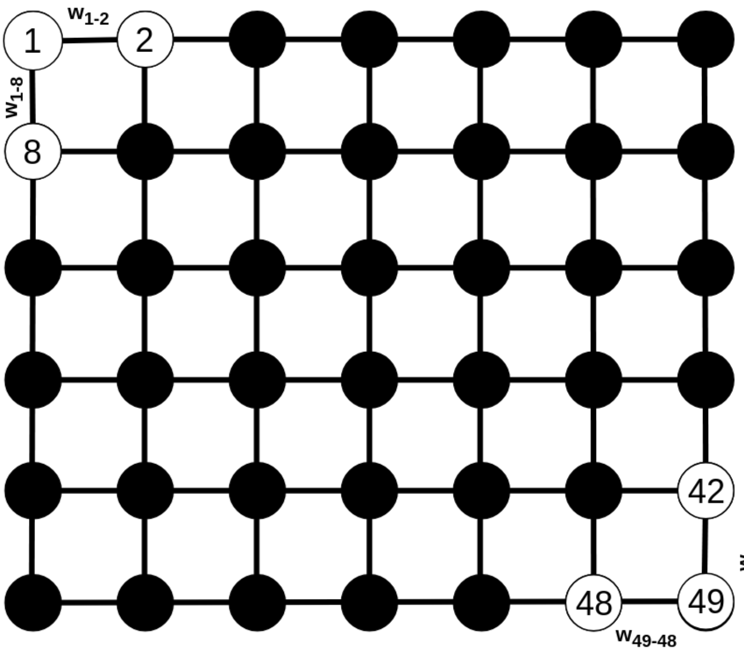


Figure 2.6 : A graph representation of a 7 image.

As can be seen, the first node is connected with the second node and the eighth node and their edge weight is calculated according to the above weight equation whereas the last node is connected with node 42 and node 48.

To compress the images we use Graph Fourier Transform. We transform the signal in the graph using GFT and use only save a few coefficients to reconstruct the signal back. We start by Eigen-decomposing the Laplacian matrix as such :

$$L = X\Lambda X^{-1} \tag{2.4}$$

where Λ is the set of eigenvalues, X is the set of orthonormal eigenvectors. We define graph Fourier transform as X^{-1} and we can transform an image as shown below:

$$\widehat{x} = \mathrm{X}^{-1}x \tag{2.5}$$

where x is the transformed signal. we can reconstruct the original signal by taking the inverse graph Fourier transform as shown in the equation below:

$$x = \mathrm{X}\widehat{x} \tag{2.6}$$

Graph Fourier transform has proven to be very effective in reconstructing signals on graphs. It is mostly used in image compression by picking a few coefficients to reconstruct the original signal.

2.4 Hyperspectral Image Reconstruction

After selecting only a few coefficients from the last seven bands we can reconstruct the hyperspectral image by taking the Inverse Discrete Wavelet Transform. After decompressing the spectral bands we join the cube back and measure the image quality between the original and the reconstructed hyperspectral images. From figure 2.4 we can see the process of reconstructing the hyperspectral image. The bands are up-sampled by a factor of two. In the end, we will have a reconstructed hyperspectral image with the same shape.

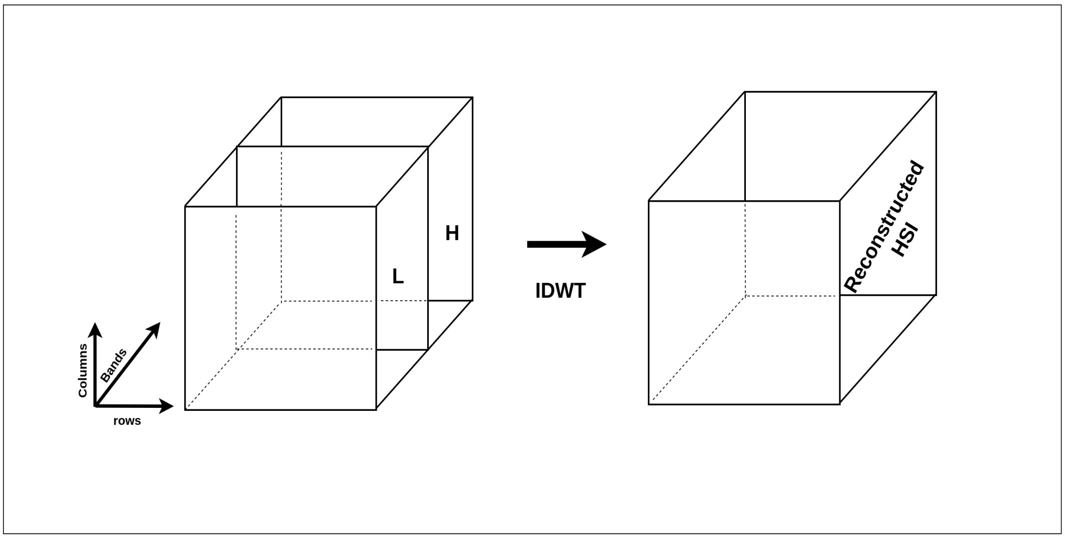


Figure 2.7 : Hyperspectral Image reconstruction using using inverse discrete wavelet transform.

In the end, the reconstructed and original hyperspectral cube must be of equal shape. Next chapter, we will be discussing the datasets used and the results obtained by our novel compression scheme. A comparison with JPEG 2000 compression standard with being shown and the results will be compared with other compressing methods. Below figures show the reconstruction of a hyperspectral image using a five-level Inverse Discrete Wavelet Transform.

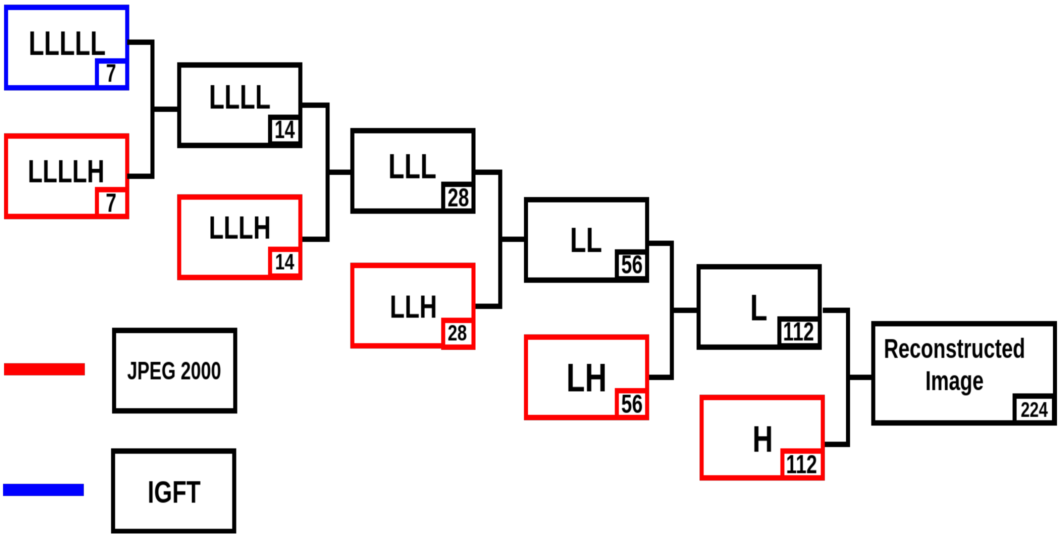


Figure 2.8 : A five-level Inverse Discrete Wavelet Transform of the hyperspectral images.

In figure 2.8, we have used Inverse Graph Fourier Transform to compress the LLLLL bands by saving only a small number of coefficients and reducing the size needed to record all the 7 bands. We decompress each band from the other LLLLH, LLLH, LLH, LH, and H cubes and use Inverse Discrete Wavelet Transform to reconstruct the image.

