# AN UNSUPERVISED PARALLELEPIPED MULTISPECTRAL IMAGE CLASSIFIER USING DIFFERENTIAL EVOLUTION

Scott Letkeman, Chih-Cheng Hung, and Bor-Chen Kuo[#]

School of Computing and Software Engineering
Southern Polytechnic State University, Marietta, GA 30060
[#]Mathematics Education Department, National Taichung University, Taichung, Taiwan, R. O. C.

## 1. INTRODUCTION

Multispectral image classification algorithms are useful methods in the interpretation of remotely sensed images. They can be categorized as supervised or unsupervised. With supervised classification algorithms, spectral (class) signatures in the image are developed from training sites defined by the analyst. They require an analyst to select training sites and calculate statistics from these samples. Then, using a decision function based on statistical results, each pixel is assigned to a class. With unsupervised classification algorithms, information about the features of an image are not required.

One of the most popular supervised classification algorithms is called parallelepiped classifier [1, 2]. Unfortunately, its main drawback is that it is supervised. The goal of this paper is to create an unsupervised multispectral parallelepiped classifier by using natural evolution process – differential evolution. With differential evolution algorithm, we modify the approach to make it more stable for finding an exemplar for each class.

The rest of this paper is organized as follows; the parallelepiped classifier is briefly introduced in section 2. Section 3 describes differential evolution. The parallelepiped classifier with DE is presented in section 4. Preliminary experimental results are presented in Section 5. Finally, we summarize our future research plan and conclusions.

## 2. THE PARALLELEPIPED CLASSIFIER

The parallelepiped classifier, a supervised classification algorithm in multispectral image classification, is also called the box classifier. It is a simple logical rule based algorithm. The threshold of each class signature is used to decide whether a given pixel falls within a class. These class signatures must come from analyst defined training sites making this a supervised algorithm. For each band in the multispectral image, a valid range of intensity values is specified by a minimum and maximum value. A pixel is classified as a member of a class if and only if all of its band information or signature fall within the corresponding ranges of the bands defined by that class. Since there may be overlapping areas between classes, a pixel is classified to the first class that its signature falls within that range. It is still preferable though to minimize overlap. Unclassified pixels are normally set to a designated class (e.g. class 0).

The process of the parallelepiped classifier consists of the following steps:

Step1. Determine training data by selecting training sites and assign training set with class number.
Step2. Derive the box or ranges of each class in the training set for each training band.
Step3: Classify the pixels in the image based on the ranges defined in Step2.

## 3. DIFFERENTIAL EVOLUTION

Differential Evolution (DE) is an evolutionary optimization algorithm developed by Kennith Price [3, 4]. The approach is unique from most competing algorithms in that potential solutions are represented exclusively as vectors.

DE works by first initializing a population of random vectors. It then begins to iterate through a number of generations during which the population vectors are regularly mutated and evaluated. Mutations are performed by making a population of trial vectors to compare against the existing population.

$$u_{j,i,g} = \begin{cases} x_{j,r0,g} + F \cdot \left(x_{j,r1,g} - x_{j,r2,g}\right), & \text{if } \left(\text{rand}_j(0,1) \le Cr \text{ or } j = j_{\text{rand}}\right) \\ x_{j,i,g} & \text{otherwise.} \end{cases} \tag{3.1}$$

To produce these trial vectors $u_{j,i,g}$, where j is the position within the vector, i is the vector within the population, and g is the generation, three random population vectors are chosen, $x_{r0}$, $x_{r1}$, and $x_{r2}$. Values are either copied into the trial vector unchanged from the corresponding population vector $x_{j,i,g}$, or if a random number exceeds the crossover likelihood Cr, the scaled difference between $x_{r1}$ and $x_{r2}$ is added to $x_{r0}$. This process is repeated for each value of each trial vector in the population. Next, the fitness values of the population vectors are calculated and compared with those of the previous generation. A trial vector will only make it to the next generation if its fitness value exceeds that of the vector to which it is compared.

The main idea behind DE is that the good attributes will survive from generation to generation, bad attributes disappear and new potentially better attributes will be discovered along the way. All of this is facilitated by a strong fitness function. That is, one that provides a true estimate of a vectors worth in relation to the property being optimized.

## 4. THE PARALLELPIPED CLASSIFIER WITH DE

The two main challenges that arise when applying DE to parallelepiped classification are how to devise a manner in which a potential solution can be represented as a vector, and finding a fitness function that always produces scores representative of a potential solution's worth.

The solution to the first problem is simply to organize the vectors so that the upper and lower bounds of the parallelepiped classifier are placed one after another as shown below.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $S1_{c1,l}$ | $S1_{c1,u}$ | $S1_{c2,l}$ | $S1_{c2,u}$ | $S1_{c3,l}$ | $S1_{c3,u}$ | . . . | $Sn_{c3,u}$ |

Segments range from *S1* to *Sn* , three colors c1 through c3, are assumed, and *l* and *u* represent the upper and lower bounds. This representation leads to vectors with lengths two times the number spectral bands times the number of segments.

