Urban Scene Detection by Material Mapping Using Hyper Spectral Imaging

Author

Syeda Sara Habib

2011-NUST-MSPhD-Mts-31

Supervisor

Brig. Dr. Javaid Iqbal

DEPARTMENT OF MECHATRONICS ENGINEERING

COLLEGE OF ELECTRICAL & MECHANICAL ENGINEERING

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY

ISLAMABAD

AUGUST, 2015

Urban Scene Detection by Material Mapping Using Hyper Spectral Imaging

Author

Syeda Sara Habib

2011-NUST-MSPhD-Mts-31

A thesis submitted in partial fulfillment of the requirements for the degree of

MS Mechatronics Engineering

Thesis Supervisor:

Brig. Dr. Javaid Iqbal

Thesis Supervisor's Signature:____

DEPARTMENT OF MECHATRONICS ENGINEERING

COLLEGE OF ELECTRICAL & MECHANICAL ENGINEERING

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY, ISLAMABAD

AUGUST, 2015

Declaration

I certify that this research work titled "*Urban Scene Detection by Material Mapping Using Hyper Spectral Imaging"* is my own work. The work has not been presented elsewhere for assessment. The material that has been used from other sources has been properly acknowledged / referred.

Signature of Student
Syeda Sara Habib
2011-NUST-MSPhD-Mts-31

Language Correctness Certificate

This thesis has been read by an English expert and is free of typing, syntax, semantic, grammatical and spelling mistakes. Thesis is also according to the format given by the university.

Signature of Student

Syeda Sara Habib

2011-NUST-MSPhD-Mts-31

Copyright Statement

Acknowledgements

First of all, I want to thank Allah, the Almighty, for making things happen in miraculous ways.

I would like to thank Dr. Rab Nawaz Chaudhry for the guidance, for all the support and for making this thesis possible. I especially have to mention his brilliant coding skills where he looks at a code and figures out the issue just with one scratch on the head.

A special thanks to Brig. Dr. Javaid Iqbal for making things easier and being such a supportive supervisor.

My parents, my friends, my colleagues, who were there to keep the morale high.

Abstract

Hyperspectral remote sensors collect images in hundreds of narrow adjacent spectral bands. This image data at pixel level is compared with field or laboratory reference spectra in order to recognize and map surface materials present at each pixel. The material mapping is further used to build applications in agriculture, food processing, surface mineralogy, chemical imaging, and surveillance. Techniques have also been proposed in the literature, which utilize material mapping as well as spatial context of materials to recognize a particular target.

We propose a novel material-mapping algorithm, which relies on the fact that pixels belonging to the same class but located at different positions in the image exhibit variability in their spectral signatures. This could be due to the difference in terrain, atmosphere and surrounding materials. Therefore, a pixel will better match to the neighboring rather than distant pixels of its own class. The algorithm avoids configuring an SVM and at the same time reduces the complexity of its Nearest Neighbor (NN) style matching scheme.

Our algorithm dynamically reduces the training set for each testing pixel. Median matching score of 20 spatially closest members of each class are compared to decide the fate of the testing pixel. Two matching algorithms namely Euclidean distance and Spectral Angle Mapper (SAM) are used. We know that SAM algorithm is robust to multiplicative distortion between test and reference spectra.

Our approach is different to, for example, using a Support Vector Machine (SVM). It resembles more to Nearest Neighbor (NN) algorithm. Its complexity is lower than that of NN owing to matching being performed with only limited number of training pixels. In case of SVM, learning takes long especially for long feature vectors. It is generally easier to deal with multiple-class problems with NN than SVM. Several parameters need to be tuned to get good accuracy and generalization from SVM.

We use unsupervised learning to help supervised learning by Euclidean and SAM classifiers. The basic idea is that all pixels belonging to a cluster should be classified to the same class. It greatly increases the accuracy of the first stage of our approach. The training data is utilized twice first by supervised learning algorithms and then by clustering. If a training pixel is present within a cluster, the whole cluster is classified as belonging to the training pixel class. Our results show that 2nd stage results into comparable accuracy of Euclidean and SAM algorithms. We perform clustering and material classification for the whole images in the datasets. The accuracy, though, is judged only on the ground truth pixels.

In most of the cases by target recognition, we mean a target material recognition. Other spatial target may be identified by the cluster analysis of pixels belonging to their material. In some instances, the material of their background may also help, e.g., a bridge is defined as concrete over water.

Our first stage uses hard classification of pixels. In the second stage, we have used K-means which provides hard clustering. Fuzzy C Mean (FCM) algorithm provides soft clustering and unmixing techniques provide fuzzy membership to each testing pixel. An interesting dimension would be to use FCM along with unmixing techniques to classify the pixels.

Key Words: *Hyper Spectral Image Processing, Material Mapping, Euclidean Distance, Spectral Angle Mapper (SAM), Reduced Training Nearest Neighbor Matching, Median Filtering, clustering*

Table of Contents

List of Figures

List of Tables

CHAPTER 1: INTRODUCTION

It is rightly said that a picture is worth a thousand words. The phrase should change now to say a picture is worth a million words as the cameras can now see many more bands of the electromagnetic spectrum.

One of the related concepts is “Hyperspectral Imagery”.

1.1 Concept

1.1.1 Hyperspectral Imagery

Remote sensing is imaging the earth or getting its information without touching it. Spectroscopy studies light and its reflectance or emission from various materials. Spectrometers use the optical remote sensing and can be used in air borne and space borne setting.

Hyperspectral consists of “Hyper” (to many) and “Spectral” (belonging to a spectrum). Color images provide information in only three R, G, and B bands. Hyperspectral images give pixel detail in the form of hundreds of contiguous bands.

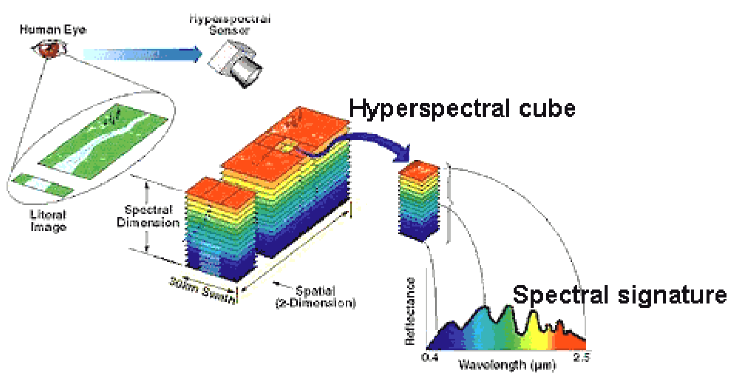

Figure 1.1: Display of hyperspectral data

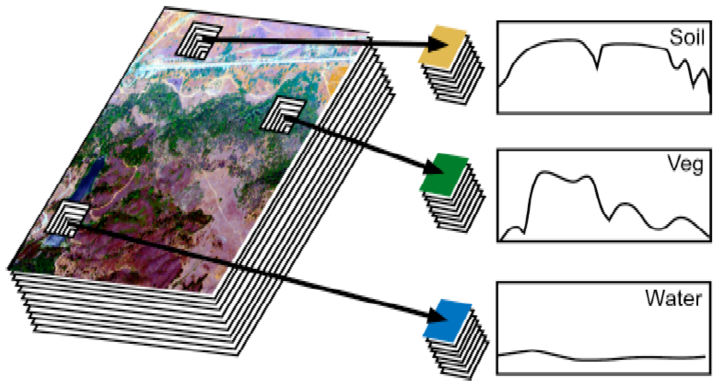

Figure 1.2: Image measurements are made many narrow contiguous bands, resulting in a complete spectrum of each pixel.

1.2 Hyperspectral Image: Multilayer Image

Hyperspectral image is a kind of multi-layer image. Each layer having different data plays its role in material mapping or pixel classification.

The figure below shows a good example of multilayered image.

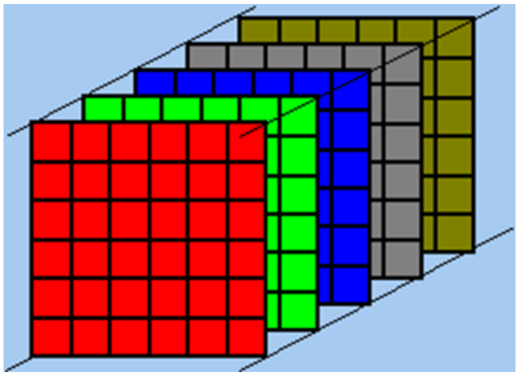

Figure 1.3: Hyperspectral imagery also comes in the category of multilayered imagery.

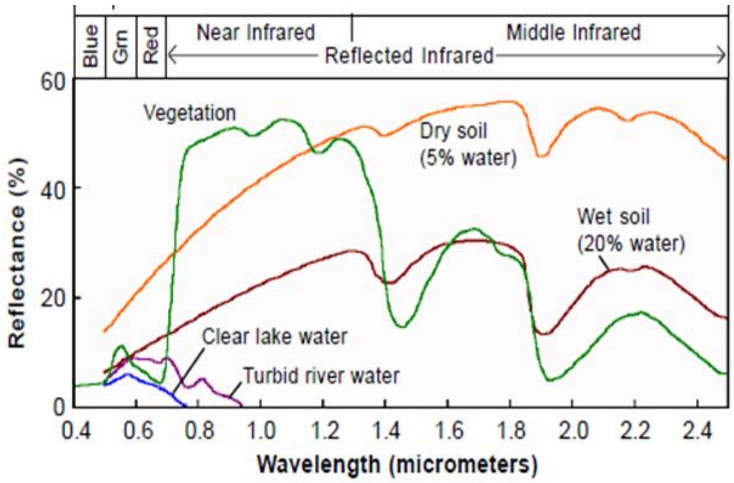

Figure 1.4: Signatures: Vegetation, water, soils etc

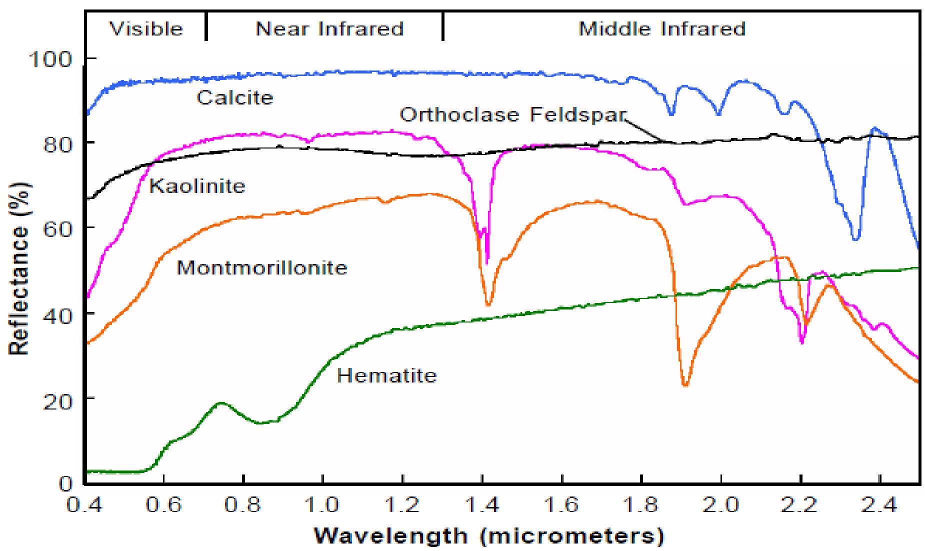

Figure 1.5: Signatures: Minerals

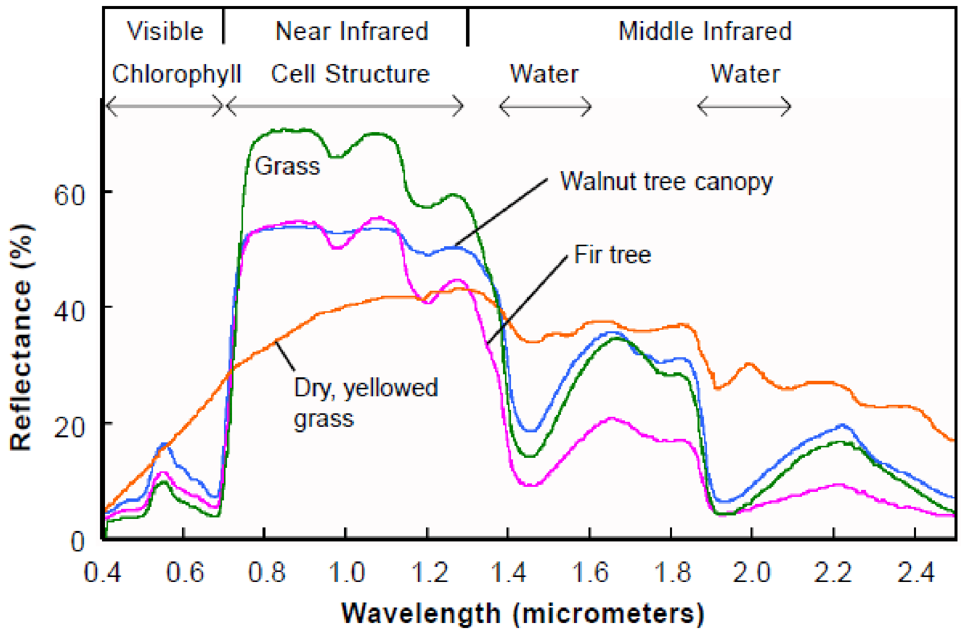

Figure 1.6: Signatures: Plants

Figures 1.3, 1.4 and 1.5 clearly show that different classes like Vegetation, water, soils, minerals and plants can be easily differentiated using hyperspectral signatures.

1.2.1 Material Mapping

A hyper-spectral image data can be used to find nature and location of materials present in the image.

The image in Figure (1.6) below shows material (mineral) mapping results.

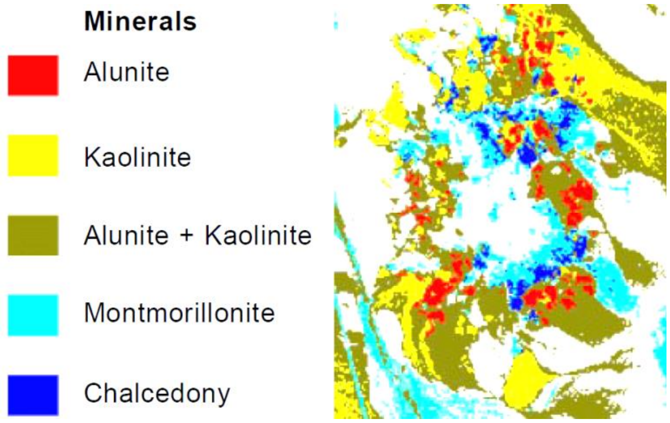

Figure 1.7: Material Mapping for Minerals

1.2.2 Sensors

There are multiple kinds of are sensors which are typically used to acquire the hyperspectrical images. They are also called “imaging spectrometers”. Their details are present in Annexure-A (table 1.1 and 1.2)

1.2.3 Spectral Libraries

Several libraries of reflectance spectra of natural and man-made materials are available for public use. These libraries provide a source of reference spectra that can aid the interpretation of hyperspectral and multispectral images.

ASTER Spectral Library: This library has been made available by NASA as part of the Advanced Spaceborne Thermal Emission and Reflection Radiometer (ASTER) imaging instrument program. It includes spectral compilations from NASA’s Jet Propulsion Laboratory, Johns Hopkins University, and the United States Geological Survey (Reston). The ASTER spectral library currently contains nearly 2000 spectra, including minerals, rocks, soils, man-made materials, water, and snow. Many of the spectra cover the entire wavelength region from 0.4 to 14 µm. The library is accessible interactively via the Worldwide Web at http://speclib.jpl.nasa.gov. One can search for spectra by category, view a spectral plot for any of the retrieved spectra, and download the data for individual spectra as a text file.

USGS Spectral Library: The United States Geological Survey Spectroscopy Lab in Denver, Colorado has compiled a library of about 500 reflectance spectra of minerals and a few plants over the wavelength range from 0.2 to 3.0 μm. This library is accessible online at http://speclab.cr.usgs.gov/spectral.lib04/spectral-lib04.html. One can browse individual spectra online, or download the entire library.

Target spectra can also be derived from regions of interest within a spectral image, or individual pixels within a spectral image.