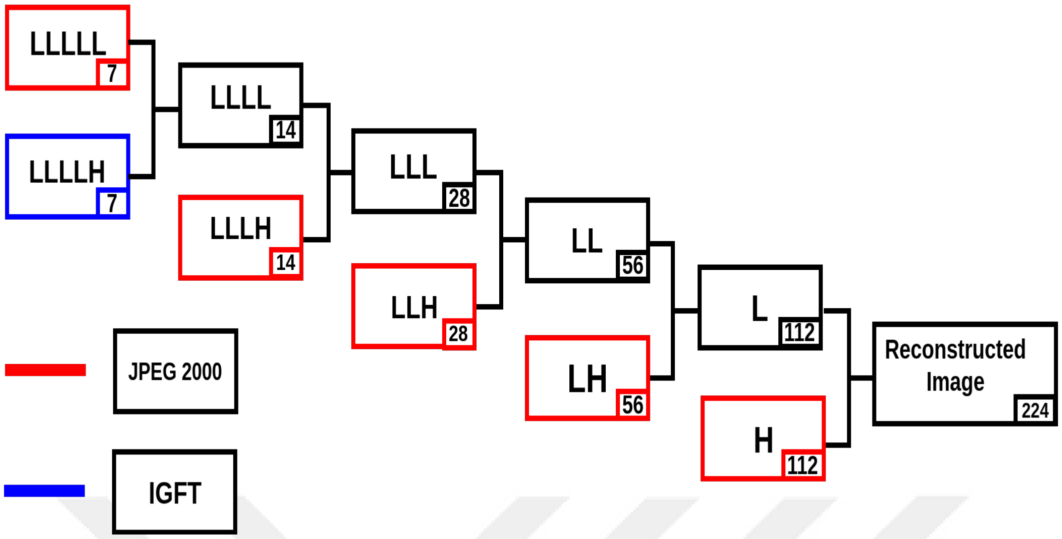


Figure 2.9 : A five-level Inverse Discrete Wavelet Transform of the hyperspectral images.

2.5 JPEG-2000 Compression Scheme

We compress most of the hyperspectral images using JPEG-2000. It is created by the Joint Photographic Experts Group (JPEG) in 2000 and it is one of the most used schemes for lossy and lossless compression. JPEG-2000 offers better image quality preservation compared to JPEG. There are several methods to measure the image quality degradation of the reconstructed image. JPEG-2000 can compress grayscale and multi-band images in lossy and lossless mode.

The most widely used quality metrics are Peak Signal to Noise Ratio (PSNR) and Mean Square Error (MSE). As stated by Hore *et al*, there are similarities and differences between these quality metrics [28]. Michael W. Marcellin *et al*, in their paper, points out that there are several dominant features of JPEG-2000 such as low bit-rate compression performance, progression transmitting by component, quality, and resolution, implementation with limited memory, etc [29].

2.6 Peak Signal to Noise Ratio (PSNR)

There are several metrics used to measure the quality of the images. Most of them are categorized as subjective and objective methods. Subjective based metrics are based on human judgment whereas objective methods are based with reference to the original image. In our research, we will continue with objective based metrics as we want to compare the original image and the reconstructed image.

MSE metrics measures the average of the squared difference between the estimated pixel and its original pixel. Given an image X and the reconstructed image Y, their MSE value is:

$$\mathrm{MSE} = \frac{1}{\mathrm{MNL}} \sum_{i=1}^{M} \sum_{j=1}^{N} \sum_{i=1}^{L} \left(X(i,j,k) - Y(i,j,k)\right)^2 \tag{2.7}$$

where M, N, and L are the pixel's row, column, and band location. It is always positive and a small value correspond to a high quality for the reconstructed image.

The PSNR is one of the most useful metrics that is used to calculate the image quality. Given an image X and the reconstructed image Y, their PSNR value is:

$$PSNR(dB) = 20 \cdot \log_{10} \left(\frac{2^B - 1}{\sqrt{MSE}} \right) \tag{2.8}$$

where B is the number of bits a pixel is stored. In our case, B has values of 12 and 16. A high PSNR value means there is a low distortion between the reconstructed and original hyperspectral image.

In our research, we use the PSNR since is the most commonly used metrics for lossy compression coding schemes.

3. DATASETS AND EXPERIMENTAL RESULTS

When applying Graph Fourier Transform into hyperspectral images of 256 × 256 we saw that it takes the computer more that one day to eigendecompose the image and do computation on it. For this reason, in order to reduce the computational time and complexity, we split the 256 × 256 hyperspectral image into 16 × 16 blocks and apply Graph Fourier Transforms to each block. In the end, the reconstructed hyperspectral is formed by joining the smaller blocks.

There are two methods to do this. In the first method, the Graph Fourier Transform is applied to the last approximation sub-bands after the five-level Discrete Wavelet Transform is applied to the hyperspectral image. We apply the Graph Fourier Transform onto the 16 × 16 blocks and reconstruct the approximation sub-bands back by joining all the small blocks. We finish by applying Inverse Discrete Wavelet Transform to reconstruct the hyperspectral image.

The second method, we split the original image into 16 × 16 blocks and apply Discrete Wavelet Transform, Graph Fourier Transform and Inverse Discrete Wavelet Transform onto each block and calculate the PSNR value for each block. In the end, we calculate the PSNR value for the whole image by averaging the smaller blocks of PSNR values.

In this thesis, we have used publicly available Hyperspectral images obtained by AVIRIS which have 224 bands. We have also used images obtained by Hyperion imaging system. Hyperion is a hyperspectral sensor able to capture 242 bands [30]. At last, we have used Pavia University hyperspectral image captured by ROSIS imaging system. Pavia University HSI contains 103 bands with each pixel written in 16 bits [31]. For our experiments, we only compare a small portion of the whole cube to reduce computational time.

Our images have a size of $256 \times 256 \times 224$ when we use AVIRIS hyperspectral images, $256 \times 256 \times 128$ when we use Botswana hyperspectral image and $256 \times 256 \times 96$ when we use Pavia University HSI in order to decompose the images five times. We can compare them with th other state-of-the-art methods. We compare the coding scheme with other methods such as: TFEMPR [14], LASSO [32] , CPPCA [33], SIP [34], gOMP [35], BCS PL-2DBS + 2D DDWT [36], BCS SPL-2DBS + 2D DDWT [37] .

Table 3.1 : Hyperspectral Image Datasets. Images are captured with different imaging systems and they have different band number. For our experiments, we will use only a portion of the images, more specifically, 256×256.

Name	Sample No	Lines No	Bands No	Bit-depth
Lunar Lake	614	3686	224	16
Jasper Ridge	614	2587	224	16
Low Altitude	614	1432	224	16
Botswana	256	1476	242	16
Pavia University	340	610	103	16

3.1 Jasper Ridge HSI

Before comparing our coding scheme with the other schemes, we have obtained PSNR values for different BPS values for Jasper Ridge Hyperspectral Image. This image is captured using AVIRIS hyperspectral imaging sensor and it captures 224 bands of the same area.

In Figure 3.1 we have shown the Jasper Ridgehyperspectral images, whose each band captures a different wavelength reflected from the area. As seen band number 10 shows a river which cannot be seen in the other bands. Bands 50 and 100 have minor differences between them. we can see some green regions in band 100 that can be associated with green fields.

Firstly we will start comparing the two methods mentioned at the beginning of this chapter. The first method consists of splitting the cube into 16×16 mini cubes and applying our method onto each mini cube and calculating the PSNR value by taking the average of all the mini cubes. The second method, called the whole cube method, consists in spectrally decorrelating the hyperspectral image five times and applying the graph Fourier transform only on the last bands of the decorrelated cube.

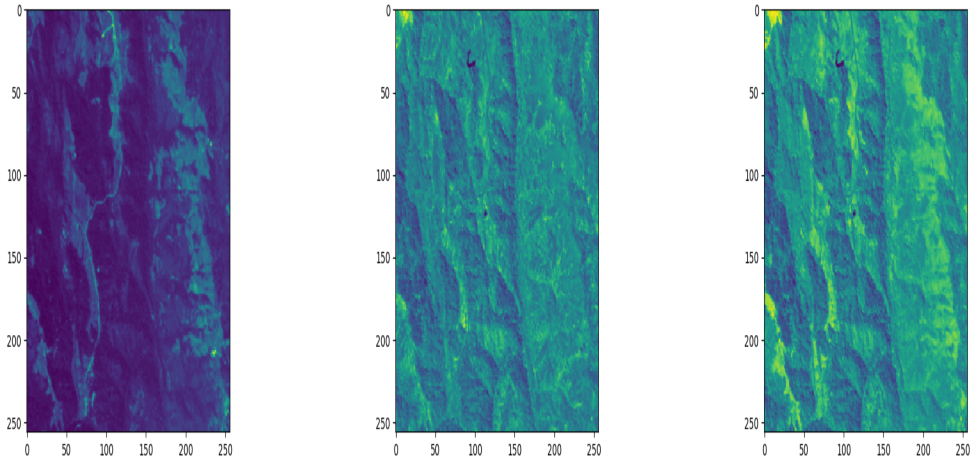


Figure 3.1 : Jasper Ridge band 10, 50, 100.