To find a suitable fitness function, one must first settle on a definition of optimality. For this application, the definition that was used is a segmentation where within-cluster variance is minimal and between-cluster variance is maximal. To estimate the first of these two numbers the average difference between means was used.

$$a_i = \frac{\sum \| m_{i1} - m_{i2} \|}{n} \tag{4.1}$$

where $m_{i1}$ and $m_{i2}$ are means different from each other, and n is the number of unique combinations. For the second, the average standard deviation was used as the within cluster variance.

$$b_i = \frac{\sum (\sqrt{(\sum (m_{ij} - p_j)^2) / 3)}}{s} = \frac{\sum \sigma_j}{s} \tag{4.2}$$

where $p_j$ is a pixel in the segment *j*, and *s* is the number of segments. The interior sum is divided by three assuming the image will have three spectral bands. These two values are combined into a single value by finding the ratio between the two. $b_i$ has been chosen as the denominator to ensure that larger fitness values indicate better results.

$$f(x_{i,g}) = a_i / b_i - c - d - e \tag{4.3}$$

Above is the final fitness function. There are three penalties in addition to the variance ratio. *c* represents a penalty for segments that are too small. For this experiment it is those having less than a quarter the number of pixels that result from dividing the number of pixels by the number of segments. *d* is a penalty for unclassified pixels. It is the number of classified pixels divided by the number of pixels. Finally, *e* is a penalty for over classified pixels. It is the number of pixels classified into more than one segment divided by the total number of pixels. All three of these penalties are initially values between 0 and 1 but have been weighted by 50 to increase their effect. This weighting has shown to be very important and can be adjusted according to which behaviors one is most interested in discouraging.

## 5. PRELIMINARY EXPERIMENTS

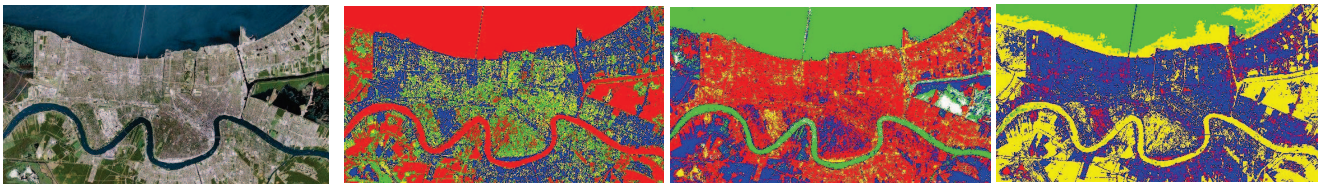
As of yet, testing of the algorithm has not been completed but early results seem promising. The algorithm demonstrates the ability to classify the vast majority of pixels into segments that are largely homogenous within and considerably disparate from each other.

The results, some of which are included in figures 1 and 2, demonstrate that the algorithm works better than the supervised parallelepiped classifier on both simple and complex images. Furthermore, the algorithm can be effective with almost any parameters if one lets it run long enough. Image (c) from figure 1 is better than its supervised counterpart, (b), because it doesn't classify any pixels

into more than one segment. Though not visible, the segments of image (b) overlap a considerable amount. Likewise Image (c) from figure 2 is much superior to Image (b). The segments of image (b) not only overlap but do not represent the distinct features of the original image (water, city, and plants) as well as image (c). Image (d) of figure 2 demonstrates the point that any parameters generate decent results given enough time. A small population size and a small crossover rate slows mutation and limits the development of multiple solutions. A large scaling factor will cause mutations to be more radical and thus less likely to be desirable. Despite the three limitations, 20,000 generations of development still generated a result better than the supervised counterpart.

(a) (b) (c)

**Figure 1**: (a) a simple image of a block toy chosen for its simplicity. (b) is a result from the supervised parallelepiped classification algorithm and (c) is a result from our unsupervised version where Cr = 0.8, Scaling = 0.5 with 40 population member over about 5000 generations.



(a) (b) (c) (d)

**Figure 2**: (a) a satellite picture of New Orleans. (b) a supervised result. (c) an unsupervised result using Cr = 0.8, Scaling = 0.5 with population size 40 over about 5000 generations. (d) an unsupervised result using Cr = 0.3 Scaling = 0.9, population size = 20, over nearly 20,000 generations.

## 6. CONCLUSIONS AND FUTURE WORK

The parallelepiped classifier is a popular and fast classification algorithm and is well accepted in the remote sensing society. However, this algorithm requires supervised training which is usually performed by an experienced analyst. With the help of DE, the supervision usually necessary to train parallelepiped classifiers can be done without supervision.

The robustness of the algorithm can be improved by taking the user provided number of segments as a maximum as opposed to an absolute. Given the unlikeliness that the user actually knows the optimal number of segments, this would be a dramatic improvement. The implementation, in theory, would only require the elimination of the empty segment penalty and the inclusion of the Davies-Bouldin Index in the fitness function. Also, the fitness function may be too general to evaluate the classification results in each evolution. This problem needs be improved to get a better classification result.

## REFERENCES


[1] P.H. Swain and S.M. Davis, "Remote Sensing: the quantitative approach," McGraw-Hill, New York, 1978.

[2] I.L. Thomas, V. M. Benning and N.P. Ching, "Classification of Remotely Sensed Images," Bristol: Adam Hilger, 1987.

[3] Price, Kenneth, Rainer Storn and Jouni Lampien. "Differential evolution: a practical approach to global optimization." Springer, New York 2005.

[4] Das, Swagatam and Ajith Abraham. "Automatic Clustering Using an Improved Differential Evolution Algorithm". IEEE, January 2008, page 218-237.