1.2.4 Factors Affecting the Response from a Pixel

- Sun spectrum
    - The solar energy is dependent upon the wavelength and it peaks in the visible range.
    - The solar energy spectrum at the image acquisition time affects the hyperspectral data.
    - However, this may not change signatures from the same land cover types across the image.
- Amount of energy received from sun
    - Angle of incidence: The angle between the path of the incoming energy and a line perpendicular to the ground surface. It depends upon sun angle at the image acquisition time. Again, this may not change signatures from the same land cover types across the image.
    - *Terrain effect: Rough terrain may change signatures from the same land cover types across the image.*
    - *Shadowing by clouds, trees, and crop rows, may change signatures from the same land cover types across the image.*
- Atmosphere (incoming and reflected solar energy)

- ○ The atmosphere affects solar energy from and to the sun. Some bands energy is badly reduced in both paths. Note that in most of the datasets, these bands have been removed.
- ○ *If the atmosphere varies to a greater extent across the image/scene, or if the scene has high ground elevation variability, we may notice change in signatures from the same land cover types across the image.*

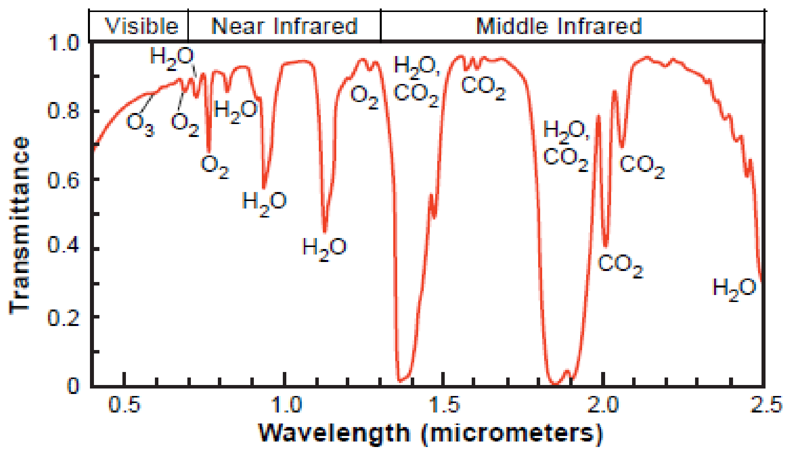

Figure 1.8: Atmospheric transmittance

- Sensor effects
  - ○ *Atmosphere effects the way a hyperspectral sensor takes an image, so it requires favorable conditions.*
  - ○ *Electronic malfunction can occur.*

1.1.6 Application Areas

- Agriculture
- Food Processing
- Mineralogy
- Surveillance
- Chemical Imaging

The primary advantage to hyperspectral imaging is that an entire spectrum is acquired at each point.

Hyperspectral imaging can also take advantage of the spatial relationships among the different spectra in a neighborhood, allowing more elaborate spectral-spatial models for a more accurate segmentation and classification of the image.

The primary disadvantages are cost and complexity.

The following requirements, at times, becomes a disadvantage for the system:

- Fast computers
- Sensitive detectors
- Large data storage capacities are needed for analyzing hyperspectral data

Chapter (1) was inspired by the References (1-8).

CHAPTER 2: LITERATURE REVIEW

2.1 SVM

There are two paradigms for material mapping using Hyper Spectral images. One is based on using an SVM (Support Vector Machine). SVM provides good accuracy even for small training sets [11]. In case of SVM, learning takes long especially for long feature vectors. It is generally difficult to deal with multiple-class problems with SVM. Several parameters need to be tuned to get good accuracy and generalization from SVM. Even then the accuracy of SVM only based classifiers is not impressive. Features estimated from the hyperspectral data are fed to SVMs to get reasonable accuracy rates. This further complicates the problem. The details about SVM are available in standard textbooks and are not included here for brevity.

The other uses one of the two following algorithms.

2.2 Algorithms

There are two major types of algorithms which are used for spectral matching

- Distance Based
- Angle Based

2.2.1 Distance Based

Where,

Ed = Euclidian Distance
t = target material reference signature
p = testing pixel signature
n = Number of Spectral bands

$$Ed = \sqrt{\sum_{i=1}^{n} (t_i - p_i)^2}$$

In this method, each pixel is matched against the total number of target materials of interest. The pixel is assigned a label of the material with which it has minimum dissimilarity Ed.

2.2.2 Angle Based (SAM)

Also known as the Spectral Angle Mapper (SAM)

Where,

α = Angle
t = target material reference signature
p = testing pixel signature
n = Number of Spectral bands

$$\alpha = \cos^{-1} \frac{\sum_{i=1}^{n} t_i p_i}{\sqrt{\sum_{i=1}^{n} t_i^2} \sqrt{\sum_{i=1}^{n} p_i^2}}$$

In this method, each pixel is matched against the total number of target materials of interest. The pixel is assigned a label of the material with which it has minimum Angle. SAM is resilient to multiplicative noise between the target and testing pixel spectra.

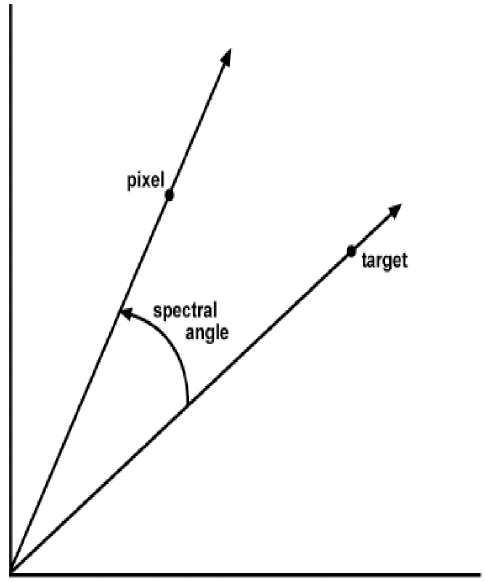

Figure 2.1: Spectral Angle is small even if two pixels have multiplicative distortion

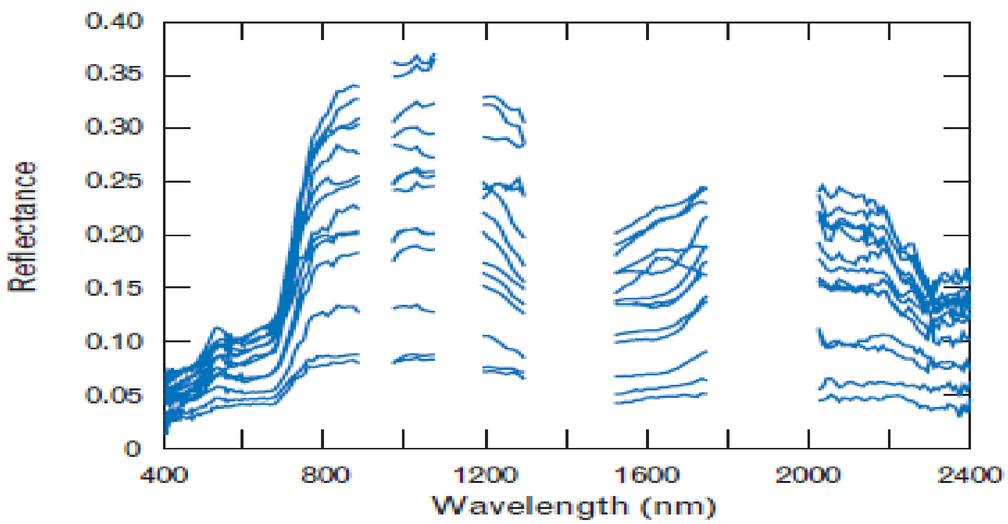

Figure 2.2: Signature for several pixels from the same target with multiplicative scaling effect

In above methods the target material reference signature is obtained either from publically available datasets or derived from the image itself [9].

Generally material mapping based on only Euclidean or SAM algorithms only, do not yield good accuracy [10].

2.3 Building material detection

An image from Compact Airborne Spectrographic Imager (CASI) is used in one of the experiments.

Table 2.1: Experimental Data

	Image
Acquisition Altitude	2540m
Resolution	4m
No of Bands	48
Pixels	64 X 64

Figure 2.3: Experimental Data: Top: The 64 x 64 image, Middle: Building material ground truth, Bottom: Average reference spectrum for building material

2.3.1 Evaluation of the algorithms

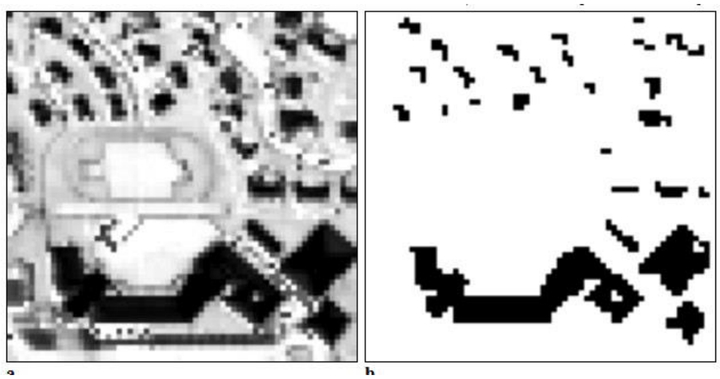

Figure 2.4: Evaluation of the algorithms, Left: Distance image, Right: Threshold distance image

A complete ROC curve can be obtained by changing the threshold. False positives are non-building pixels classified as buildings pixels. False negative pixels are building pixels classified as non-building pixels. True positive pixels are building pixels classified as building pixels. True negatives pixels are non-building pixels classified as non-building pixels.

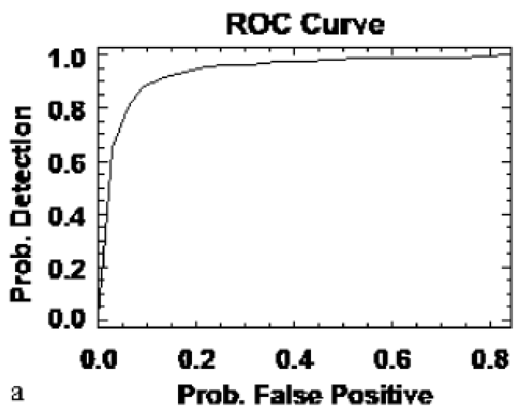

Figure 2.5: ROC Curve

Most algorithms assume spatially invariant model that applies to the whole hyperspectral image. This can be applicable for images spanning small areas but not to large area images [12]. The profile of the same class can substantially change across space due to varying soil type, terrain and climatic conditions. This decreases the accuracy achieved by conventional classifiers. We will address this problem in the next chapter.

CHAPTER 3: PROPOSED METHODOLOGY

3.1 Proposed Methodology

Most algorithms assume spatially invariant model that applies to the whole hyperspectral image. This can be applicable for images spanning small areas but not to large area images [12]. The profile of the same class can substantially change across space due to varying soil type, terrain and climatic conditions. This decreases the accuracy achieved by conventional classifiers.

We therefore propose to use spatially nearest pixels from the ground truth samples for each class. It is effectively a dynamically reduced training set scheme which otherwise resembles with Nearest Neighbor (NN) type algorithms. We tested Euclidean as well as SAM algorithms for matching.

We use unsupervised learning to help supervised learning by Euclidean and SAM classifiers. The basic idea is that all pixels belonging to a cluster should be classified to the same class. It greatly increases the accuracy of the first stage of our approach. The training data is utilized again by clustering. If a training pixel is present within a cluster, the whole cluster is classified as belonging to the training pixel class. Our results show that 2nd stage results into comparable accuracy of Euclidean and SAM algorithms. We perform clustering and material classification for the whole images in the datasets. The accuracy, though, is judged only on the ground truth pixels. We also exploit the unlabeled pixels in the hyper spectral image in our clustering technique. A scheme resembling our approach in its outlook is given in [13]. The details about K-means clustering are available in standard textbooks and are not included here for brevity.

The step by step procedure of our proposed scheme is given below

1) Create training data set to be equal to 10% of the provided ground truth data
2) Perform supervised classification of the whole image with Euclidean and SAM algorithm
    a. Set the number of near pixels to 20

b. Use median filtering to classify from dissimilarity of 20 pixels from each class

3) The method dynamically changes training data set for each pixel based on proximity.
4) Note down the accuracy. We call this stage 1.
5) Perform clustering of the whole image with K-means algorithm
6) Set K equal to the number of classes
7) To reduce the complexity uses every 10^{th} band from the feature vector for clustering.
8) Get pixels belonging to local connected regions of clusters.
9) If they contain any training pixel, change the classification of connected region pixels to be the mode of the training pixels classification.
10) Otherwise, change the classification of the connected region pixels to be their mode in the stage 1 classification.
11) Note down the accuracy, Perform multiple K-means iterations and record accuracy for each.
12) Average the accuracy. This is accuracy against first instance of the randomly chosen training set.
13) Do above procedure for 10 iterations of the randomly chosen training dataset.
14) Average the accuracy results.
15) The deliverables are high Overall Accuracy and Average Accuracy at the ground truth pixels and a smoother material mapping of the whole image.

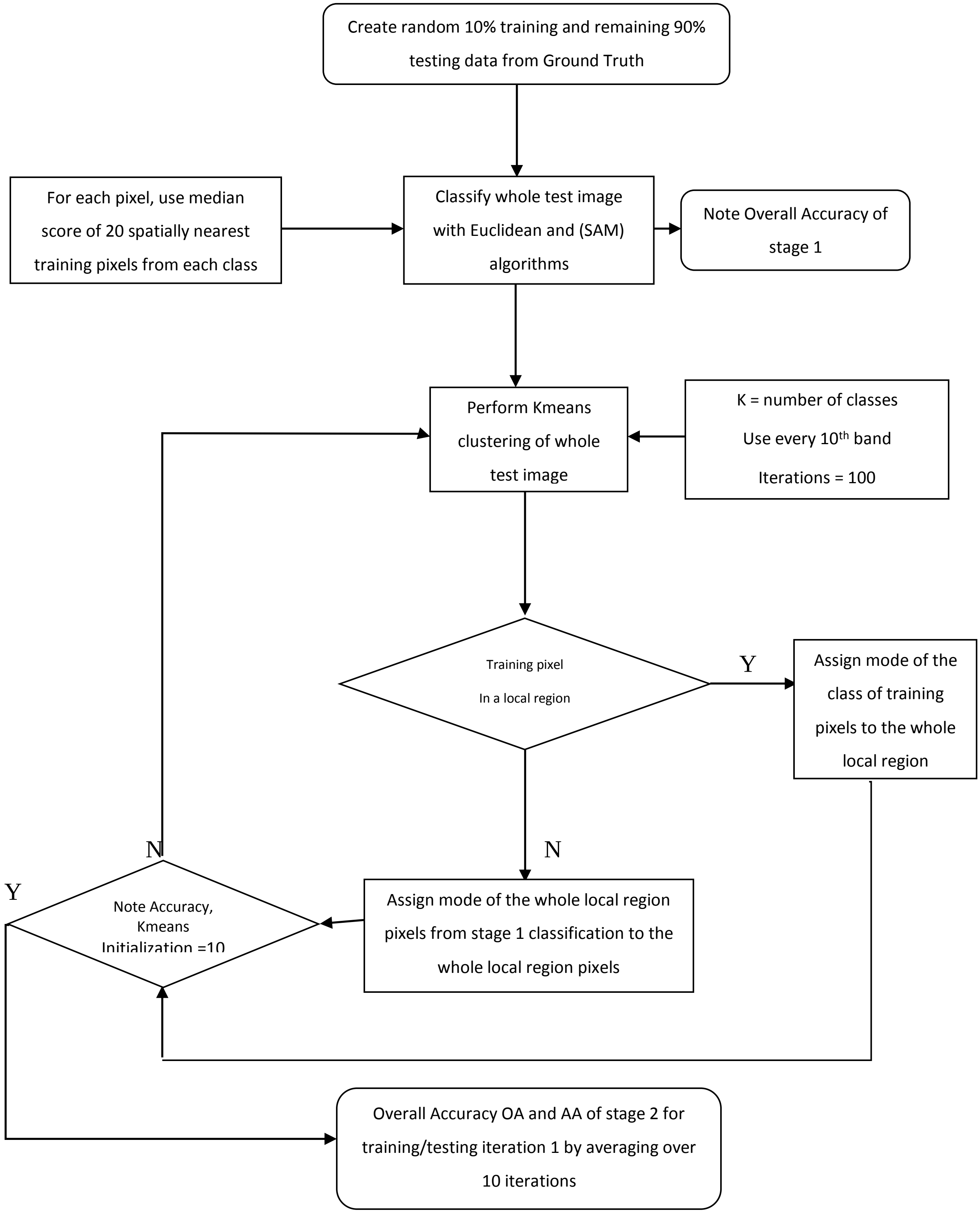

Figure 3.1: Flow diagram of the proposed methodology

Overall Accuracy (OA) and Average Accuracy (AA) are defined as

Overall Accuracy = Number of correctly classified pixels/Total number of pixels in ground truth

Average Accuracy = Average of the Overall Accuracies for each individual classes.