From Table 3.2 we see that the 16 × 16 blocks method gives better results than using GFT only on the last decorrelated bands. bands. There is no major difference between the filters used. In both methods, the Haar filter produces the lowest PSNR value whereas 9/7-m and 2/10 filters produce the best PSNR value. There is no significant difference among the PSNR values since the bitrate value is 0.1 and the loss in image quality is very high.

Table 3.2 : Comparison of the two methods for BPS = 0.1 applied to the Jasper Ridge Hyperspecral Image.

Methods Filter	Whole Image	16 × 16 Blocks
	PSNR (dB)	PSNR (dB)
Haar	58.54	58.72
5/3	58.93	59.10
2/6	58.92	59.11
CDF 9/7-m	53.93	59.14
2/10	58.96	59.14

From Table 3.3, we can see that 16 × 16 coding scheme performs better than the whole image coding scheme for a BPS of 0.3. Compared with the other filters 2/6 filter produces the highest PSNR value together with 9/7-m. Although, we must say that the difference is very small.

From Table 3.4, we can see that 16 × 16 coding scheme performs better than the whole image coding scheme for a BPS of 0.5. Compared with the other filters 2/6 filter produces the highest PSNR value. The difference is very small.

Table 3.3 : Comparison of the two methods for BPS = 0.3 applied to the Jasper Ridge Hyperspecral Image.

Filter \ Methods	Whole Image	16 × 16 Blocks
	PSNR (dB)	PSNR (dB)
Haar	62.53	62.64
5/3	63.1	63.21
2/6	63.45	63.61
CDF 9/7-m	63.16	63.27
2/10	63.56	63.72

Table 3.4 : Comparison of the two methods for BPS = 0.5 applied to the Jasper Ridge Hyperspectral Image.

Filter \ Methods	Whole Image	16 × 16 Blocks
	PSNR (dB)	PSNR (dB)
Haar	64.16	64.27
5/3	64.72	64.81
2/6	65.49	65.62
CDF 9/7-m	64.92	65.03
2/10	65.63	65.77

We have shown that 16 $\times$ 16 blocks coding scheme gives better PSNR values for all BPS values for all filters. In Jasper Ridge hyperspectral image, 2/10 filter outputs the best result in most of the cases, although, the difference is not very large. We will move on the second part of our coding scheme.

Since the last output of a five-level discrete wavelet transform are two small cubes with the same shape, we want to use Graph Fourier Transform on both of them and compute which one of the cubes, LLLLL or LLLLH, outputs the highest PSNR. When we use graph Fourier transform on the LLLLL bands we will be compressing the approximation coefficients and when we use graph Fourier transform on the LLLLH we will be compressing the detail coefficients.

As we can see from Table 3.5, for a BPS of 0.1 using Graph Fourier Transform on LLLLL bands gives better PSNR values. When we use LLLLH bands, we see similar results with small change for all the filters. When we compress LLLLL bands, we conclude that the 2/10 filter gives the best PSNR value of 59.14 dB for a bitrate equal to 0.1.

Table 3.5 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.1

Methods Filter	LLLLH	LLLLL
	PSNR (dB)	PSNR (dB)
Haar	56.23	58.72
5/3	56.35	59.10
2/6	56.22	59.11
CDF 9/7-m	56.33	59.10
2/10	56.19	59.14

Table 3.6 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.3

Methods Filter	LLLLH	LLLLL
	PSNR (dB)	PSNR (dB)
Haar	58.00	62.64
5/3	58.54	63.21
2/6	57.97	63.61
CDF 9/7-m	58.42	63.27
2/10	57.93	63.72

In Table 3.6, for a BPS of 0.3 using Graph Fourier Transform on LLLLL bands gives better PSNR values. We see that the filter 2/6 gives the best PSNR value of 63.72 dB. It is important to mention that there is not a huge difference between the results between different filters but there is a significant difference if we apply Graph Fourier Transform between sub-bands. LLLLH bands contain detail coefficients whereas LLLLL contain approximation coefficients.

In Table 3.7, for a BPS of 0.5 using Graph Fourier Transform on LLLLL bands gives better PSNR values. We see that the filter 9/7-m gives the best PSNR value of 63.72 dB for a BPS equal to 0.5. It is important to mention that there is not a huge difference between the results from different filters. They produce similar results with a small difference among them.

At Table 3.8, we have presented the highest values of each filter produced for each bitrate value. We see that by using the Haar filter, our PSNR value increases from 58.72 dB to 64.27 dB. Filter 2/10 produces the best results where

When we compare the filters we see that 2/10 produces the highest PSNR values for each BPS.

Table 3.7 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.5

Methods Filter	LLLLH	LLLLL
	PSNR (dB)	PSNR (dB)
Haar	56.23	64.27
5/3	64.72	64.81
2/6	65.49	65.62
CDF 9/7-m	64.92	65.03
2/10	65.63	65.77

Table 3.8 : PSNR (dB) vs BPS for Jasper Ridge Hyperspectral Image.

BPS Filter	0.1	0.3	0.5
	PSNR (dB)	PSNR (dB)	PSNR (dB)
Haar	58.72	62.65	64.27
5/3	59.10	63.21	64.81
2/6	59.11	63.61	65.62
CDF 9/7-m	59.10	63.72	65.03
2/10	**59.15**	**63.60**	**65.77**

In Figure 3.2, we have plotted the PSNR vs BPS for all filters of the Jasper Ridge hyperspectral image. As we previously stated, 2/10 integer based filter produces a better result than the Haar filter. Also, we see that 2/10 filter output approximately the same result as 2/10 filter.

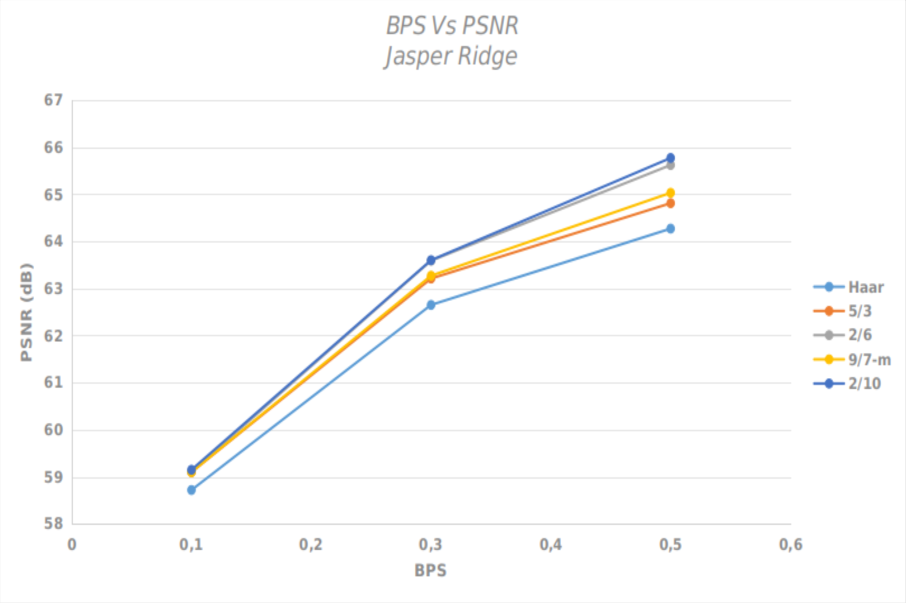


Figure 3.2 : A comparion of different filter for Jasper Ridge hyperspectral image.

To conclude with Jasper Ridge, we have compressed the image using different filters and compared the reconstructed hyperspectral image with the original hyperspectral images by their PSNR values. We can see from Table 3.9 that the PSNR values increase when BPS increases. The 9/7-m filter, which is commonly used for lossy compression, gives almost a good result as 2/10 and 2/6 filters but the computation time is higher. Compression of the hyperspectral image using 2/10 filter gives the highest PSNR value for all bit rate values.

Table 3.9 : Comparison of our compression scheme with other methods for Jasper Ridge HSI.

BPS Filter	0.1	0.3	0.5
	PSNR (dB)	PSNR (dB)	PSNR (dB)
GFT + DWT	59.15	63.6	65.77
SIP	58.06	66.74	71.34
CPPCA	30.20	71.31	76.40
LASSO	59.30	70.60	73.71
TFEMPR	70.99	80.50	83.51
gOMP	59.40	70.01	71.14
BCS PL-2DBS + 2D DDWT	50.60	54.18	57.11
BCS SPL-2DBS + 2D DDWT	50.30	53.67	56.45

Compared with other methods, our method produces better results than block compressed sensing methods, is on par with methods such as SIP and gOMP but lags behind methods such as LASSO and TFEMPR. We believe that our method of compressing the images with JPEG 2000 in lossless mode reduces the quality of the images when we go to BPS values of 0.1, 0.3 and 0.5.