As noted in chapter (2), terrain, shadowing by clouds, crops and trees, and atmosphere may change signatures from the same land cover types across the image. The material present in different parts of the image will exhibit variability in its signatures. Note that labeling data with class labels is independent process mostly carried out with the help of maps or ground surveys. The pixels, e.g., from grass class in different parts of the image will have different signatures due to above mentioned reasons. If the training pixels of grass belong only to one area of the scene, there is a chance that the grass pixels from the other areas will be misclassified. This phenomenon results into low accuracy of stage 1 results. Even if our Kmeans algorithm clusters area 2 grass pixels with area 1 grass pixels, stage 2 won't be able to correctly classify area 2 grass pixels from its previous stage 1 classification. All it means that we need to put training pixels for each class across the whole scene. Is that a prohibitive or costly requirement? The answer is NO because it is recommended to visit 20 sites for generating training data for one land cover type with 40 pixels than one site with 800 pixels of the same class [14].

Therefore our approach hinges on two factors

- Availability of training pixels across the whole image for each class. This will give better accuracy at stage 1. But there is obviously a limit to producing dense training.
- Kmeans clustering correctly classifies those remaining incorrectly classified pixels which segment out as one cluster but were not homogenously classified by first stage.

We prove the efficacy of our proposed methodology in the next chapter through exhaustive simulations.

CHAPTER 4: EXPERIMENTS & RESULTS

We have used publicly available benchmark databases from [15] to evaluate the efficacy of our proposed algorithm. These images have been taken from different space borne and airborne sensors. We particularly have used the following images for our experiments:

- Pavia University, Pavia, Italy
- Indian Pines, North-western Indiana, USA
- Salinas, Salinas Valley, California, USA
- Salinas-A, Salinas Valley, California, USA
- Okavango Delta
- Kennedy Space Center, Florida, USA

4.1 Pavia University

This image has been provided by Pavia University. They have been taken from the ROSIS sensor over the town of Pavia in the North of Italy. The number of bands in this particular image are 103 (i.e there are 103 bands and each band has a different value for a single pixel). This image contains 9 classes. Please see the image and its ground truth below (the class sample number and other details are given in Annexure-A Table 4.1).

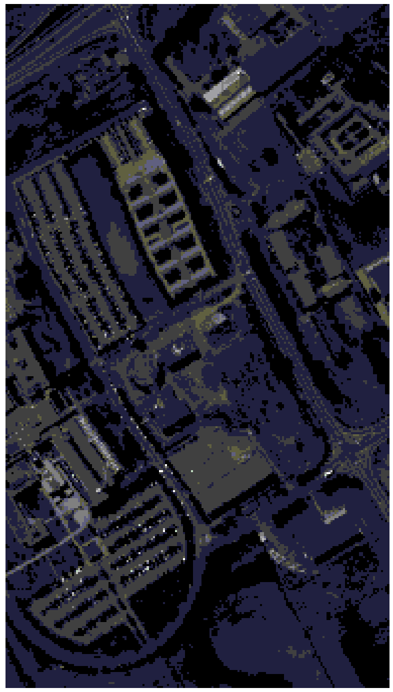

Figure 4.1: Pavia University

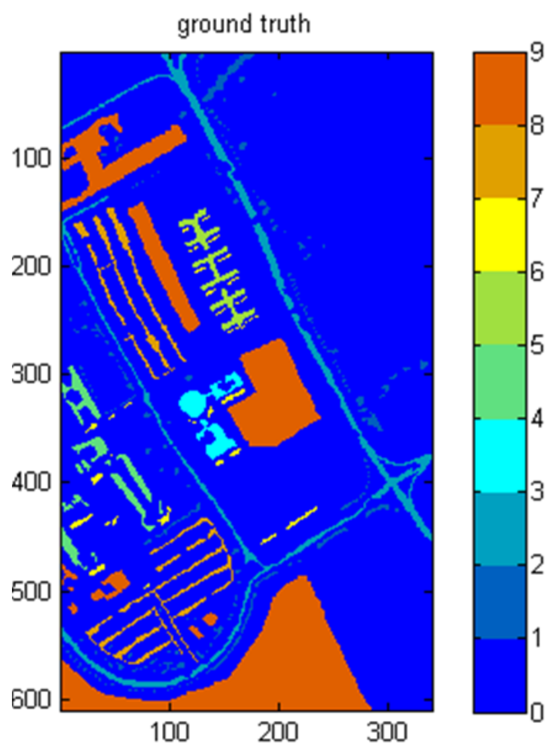

Figure 4.2: Ground Truth of Pavia University

4.2 Indian Pines

This image has been provided by Pursue's University. They have been taken from the AVIRIS sensor over the Indian Pines test site in North-western Indiana, in the United States of America. The number of bands in this particular image is 200 (i.e. there are 220 bands and each band has a different value for a single pixel). It contains 16 classes.

Please see the image and its ground truth below (the class sample number and other details are given in Annexure-A Table 4.2).

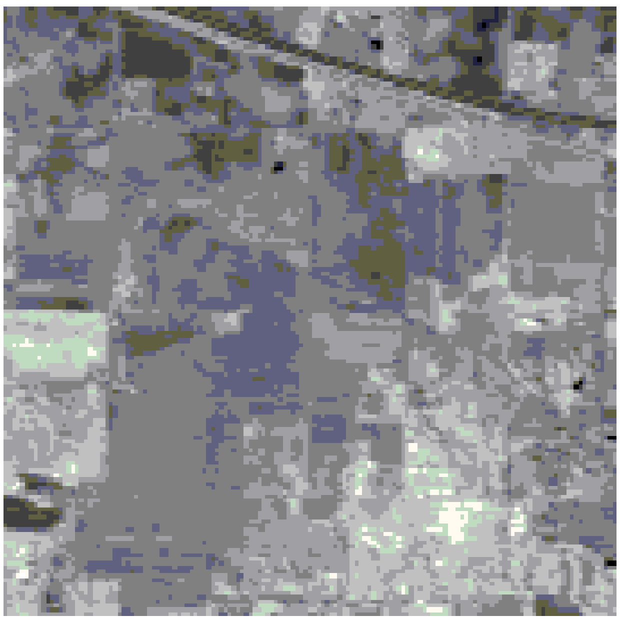

Figure 4.3: Indian Pines

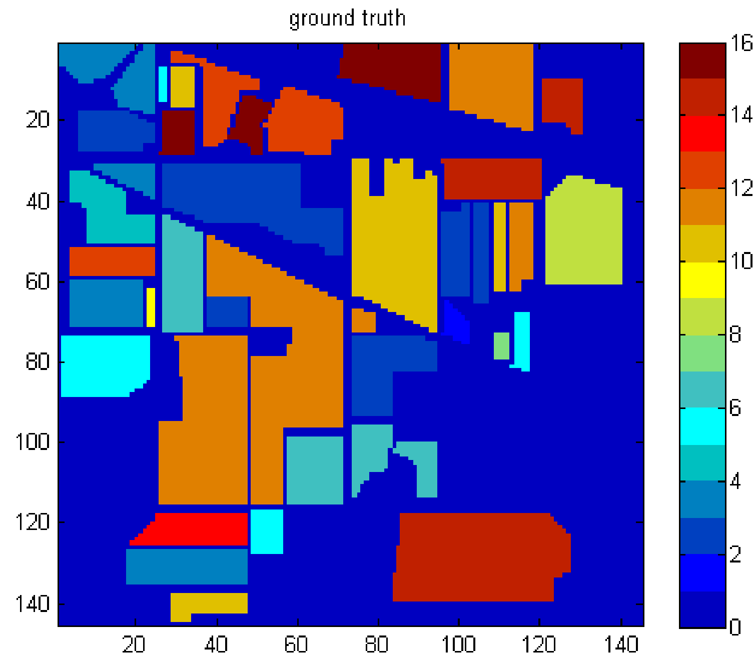

Figure 4.4: Ground Truth of Indian Pines

4.3 Salinas

This image has been taken from the AVIRIS sensor over Salinas Valley, California, in the United States of America. The number of bands in this particular image is 224 (i.e there are 224 bands and each band has a different value for a single pixel).

The image has been taken for the vegetation experiment and hence, it includes Vegetables, soil, and vine fields. It contains 16 classes.

Please see the image and its ground truth below (the class sample number and other details are given in Annexure-A Table 4.3).

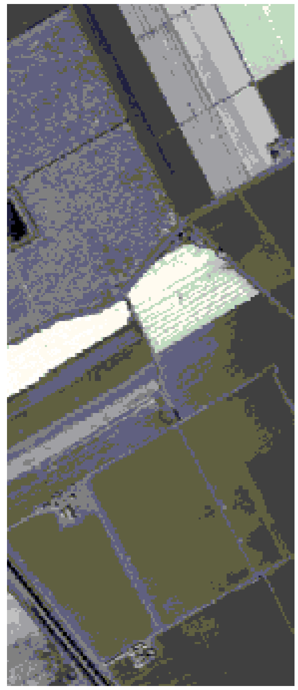

Figure 4.5: Salinas

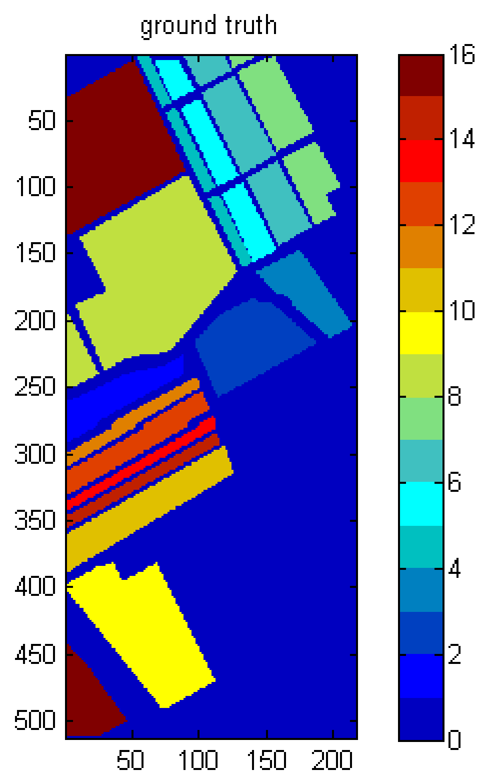

Figure 4.6: Ground Truth of Salinas

4.4 Salinas-A

This image has been taken from the AVIRIS sensor over Salinas Valley, California, in the United States of America. The number of bands in this particular image is 224 (i.e there are 224 bands and each band has a different value for a single pixel).

The image has been taken for the vegetation experiment and hence, it includes Vegetables, soil, and vine fields. It contains 6 classes. It is a part of the previous image.

Please see the image and its ground truth below (the class sample number and other details are given in Annexure-A Table 4.4).

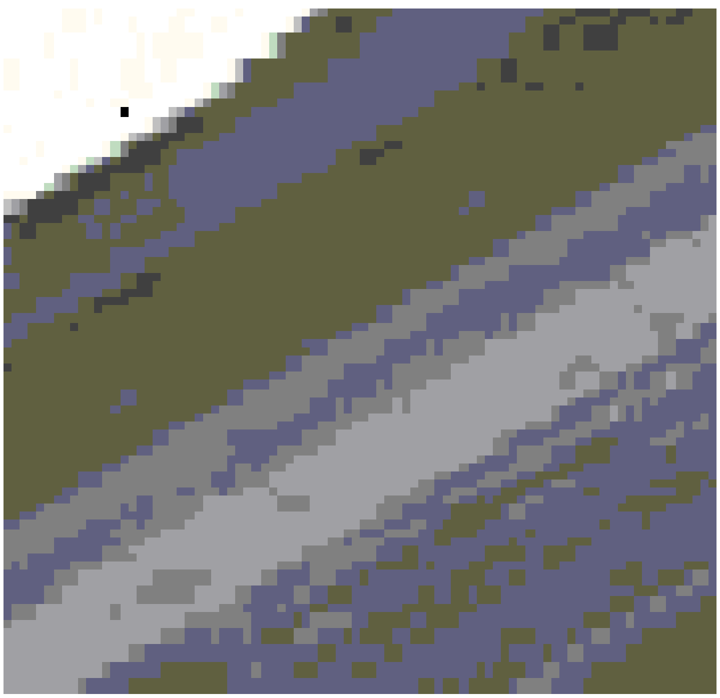

Figure 4.7: Salinas-A

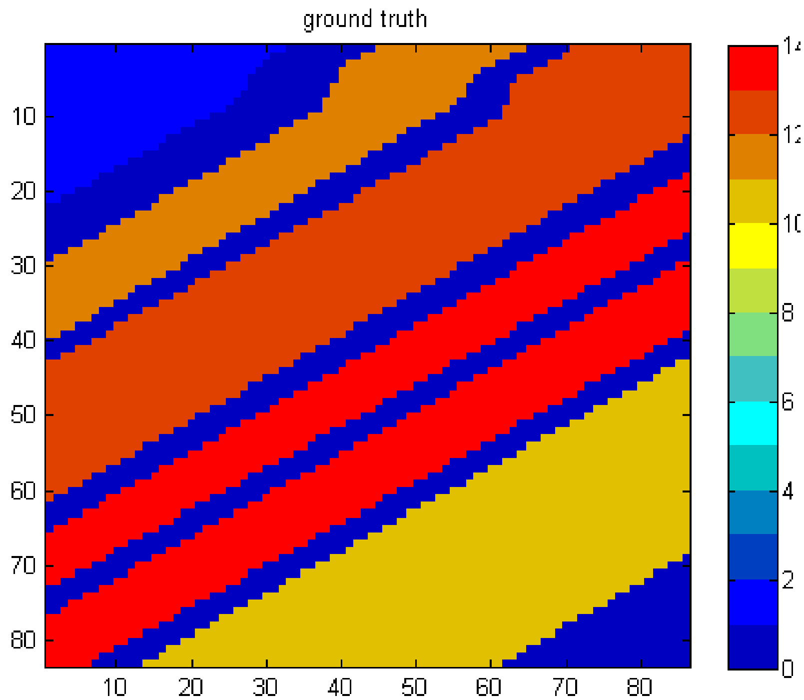

Figure 4.8: Ground Truth of Salinas-A

4.5 Botswana

This image has been taken from the NASA EO-1 satellite (space borne sensor) over the Okavango Delta, Botswana, in Africa. It contains 14 classes.

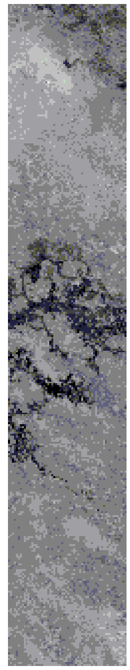

Figure 4.9: Botswana

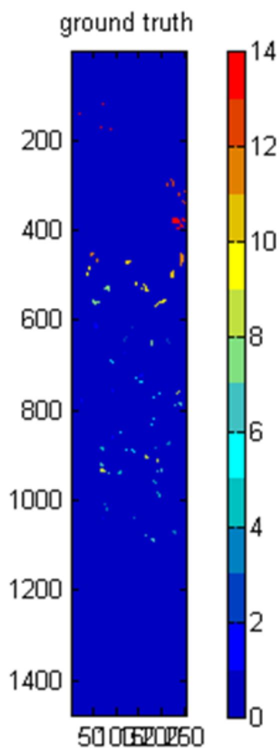

Figure 4.10: Ground Truth of Botswana

4.6 Kennedy Space Center (KSC)

This image has been taken from the AVIRIS sensor over the Kennedy Space Center (KSC), Florida, in the United States of America. It was acquired from approx. 20 km with a resolution of 18 m. It contains 13 classes.

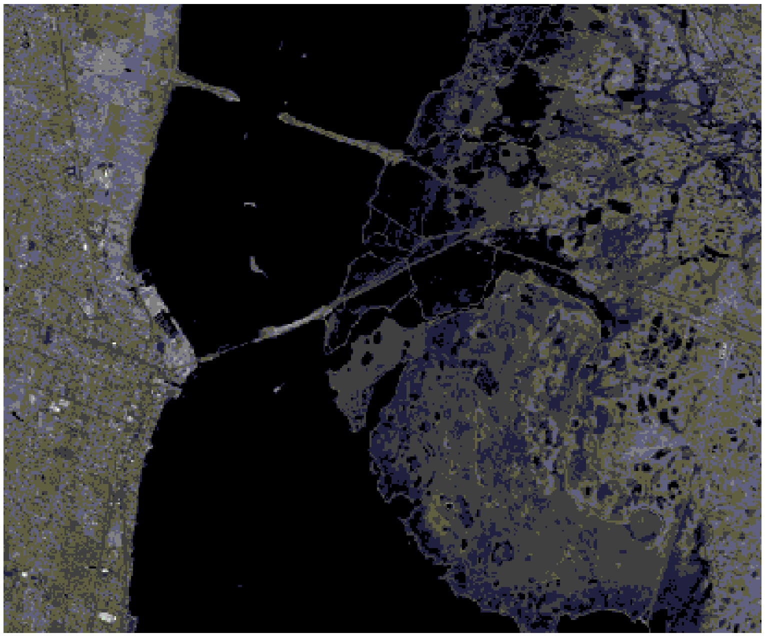

Figure 4.11: Kennedy Space Center (KSC)

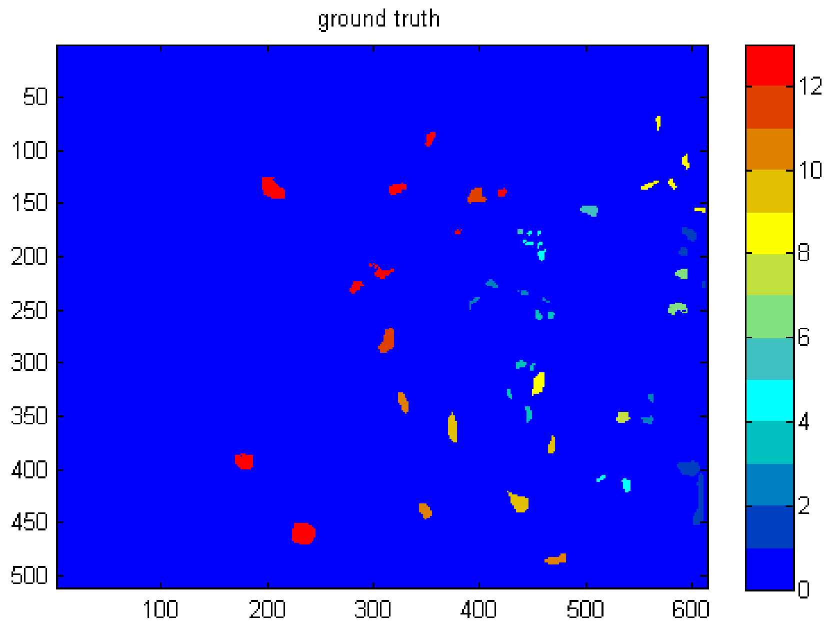

Figure 4.12: Ground Truth of Kennedy Space Center (KSC)

A summary of salient features of each dataset is given in Table (1). The reference [15] provides details about the types of classes and their number of pixels in each of the six databases used.

Table 4.5: A summary of salient features of each dataset [15]

Data Set/Algorithm	**Region**	**Sensor**	**Resolution (m)**	**Bands (Total Bad Used)**	**Wavelength (nm)**	**# of Classes**	**Pixels**	**Remarks**
PaviaU	Pavia University, Northern Italy	Reflective Optics Spectrographic Imaging System (ROSIS)	1.3	103 0 103	NA	9	610 x 340	
Indian Pines	Indian Pines, North-Western Indiana, United States of America	NASA AVIRIS	NA	224 24 200	400-2500	16	145 x 145	2/3 agricultural, 1/3 forest or vegetation, June so 50% crops
Salinas	Salinas Valley,	NASA AVIRIS	3.7	224 20	NA	16	512 x	Vegetables, bare soil

	California, United States of America			204			217	and vineyard fields
SalinasA	Salinas Valley, California, United States of America	NASA AVIRIS	3.7	224 20 204	NA	6	83 x 86	Vegetables
Botswana	Okavango Delta, Botswana, Africa	Hyperion NASA E-O1 Satellite	30	247 97 145	400-2500	14	1476 x 256	Seasonal swamps, occasional swamps, and drier woodlands
Kennedy Space Center	Kennedy Space Center, Florida, United States of America	NASA AVIRIS	18	224 48 176	400-2500	13	512 x 614	20km altitude, color interfaced photography, and landsetthermatic mapper (TM) imagery

4.7 Discussion on Results

4.7.1 Pavia University Dataset

The Figure 4.13 till 4.17 represent results for Pavia University dataset. Their captions are self-explanatory. We see that stage 1 and stage 2 together give good material mapping matching ground truth images well.

It shows that how the image has first received the clustered results. The next image shows that random samples have been taken from the ground truth for training and testing purposes. 2 stages have been assigned and hence the final result shows how both the stages shows different accuracy. The images also shows that at stage 2, the target has been recognized with a better accuracy. The results can been seen in table 4.6.Even from the simulated results, it can been seen the accuracy which has been obtained has increased.

4.7.2 Indian Pines Dataset

The Figure 4.18 till 4.22 represent results for Indian Pines dataset. Their captions are self-explanatory. We see that stage 1 and stage 2 together give good material mapping matching ground truth images well.

It shows that how the image has first received the clustered results. The next image shows that random samples have been taken from the ground truth for training and testing purposes. 2 stages have been assigned and hence the final result shows how both the stages shows different accuracy. The images also shows that at stage 2, the target has been recognized with a better accuracy. The results can been seen in table 4.6.Even from the simulated results, it can been seen the accuracy which has been obtained has increased.

4.7.3 Salinas Dataset

The Figure 4.23 till 4.27 represent results for Salinas dataset. Their captions are self-explanatory. We see that stage 1 and stage 2 together give good material mapping matching ground truth images well.

It shows that how the image has first received the clustered results. The next image shows that random samples have been taken from the ground truth for training and testing purposes. 2 stages have been assigned and hence the final result shows how both the stages shows different accuracy. The images also shows that at stage 2, the target has been recognized with a better accuracy. The results can been seen in table 4.6.Even from the simulated results, it can been seen the accuracy which has been obtained has increased.

4.7.4 Salinas-A Dataset

The Figure 4.28 till 4.32 represent results for Salinas-A dataset. Their captions are self-explanatory. We see that stage 1 and stage 2 together give good material mapping matching ground truth images well.

It shows that how the image has first received the clustered results. The next image shows that random samples have been taken from the ground truth for training and testing purposes. 2 stages have been assigned and hence the final result shows how both the stages shows different accuracy. The images also shows that at stage 2, the target has been recognized with a better accuracy. The results can been seen in table 4.6.Even from the simulated results, it can been seen the accuracy which has been obtained has increased.

4.7.5 Botswana Dataset

The Figure 4.38 till 4.42 represent results for Botswana dataset. Their captions are self-explanatory. We see that stage 1 and stage 2 together give good material mapping matching ground truth images well.

It shows that how the image has first received the clustered results. The next image shows that random samples have been taken from the ground truth for training and testing purposes. 2 stages have been assigned and hence the final result shows how both the stages shows different accuracy. The images also shows that at stage 2, the target has been recognized with a better accuracy. The results can been seen in table 4.6.Even from the simulated results, it can been seen the accuracy which has been obtained has increased.

4.7.6 Kennedy Space Center (KSC) Dataset

The Figure 4.11 till 4.12 represent results for KSC dataset. Their captions are self-explanatory. We see that stage 1 and stage 2 together give good material mapping matching ground truth images well.

It shows that how the image has first received the clustered results. The next image shows that random samples have been taken from the ground truth for training and testing purposes. 2 stages have been assigned and hence the final result shows how both the stages shows different accuracy. The images also shows that at stage 2, the target has been recognized with a better accuracy. The results can been seen in table 4.6.Even from the simulated results, it can been seen the accuracy which has been obtained has increased.

4.8 Result Data

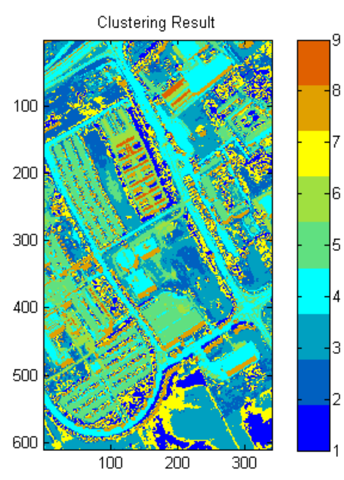

Figure 4.13: Clustering Results of Pavia University Data Set

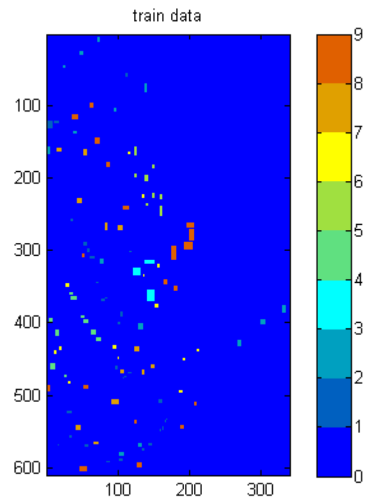

Figure 4.14: Training samples chosen randomly from the ground truth

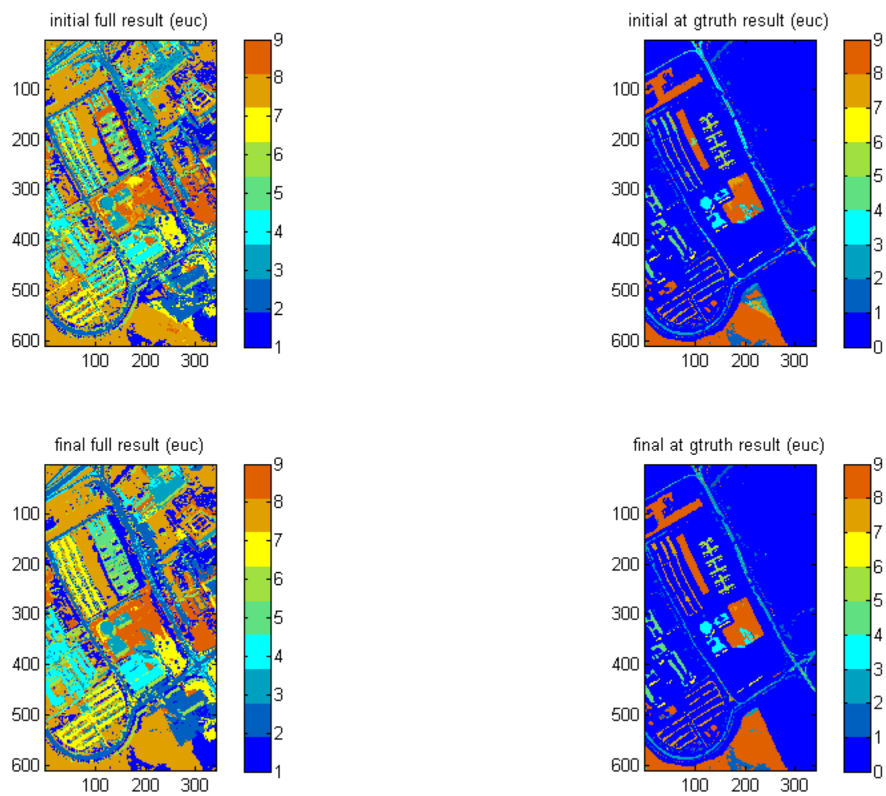

Figure 4.15: Euclidean results: top-left stage 1 classification, top right stage 1 results at ground truth pixels. Bottom-left stage 2 classification, bottom right stage 2 results at ground truth pixels.

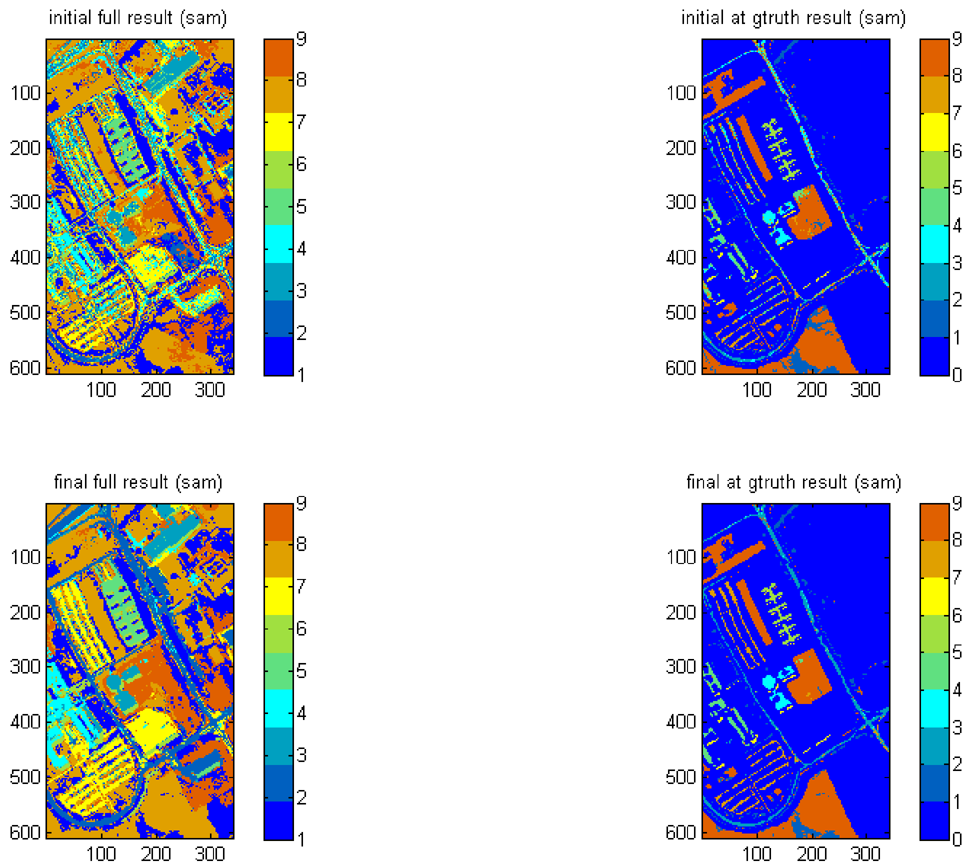

Figure 4.16: SAM results: top-left stage 1 classification, top right stage 1 results at ground truth pixels. Bottom-left stage 2 classification, bottom right stage 2 results at ground truth pixels.

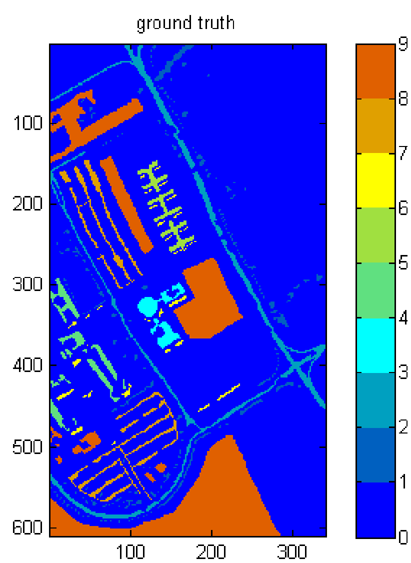

Figure 4.17: Ground truth pixels

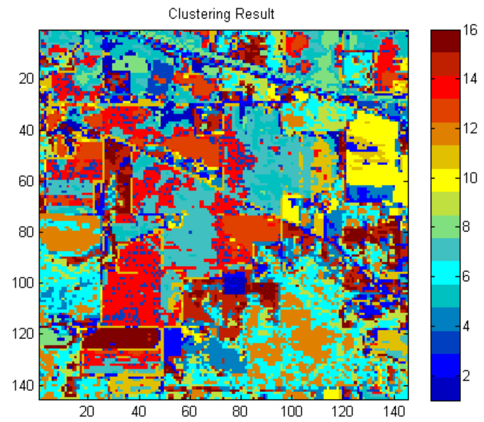

Figure 4.18: Clustering Result of Indian Pines

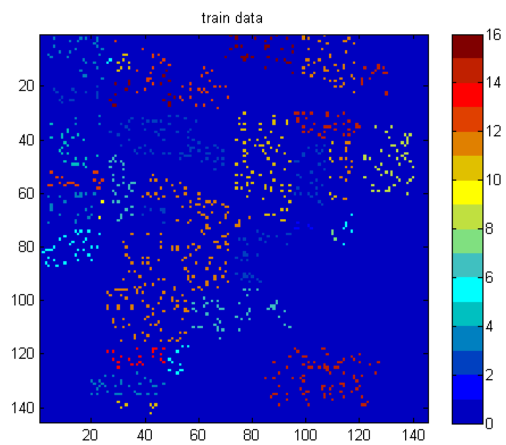

Figure 4.19: Training pixels

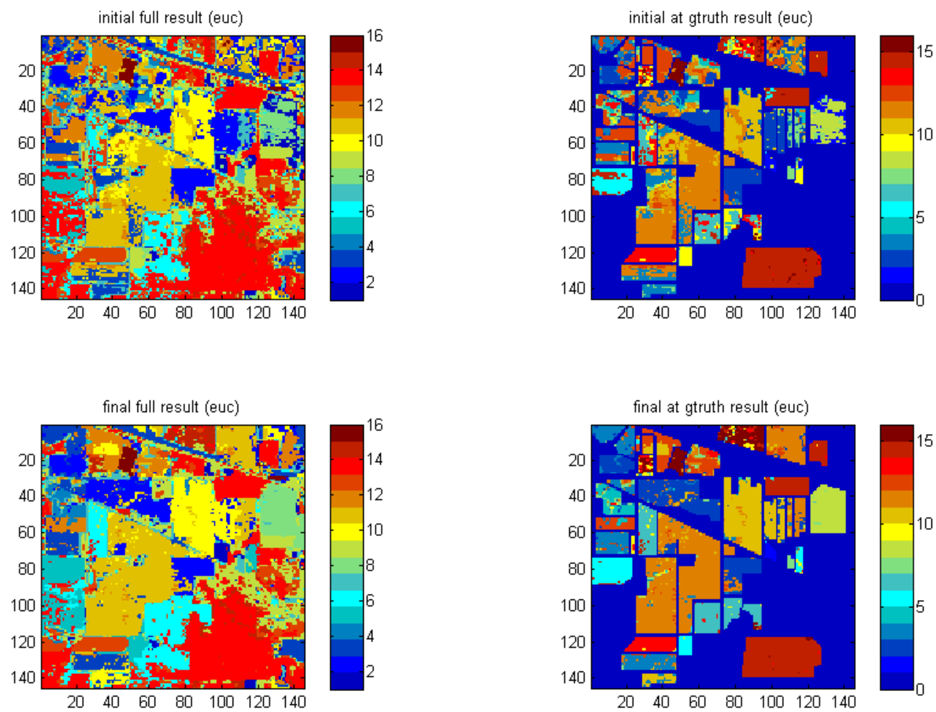

Figure 4.20: Euclidean results: top-left stage 1 classification, top right stage 1 results at ground truth pixels. Bottom-left stage 2 classification, bottom right stage 2 results at ground truth pixels.