3.2 Low Altitude HSI

In Figure 3.3 we have shown the Low Altitude hyperspectral image bands where each band captures a different wavelength reflected from the area. We can see that band number 10 shows segmented parcels of fields whereas band 50 and 100 shows a small lake in the upper part of the image.

After determining that compressing the images in 16 $\times$ 16 blocks and using Graph Fourier Transform on LLLLL band outputs the highest PSNR values. For this reason, for the rest of the images, we will be compressing out images using this compression scheme.

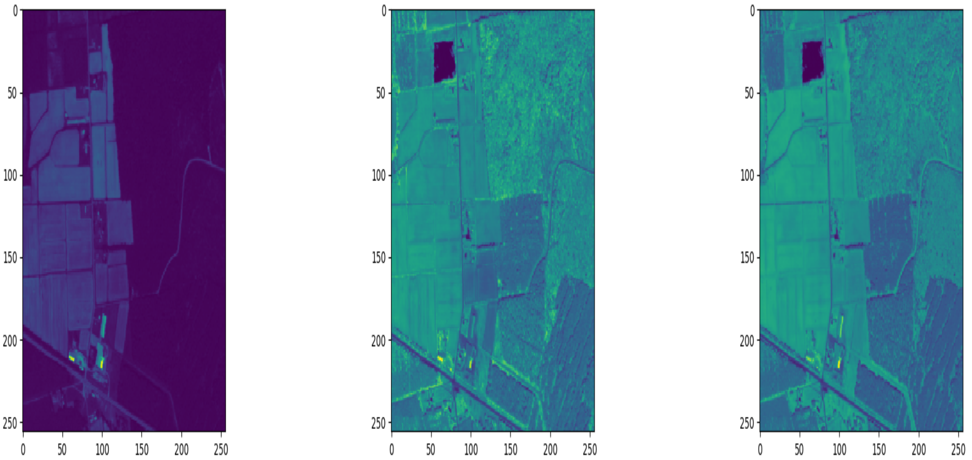


Figure 3.3 : Low Altitude band 10, 50, 100.

We continue compressing the Low Altitude hyperspectral image for different BPS values. We will decorrelate the spectral bands using five integer based filters such as Haar, 2/6, 2/10, 5/3, 9/7-m. We will follow the same path and we will be comparing between LLLLL and LLLLH bands.

Table 3.10 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.1

Methods Filter	LLLLH	LLLLL
	PSNR (dB)	PSNR (dB)
Haar	54.89	57.04
5/3	55.14	57.32
2/6	55.91	57.45
CDF 9/7-m	54.19	57.27
2/10	54.88	57.51

As we can see from Table 3.10, for a BPS of 0.1 using Graph Fourier Transform on LLLLL bands gives better PSNR values. When we use LLLLH bands, we see similar results with small change for all the filters. When we use LLLLL bands, we conclude that the filter 2/10 gives the best PSNR value of 57.51 dB for a BPS equal to 0.1.

In Table 3.11, for a BPS of 0.3, using Graph Fourier Transform on LLLLL bands gives better PSNR values. We see that the filter 2/6 gives the best PSNR value of 61.75 dB for a BPS equal to 0.3. It is important to mention that there is not a huge

Table 3.11 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.3

Methods Filter	LLLLH	LLLLL
	PSNR (dB)	PSNR (dB)
Haar	56.81	60.89
5/3	57.47	61.15
2/6	56.81	61.63
CDF 9/7-m	57.40	61.09
2/10	59.56	61.75

difference between different filters but there is a significant difference if we apply Graph Fourier Transform between LLLLL or LLLLH. LLLLH bands contain detail coefficients whereas LLLLL contain approximation coefficients.

In Table 3.12, for a BPS of 0.5 using Graph Fourier Transform on LLLLL bands gives better PSNR values. We see that the filter 9/7-m gives the best PSNR value of 63.50 dB for a BPS equal to 0.5.

Table 3.12 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.5

Methods Filter	LLLLH	LLLLL
	PSNR (dB)	PSNR (dB)
Haar	59.58	62.24
5/3	60.24	62.44
2/6	59.62	63.35
CDF 9/7-m	60.03	62.49
2/10	59.56	63.50

It is important to mention that there is not a huge difference between the results from different filters. They produce similar results with a small difference among them

Table 3.13 : PSNR Vs BPS for Low Altitude Hyperspecral Image.

BPS Filter	0.1	0.3	0.5
Haar	57.04 dB	60.89 dB	62.24 dB
5/3	57.32 dB	61.15 dB	62.44 dB
2/6	57.45 dB	61.63 dB	63.35 dB
CDF 9/7-m	57.27 dB	61.09 dB	62.49 dB
2/10	**57.51 dB**	**61.75 dB**	**63.50 dB**

From Table 3.13, we compressed the image using different filters and compared the reconstructed hyperspectral image with the original hyperspectral images by their PSNR values. As bit rate increases we see an increase in PSNR values. The 9/7-m filter, which is commonly used for lossy compressions, gives almost a good result as filer 2/10 and 2/6. Their higher result also results in higher computation time.

Compression of the hyperspectral image using 2/10 filter gives the highest PSNR value for all bit rate values. We use these values when comparing our coding scheme with other methods.

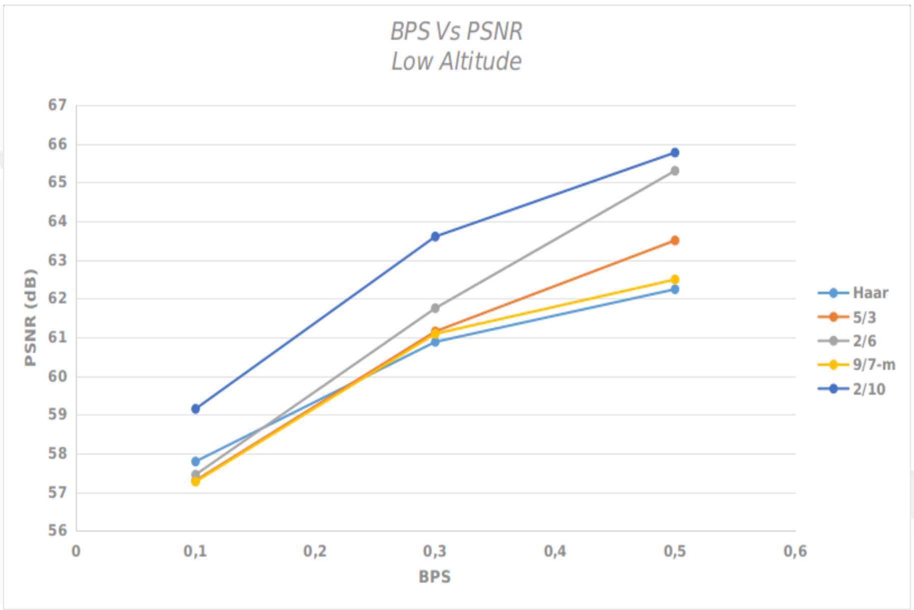


Figure 3.4 : A comparion of different filter for Low Altitude hyperspectral image.

Compared with other methods, our method is on par with methods such as SIP and gOMP but lags behind methods such as LASSO and TFEMPR. We believe that our method of compressing the images with JPEG 2000 in lossless mode reduces the quality of the images when we go to very small values of BPS such as 0.1, 0.3 or 0.5.

3.3 Lunar Lake HSI

We continue compressing the Lunar Lake hyperspectral image for different BPS values. We will decorrelate the spectral bands using five integer based filters such

Table 3.14 : Comparison of our compression scheme with other methods for Low Altitude HSI.

Filter \ BPS	0.1	0.3	0.5
	PSNR (dB)	PSNR (dB)	PSNR (dB)
GFT + DWT	57.51	61.75	63.50
SIP	58.06	66.74	71.34
CPPCA	30.20	71.31	76.40
LASSO	59.30	70.60	73.71
TFEMPR	70.99	80.50	83.51
gOMP	59.40	70.01	71.14
BCS PL-2DBS + 2D DDWT	47.97	51.67	54.45
BCS SPL-2DBS + 2D DDWT	48.02	51.46	54.40

as Haar, 2/6, 2/10, 5/3, 9/7-m. We will follow the same path we followed with the other images and we will compare LLLLL and LLLLH bands.