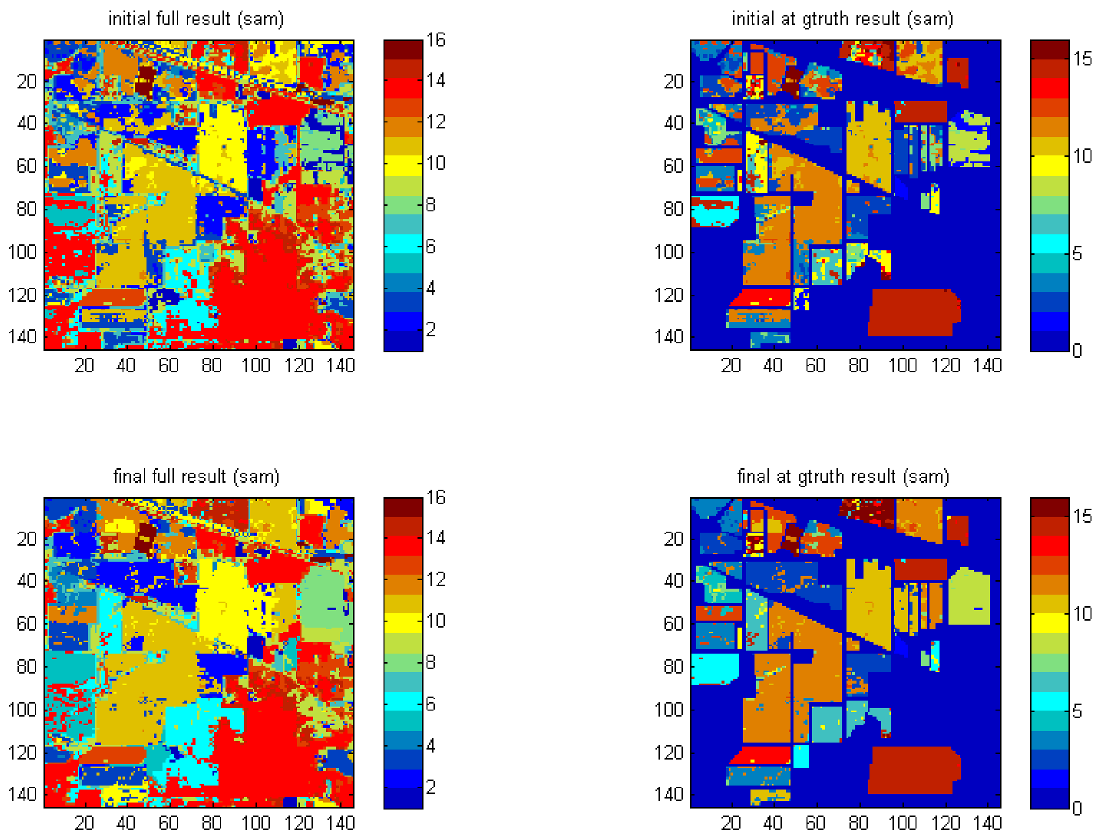

Figure 4.21: SAM results: top-left stage 1 classification, top right stage 1 results at ground truth pixels. Bottom-left stage 2 classification, bottom right stage 2 results at ground truth pixels.

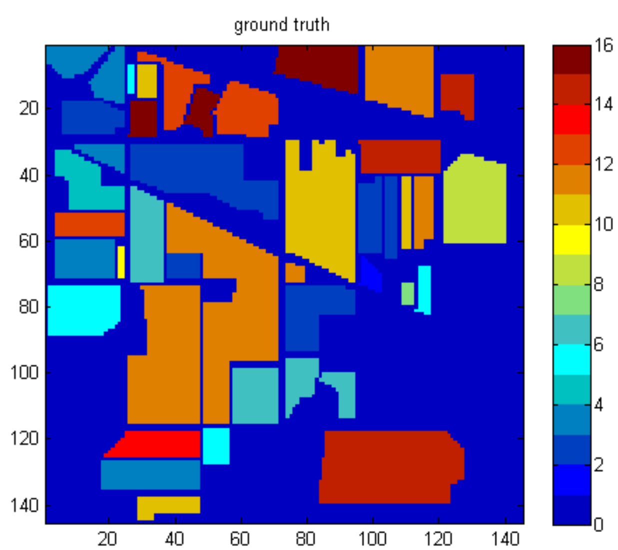

Figure 4.22: Ground truth pixels

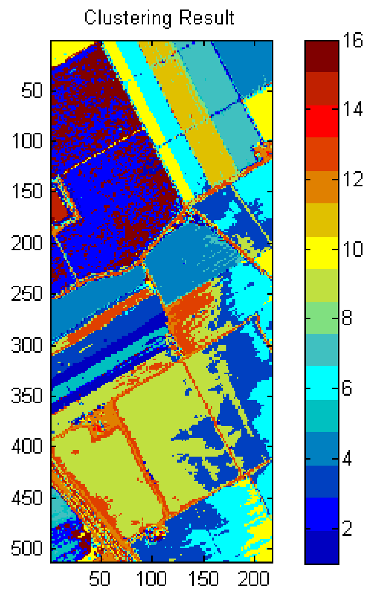

Figure 4.23: Clustering Result of Salinas

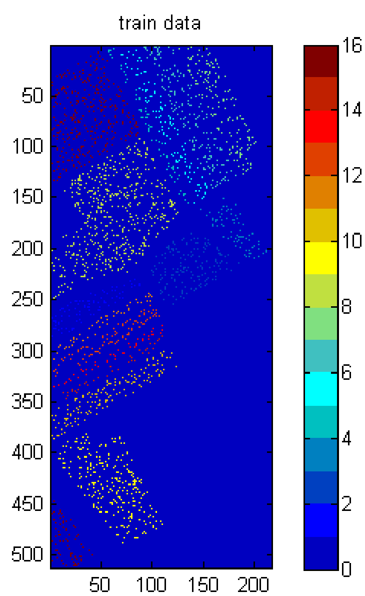

Figure 4.24: Training pixels

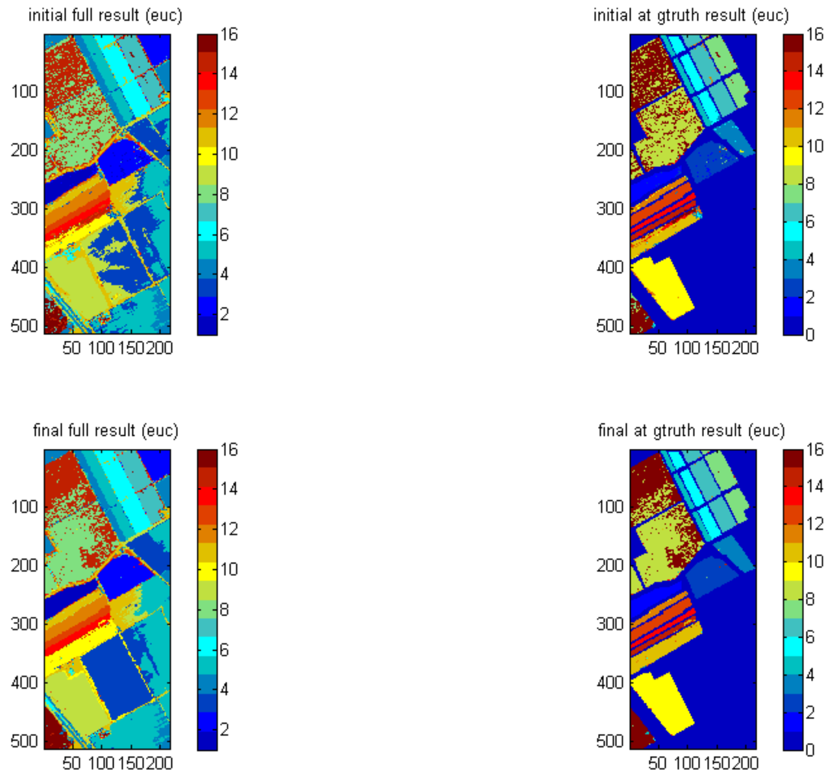

Figure 4.25: Euclidean results: top-left stage 1 classification, top right stage 1 results at ground truth pixels. Bottom-left stage 2 classification, bottom right stage 2 results at ground truth pixels

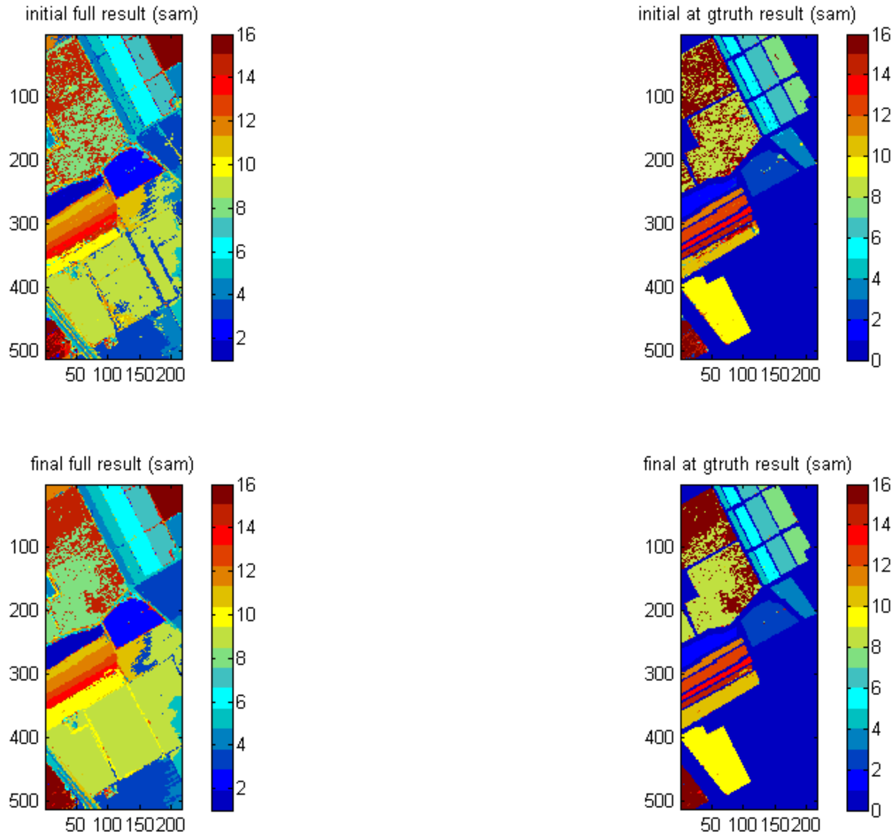

Figure 4.26: SAM results: top-left stage 1 classification, top right stage 1 results at ground truth pixels. Bottom-left stage 2 classification, bottom right stage 2 results at ground truth pixels.

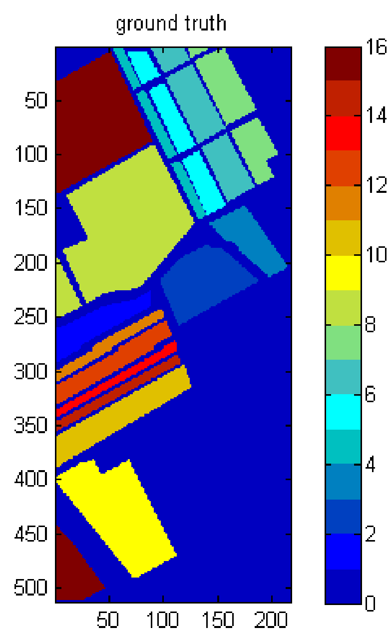

Figure 4.27: Ground truth pixels

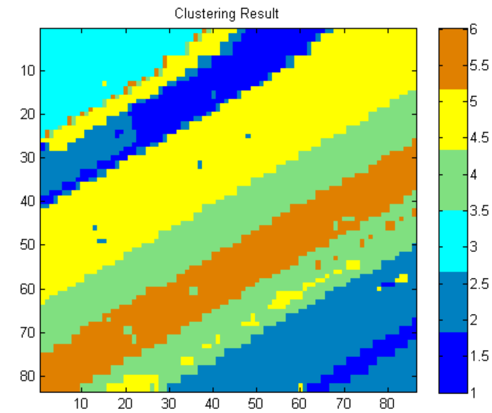

Figure 4.28: Clustering Result of Salinas-A

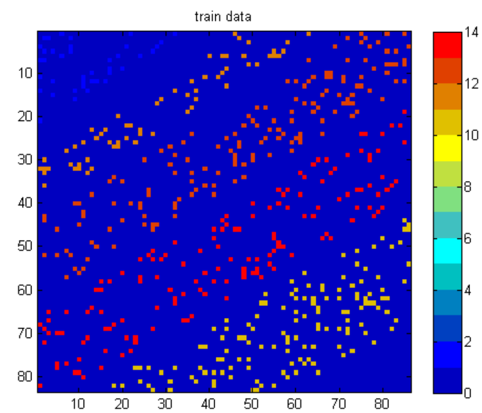

Figure 4.29: Training pixels

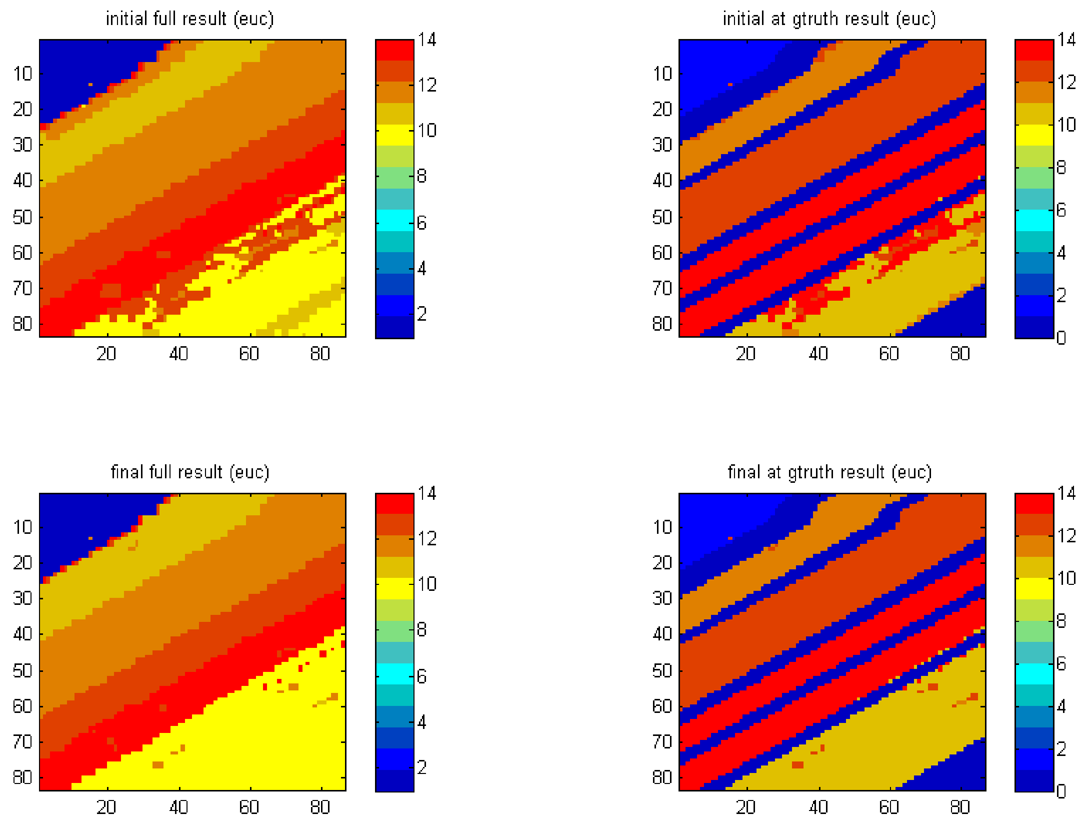

Figure 4.30: Euclidean results: top-left stage 1 classification, top right stage 1 results at ground truth pixels. Bottom-left stage 2 classification, bottom right stage 2 results at ground truth pixels.

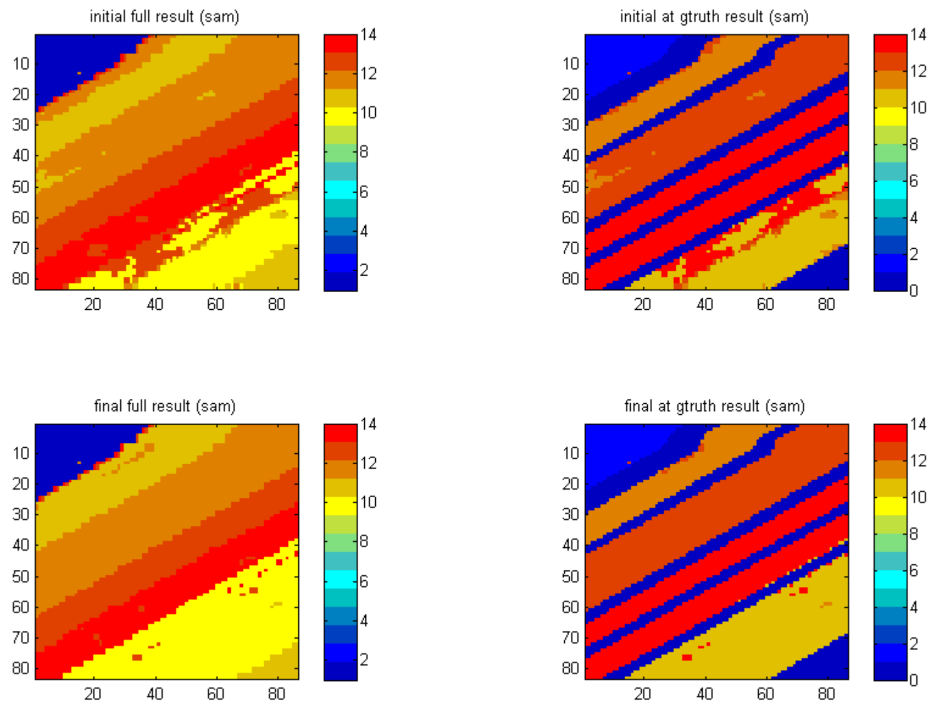

Figure 4.31: SAM results: top-left stage 1 classification, top right stage 1 results at ground truth pixels. Bottom-left stage 2 classification, bottom right stage 2 results at ground truth pixels.

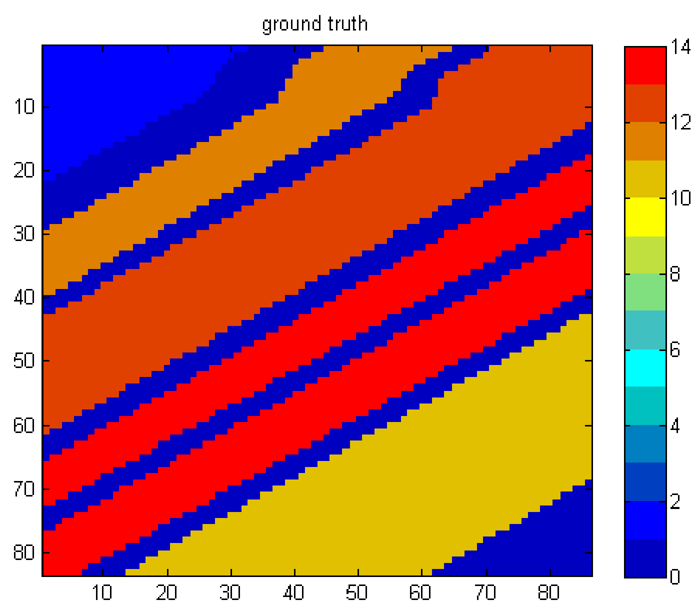

Figure 4.32: Ground truth pixels

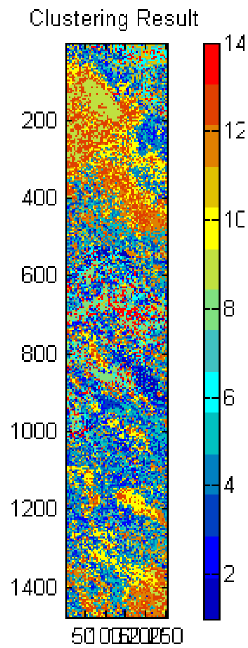

Figure 4.33: Clustering Result of Botswana

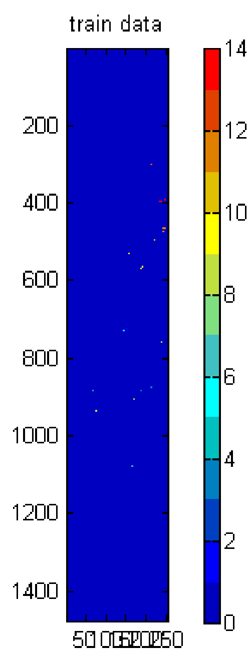

Figure 4.34: Training pixels

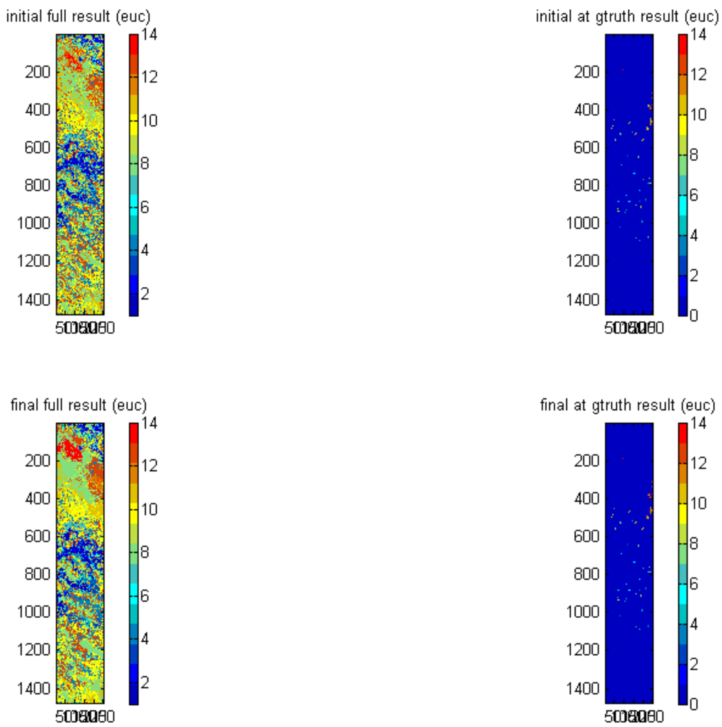

Figure 4.35: Euclidean results: top-left stage 1 classification, top right stage 1 results at ground truth pixels. Bottom-left stage 2 classification, bottom right stage 2 results at ground truth pixels.

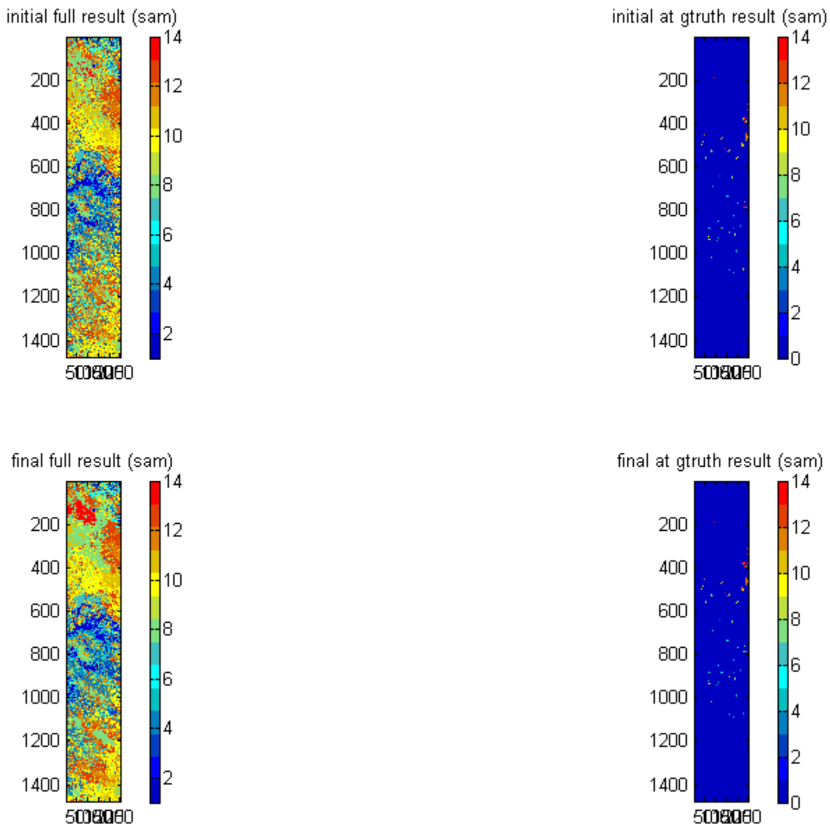

Figure 4.36: SAM results: top-left stage 1 classification, top right stage 1 results at ground truth pixels. Bottom-left stage 2 classification, bottom right stage 2 results at ground truth pixels.

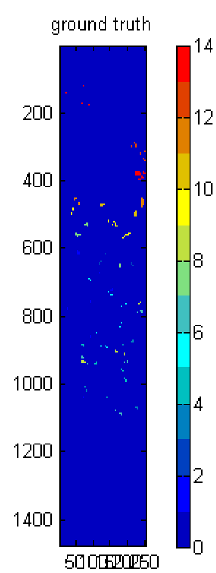

Figure 3.37: Ground truth pixels

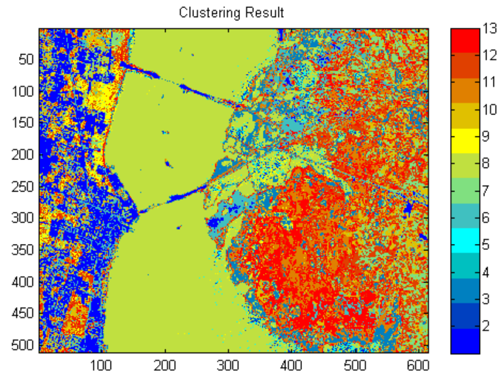

Figure 4.38: Clustering Result of KSC

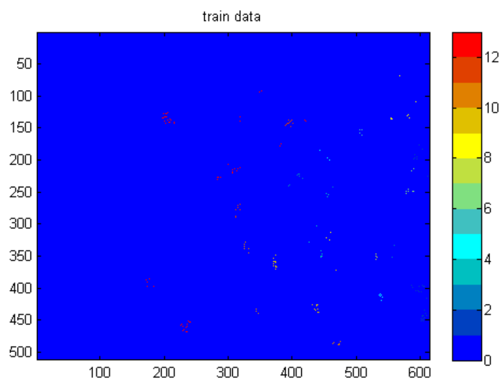

Figure 4.39: Training pixels

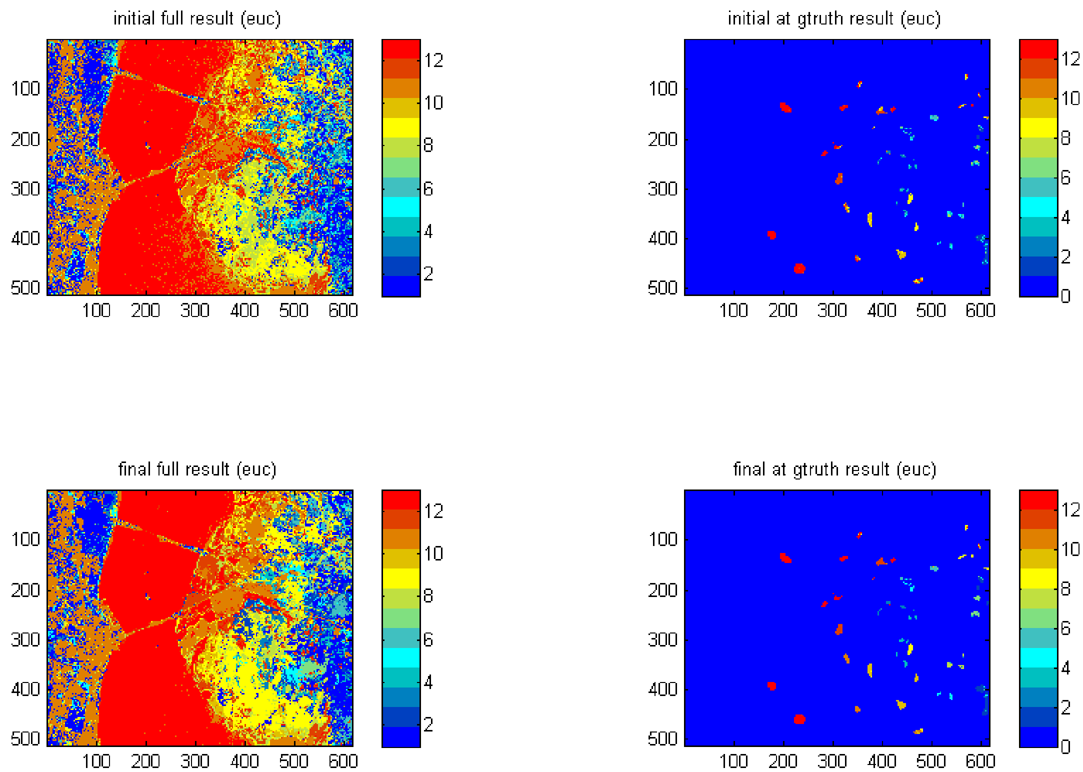

Figure 4.40: Euclidean results: top-left stage 1 classification, top right stage 1 results at ground truth pixels. Bottom-left stage 2 classification, bottom right stage 2 results at ground truth pixels.

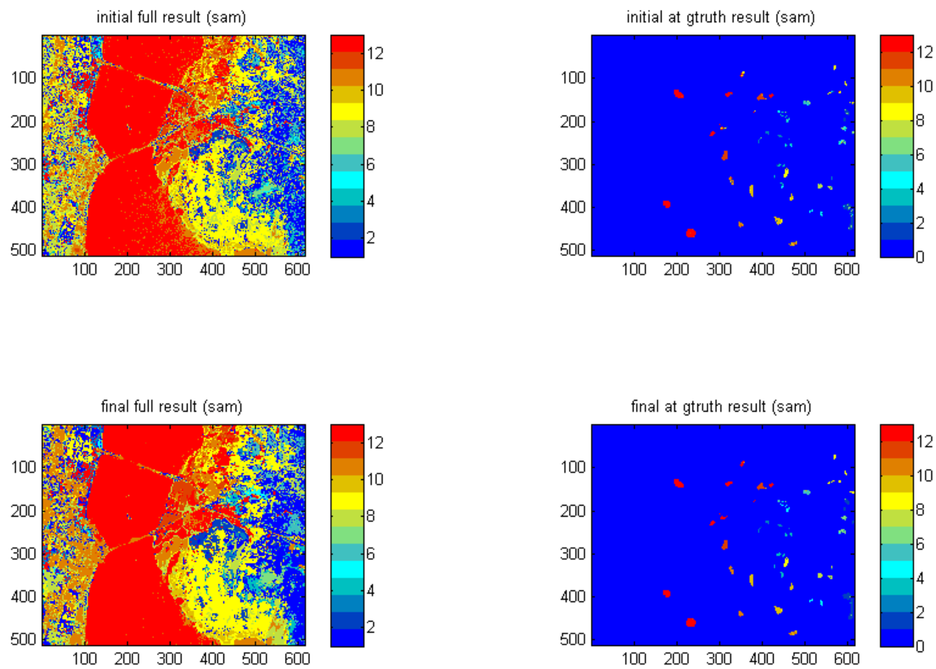

Figure 4.41: SAM results: top-left stage 1 classification, top right stage 1 results at ground truth pixels. Bottom-left stage 2 classification, bottom right stage 2 results at ground truth pixels.