In Figure 3.5 we have shown the Lunar Lake hyperspectral image. Here we have shown only three bands: 10, 50 and 100.

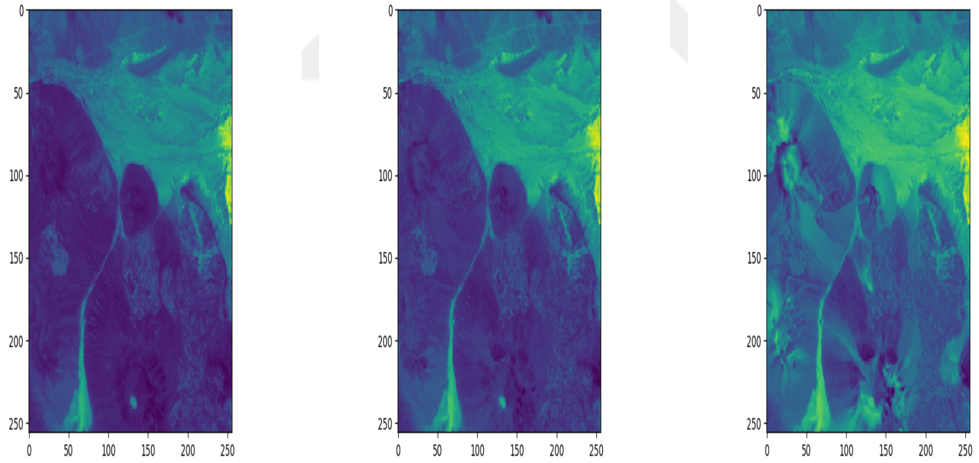


Figure 3.5 : Lunar Lake band 10, 50, 100.

As we can see from Table 3.14, for a BPS of 0.1 using Graph Fourier Transform on LLLLL bands gives better PSNR values. When we use LLLLH bands, we see similar results with small change for all the filters. When we use LLLLL bands, we conclude that the filter 5/3 gives the best PSNR value of 62.11 dB for a BPS equal to 0.1.

In Table 3.15, for a BPS of 0.3, using Graph Fourier Transform on LLLLL bands gives better PSNR values. We see that the filter 5/3 gives the best PSNR value of

Table 3.15 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.1

Methods Filter	LLLLH	LLLLL
	PSNR (dB)	PSNR (dB)
Haar	57.74	61.08
5/3	57.80	62.11
2/6	57.74	61.62
CDF 9/7-m	56.76	62.03
2/10	57.72	61.72

68.39 dB for a BPS equal to 0.3. It is important to mention that there is not a huge difference between different filters but there is a significant difference if we apply Graph Fourier Transform between LLLLL or LLLLH. LLLLH bands contain detail coefficients whereas LLLLL contain approximation coefficients.

Table 3.16 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.3

Methods Filter	LLLLH	LLLLL
	PSNR (dB)	PSNR (dB)
Haar	60.05	66.00
5/3	60.12	68.39
2/6	60.03	67.90
CDF 9/7-m	60.08	68.27
2/10	59.99	67.33

In Table 3.17, for a BPS of 0.5 using Graph Fourier Transform on LLLLL bands gives better PSNR values. We see that the filter 5/3 gives the best PSNR value of 71.17 dB for a BPS equal to 0.5. It is important to mention that there is not a huge difference between the results from different filters. They produce similar results with a small difference among them.

From Table 3.18, we compressed the image using different filters and compared the reconstructed hyperspectral image with the original hyperspectral images by their PSNR values. As bit rate increases we see an increase in PSNR values. The 9/7-m filter, which is commonly used for lossy compressions, gives almost a good result as filer 2/10 and 2/6. Their higher result also results in higher computation time. Compression of the hyperspectral image using a 5/3 filter gives the highest PSNR

Table 3.17 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.5

Methods / Filter	LLLLH	LLLLL
	PSNR (dB)	PSNR (dB)
Haar	63.05	67.78
5/3	63.26	71.17
2/6	63.03	68.77
CDF 9/7-m	63.14	70.96
2/10	62.99	69.82

Table 3.18 : PSNR (dB) Vs BPS for Lunar Lake Hyperspecral Image.

BPS / Filter	0.1	0.3	0.5
	PSNR (dB)	PSNR (dB)	PSNR (dB)
Haar	61.08	66.00	67.78
5/3	**62.11**	**68.39**	**71.17**
2/6	61.62	67.90	68.77
CDF 9/7-m	62.03	68.27	70.96
2/10	61.72	67.33	69.82

value for all bit rate values. We use these values when comparing our coding scheme with other methods.

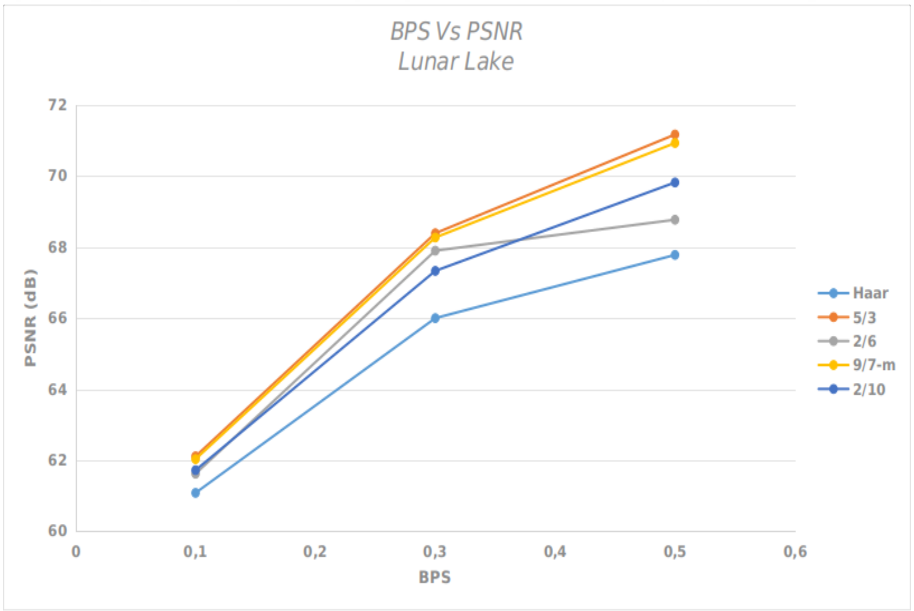


Figure 3.6 : A comparison of different filter for Lunar Lake hyperspectral image. We see that 5/3 outputs the best PSNR value compared to the other filters.

We can see from figure 3.6 that 5/3 filter output the best result among the other filters. Haar filter, which has the simplest transform, outputs the poorest results for all BPS values.

Table 3.19 : Comparison of our compression scheme with other methods for Lunar Lake HSI.

Filter \ BPS	0.1	0.3	0.5
	PSNR (dB)	PSNR (dB)	PSNR (dB)
GFT + DWT	62.11	68.39	71.17
SIP	58.86	70.71	72.39
CPPCA	48.43	72.19	76.82
LASSO	59.54	73.34	75.20
TFEMPR	73.67	81.06	83.04
gOMP	58.37	73.84	74.92
BCS PL-2DBS + 2D DDWT	54.62	59.39	63.06
BCS SPL-2DBS + 2D DDWT	54.05	58.18	61.35

Compared with other methods, our method is on par with methods such as SIP and gOMP but lags behind methods such as LASSO and TFEMPR. We believe that our method of compressing the images with JPEG 2000 in lossless mode reduces the quality of the images when we go to very small values of BPS such as 0.1, 0.3 or 0.5.

3.4 Pavia University HSI

In Figure 3.7 we have shown the Pavia University hyperspectral image bands where each band captures a different wavelength reflected from the area. From the images below we can see the department buildings. In band number 10, the building roofs can be seen more clearly whereas in band 100 of the image we can see the area of the university more clearly.

After determining that compressing the images in 16×16 blocks and using Graph Fourier Transform on LLLLL band outputs the highest PSNR values.