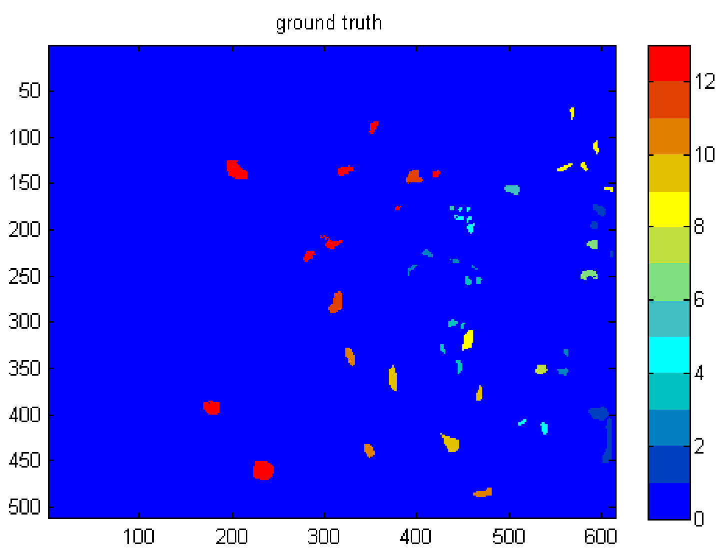

Figure 4.42: Ground truth pixels

Table 4.6: Results

Datasets/Algorithms	Stage 1 (OA)		Stage 2 (OA)		Stage 2 (AA)	
	Euclidean	SAM	Euclidean	SAM	Euclidean	SAM
PaviaU	73.39	65.51	91.38	91.25	93.4	93.37
Indian Pines	61.02	65.45	87.26	88.52	87.62	88.84
Salinas	87.85	86.02	92.56	92.61	97.15	97.37
Salinas-A	91.00	89.83	98.73	98.60	98.84	98.56
Botswana	79.68	81.65	91.47	92.76	92.36	93.79
Kennedy Space Center	68.11	76.21	93.36	94.52	91.54	92.29

The results have been tabulated in Table (4.6). Dynamic Training Data Condensation for Euclidean and SAM algorithm (stage 1) results are in column 2 and 3, respectively. The results are provided in the form of Overall Accuracy (OA) and Average Accuracy (AA). We have used 10% of the available ground truth for training for each class for all the databases. The accuracies are averaged over 10 random runs of training pixels selection. The Kmeans algorithm was run 10 times for each training set and the OA and AA were averaged for these runs. Number of near pixels was set to 20. We achieved efficiency as well as accuracy by using every 10^{th} band for Kmeans clustering. The number of clusters was set to be equal to the number of classes. The number of Kmeans iterations was set to 100. In most of the cases Kmeans converged in 100 iterations.

We can see that in Stage 2, both accuracies are impressive. The results show that both Euclidean and SAM algorithms perform equally well. SAM is robust to multiplicative distortions. This is demonstrated in stage 1 results for Kennedy Space Center and Indian Pines dataset where SAM outperforms Euclidean. On the other hand, SAM is inferior to Euclidean for Pavia University dataset.

The accuracies are comparable to those obtained by state of the art which predominantly uses SVM and exhaustive post-processing stages. Most of the existing algorithms eliminate bad data in data sets and show their results on a limited portion. They either simulate less number of classes, or less number of bands. Moreover, they simulate one or two datasets while we have shown performance over full 6 datasets. Note that our simulations find some errors in the labeling of the ground truth pixels as well. Therefore, accuracy rates would increase if those errors are not present.

An interesting comparison can be made with [16]. The thesis in [16] talks about reducing the training set for kNN matching, resembling our approach. Though, it simulates only two datasets, i.e., the Pavia University and the Indian Pines dataset. The authors in [16] use 80% as training and 20% as testing pixels for each class. They achieve accuracies of 89% and 84% for the two datasets, respectively, versus ours accuracies of 91% and 88%. Note that we are achieving this by using a much reduced training set i.e., 10% for each class is retained as a training set. Moreover, we use only 20 pixels for each class for matching.

The authors in [17] use 3D DWT features and structured sparse classifiers in place of SVMs. They simulate four datasets i.e., Indian Pines, KSC, Botswana and Pavia University. It is the only reference we have found, which uses full datasets in their experiments. Note that we have used 6 datasets. Their best accuracies using SVM-rbf for Indian Pines dataset are 82% (70% AA) and 95% (88% AA) using simple spectral and 3D-DWT features, respectively. Our accuracy is 89% (88% AA) which is better than theirs with SVM-rbf + spectral features in both OA and AA sense. Our accuracy is smaller than theirs with SVM-rbf + 3D-DWT features. But the SVM-rbf + 3D-DWT feature is not a winner over all datasets. For example for KSC dataset with 25% training, SVM-rbf + 3D-DWT features achieves 94% (93% AA) while we achieve 95% (92% AA) with only 10% training. For Botswana dataset using 25% training, their best accuracies using SVM-rbf are 94% (94% AA) and 99% (99% AA) using simple spectral and 3D-DWT features, respectively. Ours is 93% (94% AA) with using only 10% training data. For PaviaU dataset using 10% training, their best accuracies using SVM-rbf are 79% (79% AA) and 95% (94% AA) using simple spectral and 3D-DWT features, respectively. Ours is 92% with using 10% training data. They do not simulate two additional benchmark datasets i.e., Salinas and SalinasA. We have simulated Salinas and SalinasA datasets and have achieved accuracies of 93% (97% AA) and 99% (98% AA), respectively.

CHAPTER 5: CONCLUSIONS AND FUTURE WORK

We have proposed a novel material-mapping algorithm, which relies on the fact that pixels belonging to the same class but located at different positions in the image exhibit variability in their spectral signatures. This could be due to the difference in terrain, atmosphere and surrounding materials. Therefore, a pixel will better match to the neighboring rather than distant pixels of its own class.

Our algorithm dynamically reduces the training set for each testing pixel. Median matching score of 20 spatially closest members of each class are compared to decide the fate of the testing pixel.

Two matching algorithms namely Euclidean distance and Spectral Angle Mapper (SAM) are used. We know that SAM algorithm is robust to multiplicative distortion between test and reference spectra.

Our approach is different to, for example, using a Support Vector Machine (SVM). It resembles more to Nearest Neighbor (NN) algorithm. It complexity is lower than that of NN owing to matching being performed with only limited number of training pixels. In case of SVM, learning takes long especially for long feature vectors. It is generally easier to deal with multiple-class problems with NN than SVM. Several parameters need to be tuned to get good accuracy and generalization from SVM.

We use unsupervised learning to help supervised learning by Euclidean and SAM classifiers. The basic idea is that all pixels belonging to a cluster should be classified to the same class. It greatly increases the accuracy of the first stage of our approach. The training data is utilized again by clustering. If a training pixel is present within a cluster, the whole cluster is classified as belonging to the training pixel class. Our results show that 2nd stage results into comparable accuracy of Euclidean and SAM algorithms. We perform clustering and material classification for the whole images in the datasets. The accuracy, though, is judged only on the ground truth pixels.

SAM algorithm generally outperform Euclidean in stage 1 results for most of the datasets but is inferior in case Pavia University dataset. SAM robustness to multiplicative noise should be used with caution as it may result into misclassifications, too.

Most of the existing algorithms eliminate bad data in data sets and show their results on a limited portion. They either simulate less number of classes, or less number of bands. Moreover, they simulate one or two datasets while we have shown performance over full 6 datasets.

In most of the cases by target recognition, we mean a target material recognition [14]. Other spatial target may be identified by the cluster analysis of pixels belonging to their material. In some instances, the material of their background may also help, e.g., a bridge is defined as concrete over water.

Our first stage uses hard classification of pixels. In the second stage, we have used Kmeans which provides hard clustering. Fuzzy C Mean (FCM) algorithm provides soft clustering and unmixing techniques provide fuzzy membership to each testing pixel. An interesting dimension would be to use FCM along with unmixing techniques to classify the pixels.

ANNEXURE-A

Table 1.1: Airborne Sensors

Airborne Sensors	Manufacturer	Number of Bands	Spectral Range
AVRIS (Airborne Visible Infrared maging Spectrometer)	NASA Jet Propulsion Lab	224	0.4 to 2.5 µm
HYDICE (Hyperspectral Digital Imagery Collection Experiment)	Naval Research Lab	210	0.4 to 2.5 µm
PROBE-1	Earth Search Sciences Inc.	128	0.4 to 2.5 µm
Casi (Compact Airborne Spectographic Imager)	ITRES Research Limited	Up to 228	04 to 1.0 µm
HyMap	Integrated Spectronics	100 to 200	Visible to Thermal Infrared
EPS-H (Experimental Protection System)	GER Corporation	VIS/NIR (76), SWIR1 (32), SWIR2 (32), TIR (12)	VIS/NIR (0.43 to 1.05 µm), SWIR1 (1.5 to 1.8 µm), SWIR2 (2.0 to 2.5 µm), TIR (8 to 1.25µm)
DAIS 7915 (Digital Airborne Imaging Spectrometer)	GER Corporation	VIS/NIR (32), SWIR1 (8), SWIR2 (32), MIR (1), TIR (6)	VIS/NIR (0.43 to 1.05 µm), SWIR1 (1.5 to 1.8 µm), SWIR2 (2.0 to 2.5 µm), MIR (3.0 to 5.0 µm), TIR (8.7 to 1.23 µm)
DAIS 7915 (Digital	GER Corporation	VIS/NIR (76),	VIS/NIR (0.40 to

Airborne Imaging Spectrometer)		SWIR1 (64), SWIR2 (64), MIR (1), TIR (6)	1.0 µm), SWIR1 (1.0 to 1.8 µm), SWIR2 (2.0 to 2.5 µm), MIR (3.0 to 5.0 µm), TIR (8..0 to 12.0 µm)
AISA (Airborne Imaging Spectrometer)	Spectral Imaging	Up to 288	0.43 to 1.0µm

Table 1.2: Satellite Sensors

Airborne Sensors	Manufacturer	Number of Bands	Spectral Range
FTHSI on MightySat II	Air Force Research Lab	256	0.35 to 1.05 µm
Hyperion on EO-1	NASA Goddard Space Flight Center	220	0.4 to 2.5 µm

Table 4.1: Ground Truth Classes and their number of samples

#	Class	Samples
1	Asphalt	6631
2	Meadows	18649
3	Gravel	2099
4	Trees	3064
5	Painted metal sheets	1345
6	Bare Soil	5029
7	Bitumen	1330
8	Self-Blocking Bricks	3682

9	Shadows	947

Table 4.2: Ground Truth Classes and their number of samples

#	Class	Samples
1	Alfalfa	46
2	Corn-not ill	1428
3	Corn-mint ill	830
4	Corn	237
5	Grass-pasture	483
6	Grass-trees	730
7	Grass-pasture-mowed	28
8	Hay-windrowed	478
9	Oats	20
10	Soybean-not ill	972
11	Soybean-mint ill	2455
12	Soybean-clean	593
13	Wheat	205
14	Woods	1265
15	Buildings-Grass-Trees-Drives	386
16	Stone-Steel-Towers	93

Table 4.3: Ground Truth Classes and their number of samples

#	Class	Samples
1	Brocoli green weeds 1	2009
2	Brocoli green weeds 2	3726
3	Fallow	1976
4	Fallow rough plow	1394
5	Fallow smooth	2678

6	Stubble	3959
7	Celery	3579
8	Grapes untrained	11271
9	Soil vineyard develop	6203
10	Corn senesced green weeds	3278
11	Lettuce romaine 4wk	1068
12	Lettuce romaine 5wk	1927
13	Lettuce_romaine_6wk	916
14	Lettuce_romaine_7wk	1070
15	Vinyard_untrained	7268
16	Vinyard_vertical_trellis	1807

Table 4.4: Ground Truth Classes and their number of samples

#	Class	Samples
1	Brocoli green_weeds_1	391
2	Corn_senesced green_weeds	1343
3	Lettuce romaine_4wk	616
4	Lettuce_romaine 5wk	1525
5	Lettuce romaine_6wk	674
6	Lettuce_romaine 7wk	799

ANNEXURE-B

Codes

Material Mapping:

```
clear all

close all

clc



%%%%%%%%%%%%% Testing on whole image %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


output_gt = load('SalinasA_gt');


load salinasA_data_test


load salinasA_data_train


output = load('SalinasA_corrected.mat');

[rr cc zz] = size(output.salinasA_corrected);

img = output.salinasA_corrected;


% output = load('SalinasA');
%
% [rr cc zz] = size(output.salinasA);
%
% img = output.salinasA;
```

```
no_bands= zz;

no_classes = max(clabels_train);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%



% %%%%%%%%% Testing  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
lim1 = 1;

lim2 = zz;




accuracy1 = 0;

error1 = 0;

accuracy2 = 0;

error2 = 0;

ours_classes_euc = zeros(rr,cc);

ours_classes_sam = zeros(rr,cc);


for jj=1:rr

    for kkk = 1:cc


    pr = squeeze(img(jj,kkk,:));

    n_pr = pr./norm(pr);


    score = zeros(1,no_classes);
    score2 = zeros(1,no_classes);


    for kk=1:no_classes

        [aa dummy2] = find(clabels_train ==kk);
```

```
%          dist_locationss = (repmat(locations_test(jj,:),length(aa),1) -
locations_train(aa,:))';


        dist_locationss = (repmat([kkk,jj],length(aa),1) -
locations_train(aa,:))';

        dist2 = zeros(size(dist_locationss,2),1);

        for mm = 1:size(dist_locationss,2)

            dist2(mm,1) = sqrt(dist_locationss(1,mm).^2 +
dist_locationss(2,mm).^2);

        end

        [dmm idx] = sort(dist2);

        training_data = cdata_train(aa,:);

        near_pixels = min(20,size(training_data,1));

        euc_dist = zeros(near_pixels,1);
        ncc = zeros(near_pixels,1);

        for nn = 1:near_pixels


            dummy = squeeze(img(jj,kkk,:))' -  training_data(idx(nn),:);
            euc_dist(nn,1) = sqrt(sum(dummy.^2))/no_bands;


            pr2 = training_data(idx(nn),:)';
            n_pr2 = pr2./norm(pr2);

            diff_pr = (n_pr - n_pr2);


            sorted_diff = sort(diff_pr.^2);

            ncc(nn,1) = 1 - 0.5*(sum(sorted_diff(lim1:lim2,:)));


        end

%          score(kk) = sum(euc_dist);
%
%          score2(kk) = sum(ncc);
```

```
        score(kk) = median(euc_dist);
        score2(kk) = median(ncc);

    end


    [rr1 cc1] = find(score == min(score));


    [rr2 cc2] = find(score2 == max(score2));

    ours_classes_euc(jj,kkk)=cc1(1);

    ours_classes_sam(jj,kkk)=cc2(1);



    end



end



save sim_whole_image ours_classes_euc ours_classes_sam;




figure

imagesc(ours_classes_euc)

colorbar


figure

imagesc(ours_classes_sam)

colorbar



figure

imagesc(output_gt.salinasA_gt)

colorbar
```