We compress the Pavia University hyperspectral image for different BPS values. We will decorrelate the spectral bands using five integer based filters such as Haar, 2/6, 2/10, 5/3, 9/7-m. We will follow the same path and we will be comparing between LLLLL and LLLLH bands

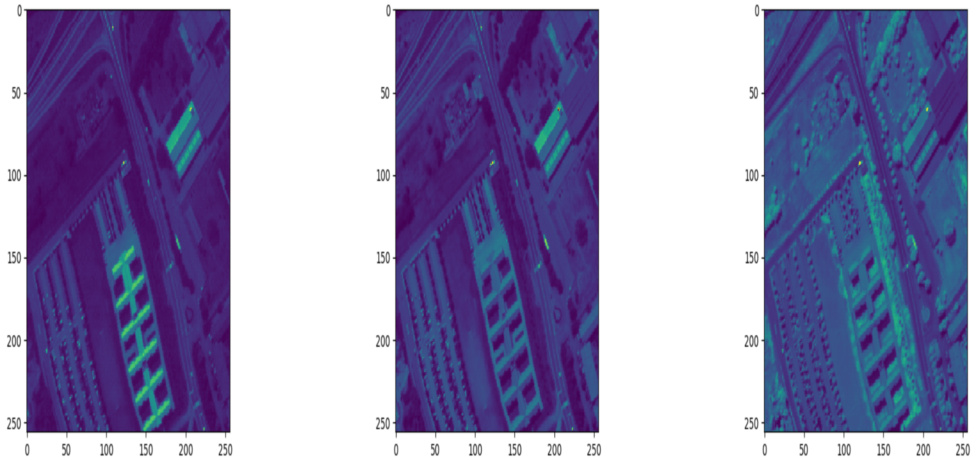


Figure 3.7 : Pavia University band 10, 50, 100.

Table 3.20 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.1

Methods Filter	LLLLH	LLLLL
	PSNR (dB)	PSNR (dB)
Haar	44.38	50.37
5/3	45.47	48.14
2/6	44.34	50.69
CDF 9/7-m	44.29	48.18
2/10	45.42	50.54

As we can see from Table 3.20, for a BPS of 0.1 using Graph Fourier Transform on LLLLL bands gives better PSNR values. When we use LLLLH bands, we see similar results with small change for all the filters. When we use LLLLL bands, we conclude that the filter 2/6 gives the best PSNR value of 50.69 dB for a BPS equal to 0.1.

Table 3.21 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.3

Methods Filter	LLLLH	LLLLL
	PSNR (dB)	PSNR (dB)
Haar	49.24	54.62
5/3	50.16	52.06
2/6	49.19	55.14
CDF 9/7-m	50.05	51.96
2/10	49.13	54.92

In Table 3.21, for a BPS of 0.3, using Graph Fourier Transform on LLLLL bands gives better PSNR values. We see that the filter 2/10 gives the best PSNR value of

55.14 dB for a BPS equal to 0.3. It is important to mention that there is not a huge difference between different filters but there is a significant difference if we apply Graph Fourier Transform between LLLLL or LLLLH. LLLLH bands contain detail coefficients whereas LLLLL contain approximation coefficients.

Table 3.22 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.5

Methods Filter	LLLLH	LLLLL
	PSNR (dB)	PSNR (dB)
Haar	51.55	57.00
5/3	52.44	54.25
2/6	51.50	57.60
CDF 9/7-m	52.25	53.97
2/10	51.42	57.41

In Table 3.22, for a BPS of 0.5 using Graph Fourier Transform on LLLLL bands gives better PSNR values. We see that the filter 2/10 gives the best PSNR value of 57.41 dB for a BPS equal to 0.5. It is important to mention that there is not a huge difference between the results from different filters. They produce similar results with a small difference among them.

From Table 3.23, we compressed the image using different filters and compared the reconstructed hyperspectral image with the original hyperspectral images by their PSNR values.

Table 3.23 : PSNR Vs BPS for Pavia University.

BPS Filter	0.1	0.3	0.5
Haar	50.37 dB	54.62 dB	57.00 dB
5/3	48.14 dB	52.06 dB	54.25 dB
2/6	**50.69** dB	**55.14** dB	**57.60** dB
CDF 9/7-m	48.18 dB	51.96 dB	53.96 dB
2/10	50.54 dB	54.92 dB	57.41 dB

As bit rate increases we see an increase in PSNR values. The 2/6 filter, which is commonly used for lossy compressions, gives almost a good result as filer 2/10 and 9/7-m. Their higher result also results in higher computation time.

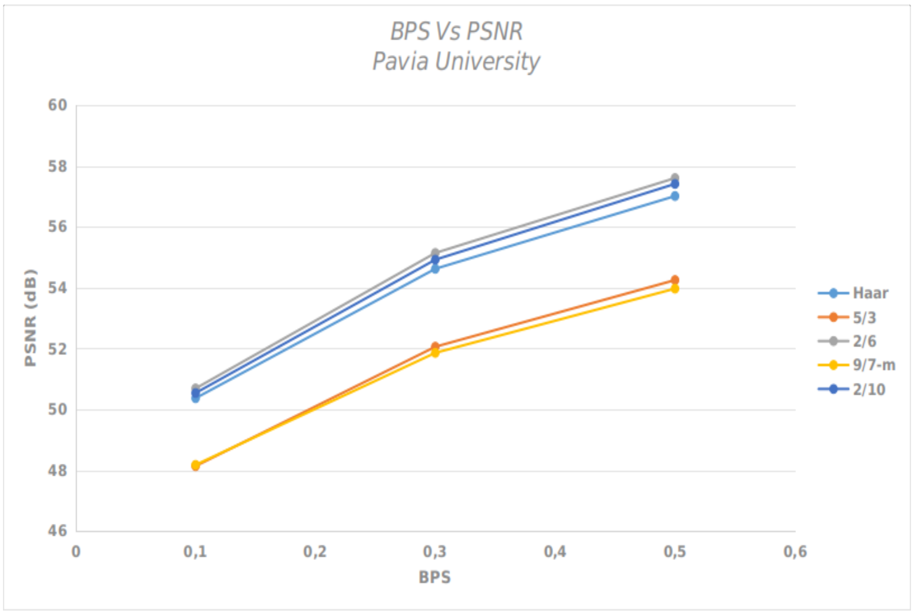


Figure 3.8 : A comparion of different filter for Pavia University hyperspectral image.

In figure 3.8, we can see that filter 9/7-m and filter 5/3 produce almost the same result but lack behind the other filters. Filter 2/6 produces the best result when compared with the Haar filter and 2/10 filters.

3.5 Botswana HSI

In Figure 3.9 we have shown Botswana hyperspectral image bands where each band captures a different wavelength reflected from the area. From the images below we can see the green fields with a small lake.

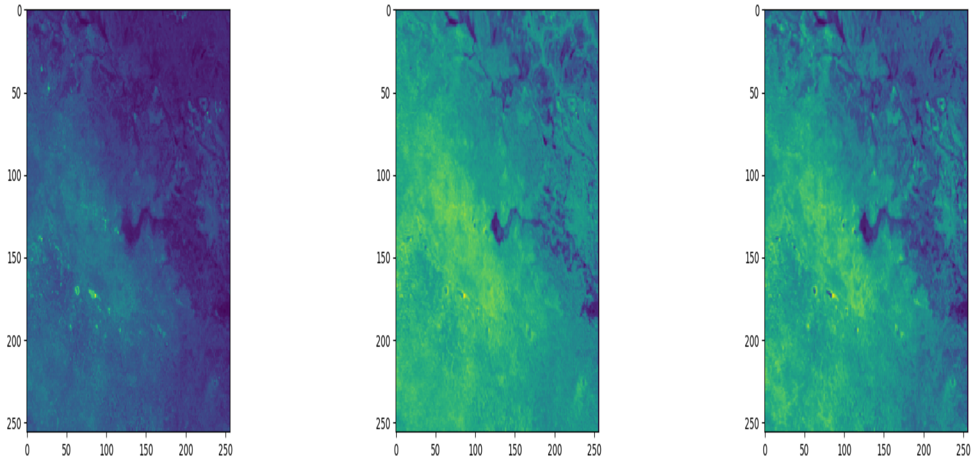


Figure 3.9 : Botswana HSI band 10, 50, 100.

After determining that compressing the images in $16{\times}16$ blocks and using Graph Fourier Transform on LLLLL band outputs the highest PSNR values.

We compress the Botswana hyperspectral image for different BPS values. We will decorrelate the spectral bands using five integer based filters such as Haar, 2/6, 2/10, 5/3, 9/7-m. We will follow the same path and we will be comparing between LLLLL and LLLLH bands

Table 3.24 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.1

Methods / Filter	LLLLH	LLLLL
	PSNR (dB)	PSNR (dB)
Haar	51.12	54.03
5/3	51.39	54.39
2/6	51.10	54.35
CDF 9/7-m	51.26	54.36
2/10	51.08	54.34

As we can see from Table 3.24, for a BPS of 0.1 using Graph Fourier Transform on LLLLL bands gives better PSNR values. When we use LLLLH bands, we see similar results with small change for all the filters. When we use LLLLL bands, we conclude that the filter 5/3 gives the best PSNR value of 54.39 dB for a BPS equal to 0.1.

Table 3.25 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.3

Methods / Filter	LLLLH	LLLLL
	PSNR (dB)	PSNR (dB)
Haar	52.15	57.43
5/3	52.42	58.24
2/6	52.11	58.36
CDF 9/7-m	52.32	58.23
2/10	52.06	58.40

In Table 3.25, for a BPS of 0.3, using Graph Fourier Transform on LLLLL bands gives better PSNR values. We see that the filter 2/10 gives the best PSNR value of 58.40 dB for a BPS equal to 0.3. It is important to mention that there is not a huge difference between different filters but there is a significant difference if we apply Graph Fourier Transform between LLLLL or LLLLH. LLLLH bands contain detail coefficients whereas LLLLL contain approximation coefficients.

Table 3.26 : Compression of LLLLL and LLLLH using Graph Fourier Transform for BPS = 0.5

Filter \ Methods	LLLLH	LLLLL
	PSNR (dB)	PSNR (dB)
Haar	54.13	59.07
5/3	54.09	59.24
2/6	54.13	59.24
CDF 9/7-m	54.26	59.24
2/10	54.09	59.25

In Table 3.26, for a bitrate of 0.5 using Graph Fourier Transform on LLLLL bands gives better PSNR values. We see that the filter 2/10 gives the best PSNR value of 59.25 dB. It is important to mention that there is not a huge difference between the results from different filters. They produce similar results with a small difference among them.

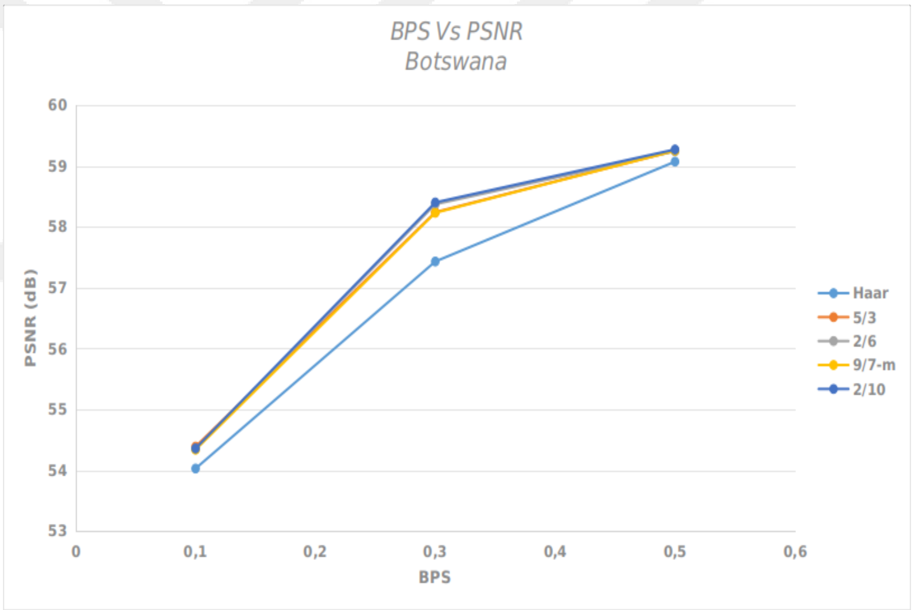


Figure 3.10 : A comparion of different filter for Pavia University hyperspectral image.

At table 3.27, we have compared the result obtained by the coding scheme for the Botswana HSI. We compare the PSNR vs BPS for BPS values from 0.1 to 0.5. We see that filter 20/10 produces the best result for bitrates equal to 0.3 and 0.5. For BPS = 0.1 5/3 filter produces the best result but there is not much difference between the filters. As for bitrate increase the PSNR value increases but the increase is not very big. The difference between filters is very small and negligible.

Table 3.27 : Results of our coding scheme over different bitrates

BPS Filter	0.1	0.3	0.5
Haar	54.03 dB	57.43 dB	59.07 dB
5/3	**54.39** dB	58.24 dB	59.25 dB
2/6	54.35 dB	58.36 dB	59.24 dB
CDF 9/7-m	54.36 dB	58.23 dB	59.24 dB
2/10	54.34 dB	**58.40** dB	**59.25** dB

In the end, we will show the results of our coding scheme on all images. In table3.28, we will show the result obtained for all five hyperspectral images and we will compare their image distortion with three different bitrates

Table 3.28 : Comparison of results for different hyperspectral Images

BPS HSI	0.1	0.3	0.5
	PSNR (dB)	PSNR (dB)	PSNR (dB)
Pavia University	50.69	55.14	57.60
Botswana	54.39	58.40	59.25
Lunar Lake	62.11	68.39	71.14
Low Altitude	57.45	61.75	63.50
Jasper Ridge	59.15	63.60	65.77

Using our method, Pavia University HSI has a PSNR increase from 50.69 dB to 57.60 dB as bitrate increases from 0.1 to 0.3. The image quality gain is comparable with the other methods. Our method produces good results since compressing to very low bitrates means the image quality loss is very high.

The same thing can be said for Botswana HSI. These images are captured with different imaging systems when compared with the last three hyperspectral images. The gain in these images is very small when we compare it with the gain that AVIRIS hyperspectral images produce.

Lunar Lake HSI has the highest gain in image quality with a PSNR increase from 57.45 dB to 71.14 dB. Lunar Lake has the highest PSNR gain when compared with the other images whereas Botswana HSI has the smallest PSNR gain for the same bitrate values.

We see that the coding scheme can compress hyperspectral images with very low bitrates while preserving the image quality. Each image PSNR increases as bitrate increases. Graph Fourier Transform has been used to compress images by treating them as signals and using only a few coefficients to reconstruct the images by using only a few coefficients.

4. CONCLUSIONS

Here, we introduce a novel compressing scheme for HSI is implemented. in order to achieve bit-rates between 0.1 and 1, we compress the images using JPEG 2000 in a lossy mode. We firstly decorrelate the hyperspectral images so we can maximize the compression ratio while preserving the Image quality. In the spectral domain, we employ integer based Discrete Wavelet Transform so that the coefficients are compressed as integer numbers thus reducing memory storage. In the spatial domain, we used Graph Fourier Transform to efficiently save a few coefficients that will help us reconstruct the image.

We have used two sets of hyperspectral images: AVIRIS and HYPERION datasets which are publicly available data. The hyperspectral images were cropped into a smaller size to reduce computational complexity. Results obtained from our coding scheme shows us that we have an improvement of image quality when compared to JPEG 2000 compression scheme and it's on par with some other methods. We managed to obtain very good PSNR values with respect to the method we choose to compress the hyperspectral images.

In order to further improve the results of our coding scheme and to compare it with the other methods, we believe that the same hyperspectral images with the same cube size must be used as a standard dataset. Since most of the researchers don't specify which part of the cube and which flight number they use. In future work, we will use more Graph Signal Processing tools such as graph filtering, graph wavelets, etc, to increase the quality of the reconstructed hyperspectral image. Further research will be conducted finding other ways to compress the hyperspectral images.

YK

REFERENCES

[1] **Teke, M.**, **Deveci, H.S.**, **Haliloğlu, O.**, **Gürbüz, S.Z. and Sakarya, U.** (2013). A short survey of hyperspectral remote sensing applications in agriculture, *2013 6th International Conference on Recent Advances in Space Technologies (RAST)*, IEEE, pp.171–176.

[2] **Govender, M.**, **Chetty, K. and Bulcock, H.** (2009). A review of hyperspectral remote sensing and its application in vegetation and water resource studies.

[3] **Parlak, C. and Bilgin, G.** (2014). Compression of hyperspectral images: A comparative study, *2014 22nd Signal Processing and Communications Applications Conference (SIU)*, pp.200–203.

[4] **Motta, G.** (2009). *Hyperspectral Data Compression*, Springer-Verlag, Berlin, Heidelberg.

[5] **King, G.**, **Seldev, C.C. and Singh, N.A.** (2014). A Novel Compression Technique for Compound Images Using Parallel Lempel-Ziv-Welch Algorithm, *Applied Mechanics and Materials*, volume626, Trans Tech Publ, pp.44–51.

[6] **Zhang, L.**, **Zhang, L.**, **Tao, D.**, **Huang, X. and Du, B.** (2015). Compression of hyperspectral remote sensing images by tensor approach, *Neurocomputing*, *147*, 358–363.