Main Context:

```
clear all;

close all;


clc


load accuracies_b_kmeans

load sim_whole_image.mat



load SalinasA_corrected.mat

[rr cc zz] = size(salinasA_corrected);



% load SalinasA.mat
%
% [rr cc zz] = size(salinasA);




output_gt = load('SalinasA_gt');


load salinasA_data_test

load salinasA_data_train






no_classes = max(clabels_train);

samples_classes_train = zeros(no_classes,1);

for kk=1:no_classes

        [aa dummy2] = find(clabels_train ==kk);
```

```
        samples_classes_train(kk,1) = length(aa);

end

for jj=1:size(cdata_train,1)

locations_train2(jj,1) = (locations_train(jj,1) - 1)*rr +
locations_train(jj,2);

end


samples_classes_test= zeros(no_classes,1);

for kk=1:no_classes

        [aa dummy2] = find(clabels_test ==kk);

        samples_classes_test(kk,1) = length(aa);

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%




Final_map_euc =
ContextualClassification_3(salinasA_corrected,ours_classes_euc,ours_classes_s
am,clabels_train,locations_train2);

Final_map_sam =
ContextualClassification_3(salinasA_corrected,ours_classes_sam,ours_classes_e
uc,clabels_train,locations_train2);


% Final_map_euc =
ContextualClassification_3(salinasA,ours_classes_euc,ours_classes_sam,clabels
_train,locations_train2);
%
% Final_map_sam =
ContextualClassification_3(salinasA,ours_classes_sam,ours_classes_euc,clabels
_train,locations_train2);


save finals Final_map_euc Final_map_sam

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

avg_accuracy_euc = zeros(no_classes,1);
```

```
avg_accuracy_sam = zeros(no_classes,1);



accuracy1 = 0;
error1 = 0;


accuracy2 = 0;
error2 = 0;





salinasA_gt2 = zeros(rr,cc);
salinasA_train = zeros(rr,cc);


salinasA_ours_euc = zeros(rr,cc);
salinasA_ours_sam = zeros(rr,cc);

salinasA_final_euc = zeros(rr,cc);
salinasA_final_sam = zeros(rr,cc);






for jj=1:size(cdata_test,1)


    poss = locations_test(jj,:);


    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


    salinasA_gt2(poss(2),poss(1)) = clabels_test(jj,1);


    salinasA_ours_euc(poss(2),poss(1)) = ours_classes_euc(poss(2),poss(1));

    salinasA_ours_sam(poss(2),poss(1)) = ours_classes_sam(poss(2),poss(1));



    salinasA_final_euc(poss(2),poss(1)) = Final_map_euc(poss(2),poss(1));
```

```
    salinasA_final_sam(poss(2),poss(1)) = Final_map_sam(poss(2),poss(1));

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


    if (Final_map_euc(poss(2),poss(1)) == clabels_test(jj,1))


        accuracy1 = accuracy1 + 1;

        avg_accuracy_euc(clabels_test(jj,1),1) =
avg_accuracy_euc(clabels_test(jj,1),1) + 1;

    else

        error1 = error1 + 1;

    end


    if (Final_map_sam(poss(2),poss(1)) == clabels_test(jj,1))


        accuracy2 = accuracy2 + 1;

        avg_accuracy_sam(clabels_test(jj,1),1) =
avg_accuracy_sam(clabels_test(jj,1),1) + 1;

    else

        error2 = error2 + 1;

    end


end




Accuracy1_a_kmeans = accuracy1/(length(cdata_test))*100

Error1 = error1/(length(cdata_test))*100;


Accuracy2_a_kmeans = accuracy2/(length(cdata_test))*100

Error2 = error2/(length(cdata_test))*100;
```

```
Avg_accuracy_euc =
avg_accuracy_euc(avg_accuracy_euc~=0)./samples_classes_test(samples_classes_t
est~=0);

Avg_accuracy_sam =
avg_accuracy_sam(avg_accuracy_sam~=0)./samples_classes_test(samples_classes_t
est~=0);


Avg_accuracy_all_euc_a_kmeans = mean(Avg_accuracy_euc)*100


Avg_accuracy_all_sam_a_kmeans = mean(Avg_accuracy_sam)*100


[Accuracy1_b_kmeans Accuracy2_b_kmeans Accuracy1_a_kmeans
Avg_accuracy_all_euc_a_kmeans Accuracy2_a_kmeans
Avg_accuracy_all_sam_a_kmeans]



for jj=1:size(cdata_train,1)


    poss = locations_train(jj,:);

    salinasA_train(poss(2),poss(1)) = clabels_train(jj,1);

end





figure

imagesc(salinasA_gt2)

colormap(jet(no_classes));

colorbar

title('ground truth')

axis image



figure
```

```
imagesc(salinasA_train)

colormap(jet(no_classes));

colorbar

title('train data')

axis image

figure

subplot(221)

imagesc(ours_classes_euc)

colormap(jet(no_classes));

colorbar

title('initial full result (euc)')

axis image


subplot(222)

imagesc(salinasA_ours_euc)

colormap(jet(no_classes));

colorbar

title('initial at gtruth result (euc)')

axis image

subplot(223)

imagesc(Final_map_euc)

colormap(jet(no_classes));

colorbar

title('final full result (euc)')
```

```
axis image

subplot(224)
imagesc(salinasA_final_euc)
colormap(jet(no_classes));
colorbar
title('final at gtruth result (euc)')
axis image



figure
subplot(221)
imagesc(ours_classes_sam)
colormap(jet(no_classes));
colorbar
title('initial full result (sam)')
axis image

subplot(222)
imagesc(salinasA_ours_sam)
colormap(jet(no_classes));
colorbar
title('initial at gtruth result (sam)')
axis image

subplot(223)
```

```
imagesc(Final_map_sam)
colormap(jet(no_classes));
colorbar
title('final full result (sam)')
axis image

subplot(224)
imagesc(salinasA_final_sam)
colormap(jet(no_classes));
colorbar
title('final at gtruth result (sam)')
axis image


figure
stem(Avg_accuracy_euc)
hold on
stem(Avg_accuracy_sam,'r')
grid on
legend('Avg_euc','Avg_sam')
```

Data Preparation:

```
clear all

close all

clc

randn('state',0)

rand('twister',5489)

tic

%%%%%%%%%%%%% Testing on whole image %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


output_gt = load('SalinasA_gt');



output = load('SalinasA_corrected');

img = output.salinasA_corrected;


% output = load('SalinasA');
%
% img = output.salinasA;


% figure
%
% imagesc(output_gt.salinasA_gt)
%
% colorbar
%
% axis image
%
%
%
```

```
%
figure

imagesc(img(:,:,50))

colormap(gray)

axis image

axis off




[rr cc] = size(output_gt.salinasA_gt);

[rr2 cc2 zz2] = size(img);


no_classes = max(max(output_gt.salinasA_gt));

dataa = [];

DATA = [];


for jj=1:rr

    for kk = 1:cc

        if (output_gt.salinasA_gt(jj,kk) ~=0)


            dataa2 =[kk jj output_gt.salinasA_gt(jj,kk)];


            dataa = [dataa; dataa2];


            DATA2 = squeeze(img(jj,kk,:))';


            DATA = [DATA; DATA2];


        end

    end
```

```
end

samples_classes_gt = zeros(no_classes,1);
percent_test = 1;
percent_train = 0.1;

locations_test = [];
clabels_test= [];
cdata_test = [];


locations_train = [];
clabels_train = [];
cdata_train = [];


for kk=1:no_classes
        dataa3 = dataa(:,3);
        [aa dummy2] = find(dataa3 ==kk);
        samples_classes_test = round(percent_test*length(aa));
        rand_order = randperm(length(aa));
        dataa4 = dataa(aa,:);
        dataa5 = DATA(aa,:);

        locationss = dataa4(rand_order(1:samples_classes_test),1:2);
        labelss = dataa4(rand_order(1:samples_classes_test),3);
        data_test = dataa5(rand_order(1:samples_classes_test),1:zz2);

        locations_test = [locations_test; locationss];
        clabels_test= [clabels_test; labelss];
```

```
        cdata_test = [cdata_test; data_test];


        samples_classes_train = round(percent_train*samples_classes_test);


        locations_train = [locations_train;
locationss(1:samples_classes_train,:)];

        clabels_train= [clabels_train; labelss(1:samples_classes_train,:)];


        cdata_train = [cdata_train; data_test(1:samples_classes_train,:)];


end

save salinasA_data_test locations_test clabels_test cdata_test


save salinasA_data_train locations_train clabels_train cdata_train


figure

imagesc(output_gt.salinasA_gt)

colorbar



material_mapping_5

post_processing_results_3

Main_contextual_2
```

Context Classification:

```
function Final_map =
ContextualClassification_3(HS,initial_classification,initial_classification2,
clabels_train,locations_train2)
% This function implements the contextual classification algorithm based on
kmeans
% clustering of input hyperspectral data and initial classification map.
%
% Usage: Final_map = ContextualClassification(HS,initial_classification);
% Input: HS-> Hyperspectral image, initial_classification-> Preliminary
% classification map obtained by arbitrary classifier
% Output: Final_map-> Improved Classification map
% Dated: 06-08-2015
% Author: Dr. Zahid Mahmood  zmahmood@gmail.com


[r,c,b]= size(HS);
N = length(unique(initial_classification));

% keyboard
NClasses = round(N);%N*1.5
% figure,imagesc(initial_classification);axis image
% title('Initial Classification map')



paviaVec = reshape(HS,[r*c,b]);

paviaVec = paviaVec(:,1:10:b);
% Kmeans clustering

randn('state',0)
rand('twister',5489)


[IDX] = kmeans(paviaVec, NClasses,
'EmptyAction','singleton','MaxIter',100,'Replicates',1);

% [IDX] = kmeans(paviaVec, NClasses,
'EmptyAction','singleton','MaxIter',20,'Replicates',1,'Start','sample','Dista
nce','correlation');
IDX_2d = reshape(IDX,[r,c]);
figure,imagesc(IDX_2d);axis image;colormap(jet(N));colorbar
title('Clustering Result')


% keyboard

Final_map = initial_classification;

Final_map2 = initial_classification2;
```

```
for i=1:NClasses
    mapi = IDX_2d;
    mapi(mapi ~= i) = 0;
    mapi =logical(mapi);
    mapi = bwlabel(mapi);%%%%%%%%% version change %%%%%%%%%%%
    S = regionprops(mapi,'centroid','PixelIdxList');
    for jj=1:length(S)
        idx = S(jj).PixelIdxList;
        if(length(idx)>=3)


%              keyboard

            locs = [];
            for kk=1:length(idx)
             [locs2 dummy] = find(locations_train2==idx(kk));

             locs = [locs; locs2];

            end

%              keyboard

           if (isempty(locs))

              initial_classes = Final_map(idx);


             class_mode = mode(initial_classes);


             Final_map(idx) = class_mode;



           else

           Final_map(idx) = mode(clabels_train(locs));




            end
        end
    end
end
% figure,imagesc(Final_map);axis image;axis off
% title('Contextual Classification map')
```

Post Processing Results:

```
clear all

close all

clc

output_gt = load('SalinasA_gt');

output = load('SalinasA_corrected.mat');

[rr cc zz] = size(output.salinasA_corrected);

% output = load('SalinasA');
%
% [rr cc zz] = size(output.salinasA);

load sim_whole_image

load salinasA_data_test

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

load salinasA_data_train

no_classes = max(clabels_train);

samples_classes_train = zeros(no_classes,1);
```

```
for kk=1:no_classes

        [aa dummy2] = find(clabels_train ==kk);

        samples_classes_train(kk,1) = length(aa);

end


samples_classes_test= zeros(no_classes,1);

for kk=1:no_classes

        [aa dummy2] = find(clabels_test ==kk);

        samples_classes_test(kk,1) = length(aa);

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

avg_accuracy_euc = zeros(no_classes,1);

avg_accuracy_sam = zeros(no_classes,1);


accuracy1 = 0;
error1 = 0;


accuracy2 = 0;
error2 = 0;

% save sim_whole_image ours_classes_euc ours_classes_sam;

for jj=1:size(cdata_test,1)


    poss = locations_test(jj,:);


    if (ours_classes_euc(poss(2),poss(1)) == clabels_test(jj,1))

        avg_accuracy_euc(clabels_test(jj,1),1) =
avg_accuracy_euc(clabels_test(jj,1),1) + 1;


        accuracy1 = accuracy1 + 1;

    else
```

```
        error1 = error1 + 1;

    end


    if (ours_classes_sam(poss(2),poss(1)) == clabels_test(jj,1))


        accuracy2 = accuracy2 + 1;

        avg_accuracy_sam(clabels_test(jj,1),1) =
avg_accuracy_sam(clabels_test(jj,1),1) + 1;


    else

        error2 = error2 + 1;

    end


end


Accuracy1_b_kmeans = accuracy1/(length(cdata_test))*100

Error1 = error1/(length(cdata_test))*100;


Accuracy2_b_kmeans = accuracy2/(length(cdata_test))*100

Error2 = error2/(length(cdata_test))*100;


Avg_accuracy_euc_b_kmeans =
mean(avg_accuracy_euc(avg_accuracy_euc~=0)./samples_classes_test(samples_clas
ses_test~=0));

Avg_accuracy_sam_b_kmeans =
mean(avg_accuracy_sam(avg_accuracy_sam~=0)./samples_classes_test(samples_clas
ses_test~=0));


salinasA_gt2 = zeros(rr,cc);
salinasA_ours_euc = zeros(rr,cc);
salinasA_ours_sam = zeros(rr,cc);


for jj=1:size(cdata_test,1)
```

```
    poss = locations_test(jj,:);

    salinasA_gt2(poss(2),poss(1)) = clabels_test(jj,1);
    salinasA_ours_euc(poss(2),poss(1)) = ours_classes_euc(poss(2),poss(1));
    salinasA_ours_sam(poss(2),poss(1)) = ours_classes_sam(poss(2),poss(1));

end

salinasA_gtt = zeros(rr,cc);

for jj=1:size(cdata_train,1)
    poss = locations_train(jj,:);

    salinasA_gtt(poss(2),poss(1)) = clabels_train(jj,1);

end

save accuracies_b_kmeans Accuracy1_b_kmeans Accuracy2_b_kmeans
Avg_accuracy_euc_b_kmeans Avg_accuracy_sam_b_kmeans

figure
imagesc(ours_classes_euc)
colorbar
axis image

figure
imagesc(ours_classes_sam)
colorbar
axis image

figure
```

```
imagesc(output_gt.salinasA_gt)
colorbar


figure
imagesc(salinasA_gt2)
colorbar
axis image

figure
imagesc(salinasA_gtt)
colorbar
axis image


figure
imagesc(salinasA_ours_euc)
colorbar
axis image



figure
imagesc(salinasA_ours_sam)
colorbar
axis image
```

REFERENCES

1) Goetz, Alexander F.H., and Boardman, J.W. (1997). Atmospheric Corrections: On Deriving Surface Reflectance from Hyperspectral Imagers. In Descour, Michael R. and Shen, S.S. (eds.), Imaging Spectrometry III: Proceedings of SPIE, 3118, 14-22.
2) van der Meer, Freek (1994). Calibration of Airborne Visible/Infrared Imaging Spectrometer Data (AVIRIS) to Reflectance and Mineral Mapping in Hydrothermal Alteration Zones: An Example from the “Cuprite Mining District”. Geocarto International, 3, 23-37.
3) Cloutis, E.A., (1996). Hyperspectral Geological Remote Sensing: Evaluation of Analytical Techniques. International Journal of Remote Sensing, 17, 2215-2242.
4) Mustard, John F., and Sunshine, J.M. (1999). Spectral Analysis for Earth Science: Investigations Using Remote Sensing Data. In Renz, Andrew N. (ed), Remote Sensing for the Earth Sciences: Manual of Remote Sensing (3rd ed.), Vol 3. New York: John Wiley & Sons, pp. 251-306.
5) Saeid Homayouni, Michel Roux, “Hyperspectral Image Analysis For Material Mapping Using Spectral Matching” GET – Télécom Paris – UMR 5141 LTCI- Département TSI
6) Randall B. Smith, 5 January 2012, “ Introduction to Hyperspectral Imagery” MicroImages, Inc.,
7) B. Bue, B. Csathó, E. Merényi, “Automated Labeling of Materials in Hyperspectral Imagery” in IEEE Transactions On Geoscience And Remote Sensing, Vol. 48, No. 11, November 2010.
8) Nasrabadi, N. M. ; Sensors Directorate, US Army Res. Lab., Adelphi, MD, USA,” Hyperspectral Target Detection : An Overview of Current and Future Challenges,” IEEE Signal Processing Magazine 2014.
9) P. Shippert, “Why Use Hyperspectral Imagery?,” Photogrammetric Engineering & Remote Sensing, April 2004.
10) J. Li, P. R. Marpuet. All, “Generalized Composite Kernel Framework for Hyperspectral Image Classification,” IEEE transactions on Geoscience and Remote Sensing, 2013.

11) G. Camps-Valls and L. Bruzzone, “Kernel-based methods for hyperspectral image classification,” IEEE Trans. Geosci. Remote Sens., vol. 43, no. 6, pp. 1351–1362, Jun. 2005.
12) G. Jun and J. Gosh, “SPATIALLY ADAPTIVE SEMI-SUPERVISED LEARNING WITH GAUSSIAN PROCESSES FOR HYPERSPECTRAL DATA ANALYSIS,” University of Michigan, 2011.
13) Y. Tarabalkaet. Al, “Spectral–Spatial Classification of Hyperspectral Imagery Based on Partitional Clustering Techniques,” IEEE Transactions on Geoscience and Remote Sensing, vol. 47, issue 8, pp. 2973-2987, 2009.
14) T. M. Lillesand and R. W. Keifer, “Remote Sensing and Image Interpretation,” John Wiley & Sons, Inc. New York, 2000.
http://www.ehu.eus/ccwintco/index.php?title=Hyperspectral_Remote_Sensing_Scenes,” accessed on August 24, 2015.
15) S. L. J. L. Tinoco and D. Menotti, “An Approach for Reducing the Training Set of the KNN Applied to Remote Sensed Hyperspectral Images Classification,” MS Thesis 2012, Computing Department UFOP - Federal University of Ouro Preto, Ouro Preto, MG, Brazil.
16) Y. Qian, M. Ye, and J. Zhou, “Hyperspectral Image Classification based on Structured Sparse Logistic Regression and 3D Wavelet Texture Features,” IEEE Transactions on Geoscience and Remote Sensing, vol. 51 (4), March 2013.