[7] **Mielikainen, J. and Huang, B.** (2012). Lossless compression of hyperspectral images using clustered linear prediction with adaptive prediction length, *IEEE geoscience and remote sensing letters*, *9*(6), 1118–1121.

[8] **Mielikainen, J.** (2006). Lossless compression of hyperspectral images using lookup tables, *IEEE signal processing letters*, *13*(3), 157–160.

[9] **Töreyın, B.U.**, **Yilmaz, O.**, **Mert, Y.M. and Türk, F.** (2015). Lossless hyperspectral image compression using wavelet transform based spectral decorrelation, *2015 7th International Conference on Recent Advances in Space Technologies (RAST)*, pp.251–254.

[10] **Töreyin, B.U.**, (2016), Spectral decorrelation of hyperspectral imagery using fractional wavelet transform, `https://doi.org/10.1117/12.2224579`.

[11] **Karaca, A.C. and Güllü, M.K.** (2018). Lossless hyperspectral image compression using bimodal conventional recursive least-squares, *Remote Sensing Letters*, *9*(1), 31–40, `https://doi.org/10.1080/2150704X.2017.1375612`, `https://doi.org/10.1080/2150704X.2017.1375612`.

[12] **Pal, M.D.**, **Brislawn, C.M. and Brumby, S.** (2002). Feature extraction from hyperspectral images compressed using the JPEG-2000 standard, *Proceedings Fifth IEEE Southwest Symposium on Image Analysis and Interpretation*, IEEE, pp.168–172.

[13] **Tang, X.**, **Pearlman, W.A. and Modestino, J.W.** (2003). Hyperspectral image compression using three-dimensional wavelet coding, *Image and Video Communications and Processing 2003*, volume5022, International Society for Optics and Photonics, pp.1037–1048.

[14] **Gündoğar, Z. and Demiralp, M.** (2015). Formulation of tridiagonal folmat enhanced multivariance products representation (TFEMPR), *AIP Conference Proceedings*, volume1702, AIP Publishing, p.170005.

[15] **Alaydın, J.G. and Töreyin, B.U.** (2016). Sparse coding based compression of spectrally uncorrelated hyperspectral data using Haar wavelet transform, *2016 24th Signal Processing and Communication Application Conference (SIU)*, pp.1945–1948.

[16] **Hayder, J.**, Hyperspectral image compression using sparse representations and wavelet transform based spectral decorrelation.

[17] **Zhu, X.**, **Ghahramani, Z. and Lafferty, J.D.** (2003). Semi-supervised learning using gaussian fields and harmonic functions, *Proceedings of the 20th International conference on Machine learning (ICML-03)*, pp.912–919.

[18] **Sandryhaila, A. and Moura, J.M.** (2014). Discrete signal processing on graphs: Frequency analysis, *IEEE Transactions on Signal Processing*, *62*(12), 3042–3054.

[19] **Huang, W.**, **Marques, A.G. and Ribeiro, A.** (2017). Collaborative filtering via graph signal processing, *2017 25th European Signal Processing Conference (EUSIPCO)*, IEEE, pp.1094–1098.

[20] **Cheung, G.**, **Magli, E.**, **Tanaka, Y. and Ng, M.K.** (2018). Graph spectral image processing, *Proceedings of the IEEE*, *106*(5), 907–930.

[21] **Hu, W.**, **Cheung, G.**, **Ortega, A. and Au, O.C.** (2015). Multiresolution graph fourier transform for compression of piecewise smooth images, *IEEE Transactions on Image Processing*, *24*(1), 419–433.

[22] **Zeng, J.**, **Cheung, G.**, **Chao, Y.H.**, **Blanes, I.**, **Serra-Sagristà, J. and Ortega, A.** (2017). Hyperspectral image coding using graph wavelets, *2017 IEEE International Conference on Image Processing (ICIP)*, IEEE, pp.1672–1676.

[23] **Adams, M.D. and Kossentni, F.** (2000). Reversible integer-to-integer wavelet transforms for image compression: performance evaluation and analysis, *IEEE Transactions on Image Processing*, *9*(6), 1010–1024.

[24] **Perozzi, B.**, **Al-Rfou, R. and Skiena, S.** (2014). DeepWalk: Online Learning of Social Representations, *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*,

KDD '14, ACM, New York, NY, USA, pp.701–710, `http://doi.acm.org/10.1145/2623330.2623732`.

[25] **Brin, S. and Page, L.** (1998). The Anatomy of a Large-Scale Hypertextual Web Search Engine, *Seventh International World-Wide Web Conference (WWW 1998)*, `http://ilpubs.stanford.edu:8090/361/`.

[26] **Jabłoński, I.** (2017). Graph Signal Processing in Applications to Sensor Networks, Smart Grids, and Smart Cities, *IEEE Sensors Journal*, *17*(23), 7659–7666.

[27] **Milanfar, P.** (2013). A Tour of Modern Image Filtering: New Insights and Methods, Both Practical and Theoretical, *IEEE Signal Processing Magazine*, *30*(1), 106–128.

[28] **Hore, A. and Ziou, D.** (2010). Image Quality Metrics: PSNR vs. SSIM, *2010 20th International Conference on Pattern Recognition*, pp.2366–2369.

[29] **Marcellin, M.W.**, **Gormish, M.J.**, **Bilgin, A. and Boliek, M.P.** (2000). An overview of JPEG-2000, *Proceedings DCC 2000. Data Compression Conference*, pp.523–541.

[30] Hyperion sensor, `https://eo1.gsfc.nasa.gov/`, accessed: 2019-05-02.

[31] ROSIS sensor, `http://www.ehu.eus/ccwintco/index.php?title=Hyperspectral_Remote_Sensing_Scenes#Pavia_Centre_and_University`, accessed: 2019-05-02.

[32] **Ülkü, İ. and Töreyin, B.U.** (2015). Sparse representations for online-learning-based hyperspectral image compression, *Applied optics*, *54*(29), 8625–8631.

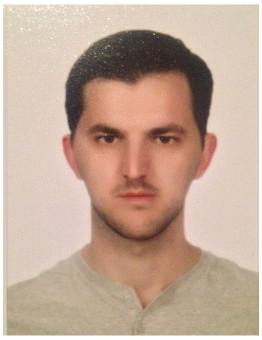
[33] **Fowler, J.E.** (2009). Compressive-projection principal component analysis, *IEEE transactions on image processing*, *18*(10), 2230–2242.

[34] **Koh, K.**, **Kim, S.J. and Boyd, S.** (2007). An interior-point method for large-scale l1-regularized logistic regression, *Journal of Machine learning research*, *8*(Jul), 1519–1555.

[35] **Wang, J.**, **Kwon, S. and Shim, B.** (2012). Generalized orthogonal matching pursuit, *IEEE Transactions on signal processing*, *60*(12), 6202–6216.

[36] **Eslahi, N.**, **Aghagolzadeh, A. and Andargoli, S.M.H.** (2014). Block compressed sensing images using accelerated iterative shrinkage thresholding, *2014 22nd Iranian Conference on Electrical Engineering (ICEE)*, pp.1569–1574.

[37] **Sungkwang Mun and Fowler, J.E.** (2009). Block compressed sensing of images using directional transforms, *2009 16th IEEE International Conference on Image Processing (ICIP)*, pp.3021–3024.



CURRICULUM VITAE

Name Surname: Indrit Nallbani

Place and Date of Birth: 29/01/1993, KAVAJE / ALBANIA

E-Mail: nallbaniindrit@gmail.com

EDUCATION:

- **B.Sc.:** 2011 - 2017, Middle East Technical University, Faculty of Arts and Science, Department of Physics
- **M.Sc.:** 2017 - Present, Istanbul Technical University, Institute of Informatics, Department of Applied Informatics

PROFESSIONAL EXPERIENCE AND REWARDS:

- 2013 - 2013: Internship at Center for Solar Energy Research and Applications, ANKARA/TURKEY
- 2015 - 2016: Research Assistant at The Laboratory for Computational Ontology, ANKARA/TURKEY
- 2018 - Present: Graduate Research Assistant at Vodafone İTÜ Future Lab, ISTANBUL/TURKEY

PUBLICATIONS, PRESENTATIONS AND PATENTS ON THE THESIS:

- **Indrit Nallbani**, Behçet Uğur Töreyin, A Novel Hyperspectral Image Compression Scheme Using Graph Fourier Transform 2019. *Signal Processing and Communications Applications (SIU 2019)*, March 24-26, 2019 Sivas, Turkey.