Terrain Classification for Autonomous RMAX Helicopter Search and Rescue

By

Erin West

Senior Project

ELECTRICAL ENGINEERING DEPARTMENT

California Polytechnic State University

San Luis Obispo

June, 2011

Table of Contents

List of Figures

Figures **Page**

Acknowledgements

Thanks to Lynne Slivovsky for allowing me the opportunity to work on this project and for advising me along the way. I also want to thank the faculty in the EE department for the instruction I have received throughout my years at Cal Poly. Finally, thanks to my family for their love and support.

I. Introduction

Northrop Grumman donated an unmanned RMAX helicopter to Cal Poly San Luis Obispo. Dr. Lynne Slivovsky is currently leading several undergraduate engineering students on projects to enable the helicopter to autonomously perform reconnaissance for the purpose of search and rescue. The helicopter will make use of a laser range finder, infrared cameras, and black and white cameras. Currently the primary focus of this project is researching ways to improve the helicopter's performance and capabilities. My project goal was to write a program that would perform terrain classification from gray scale aerial imagery.

I began by finding a library of computer vision algorithms. I chose to use OpenCV, a strong computer vision library developed by Intel and now supported by Willow Garage that is free to use under a BSD license. The primary steps involved in terrain classification are image segmentation and cluster classification according to terrain type. I decided to use k-means clustering for segmentation and template matching for classification.

II. Background

Segmentation is a process of separating a dataset into groups that naturally belong together. In this application the groups of pixels that are of similar color and texture are grouped. Clustering is common method to segment an image. There are two ways to cluster a dataset, divisive clustering and agglomerative. Divisive clustering begins with one large cluster and the recursively splits clusters until the algorithm yields good clustering. Agglomerative clustering considers each datum a cluster and recursively merges clusters until good clustering is achieved.

K-means clustering is an algorithm that aims to separate n observations into k clusters. First, all tokens are assigned to initial clusters. Then the pixels are compared to the cluster means. If the pixel is closer to another cluster center it is reassigned to that cluster and the means are recalculated. This process is iterated until termination criteria are met. Termination criteria can either be a number of iterations or a maximum within cluster distance. One problem with k-means clustering is the number of clusters, k, must be known in order to run the algorithm. Another is that clustering is affected by noise in the image. I will explain in the Implementation section how I approached these problems.

Once the clusters are formed the program must classify them by terrain type. Template matching is a process that searches an image for small sections that match a template image. Two methods exist for template matching, feature-based and

template-based. The feature-based method matches only the features of the template image to the match image, for example corners or edges. This approach is faster than the template-based method. The template-based method searches the match image using the entire template image. Although this approach is slower, it may yield better results with template images that have weak features than the feature-based method.

III. Implementation

Before clustering with the k-means algorithm I had to reduce the noise and determine the number of clusters dynamically for unknown images. I began to research and test various filters to determine the most appropriate for my application. Figure 1 is a comparison of Gaussian, Median, and Bilateral filters with 5x5 kernels.

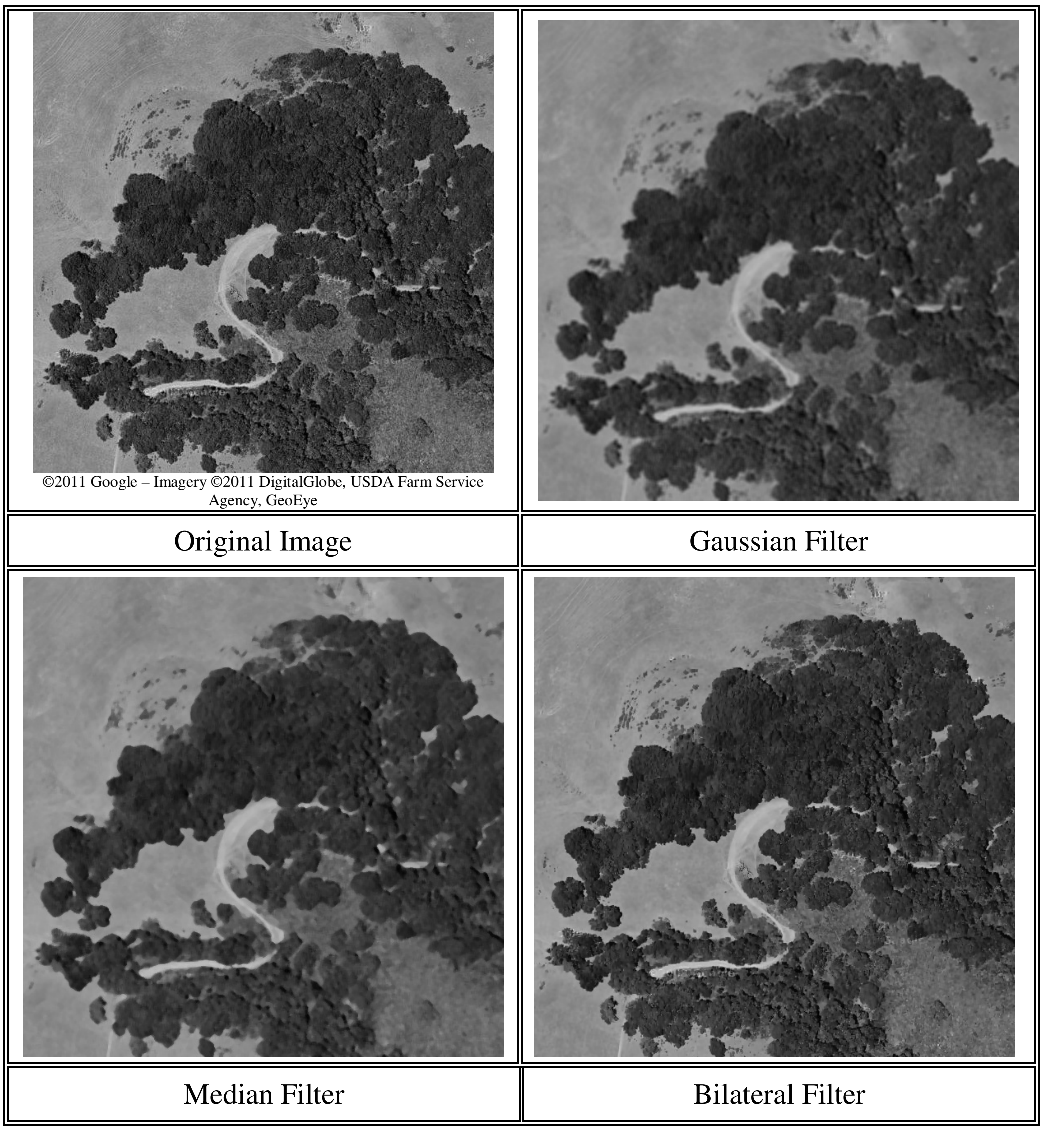


Figure 1: Comparison of Smoothing Filters

Gaussian filters are commonly used for noise reduction. While a Gaussian filter will reduce image noise it will also blur the edges in the image. In order to cluster pixels that had similar color I wanted to smooth while retaining edges present in the image. A couple edge sensitive filters that I tested were the bilateral filter and the median filter. Out of the two edge sensitive filters, the median filter is more aggressive in smoothing the image. However, the bilateral filter incorporates parameters to determine the color space and coordinate space that is affected by the smoothing filter. Furthermore, running the image through two bilateral filters with small kernel sizes yields adequate smoothing and well preserved edges. In the final program I chose to use a Gaussian filter on the image first and then two bilateral filters with small kernel sizes. The result of the filtering method is shown in Figure 2.

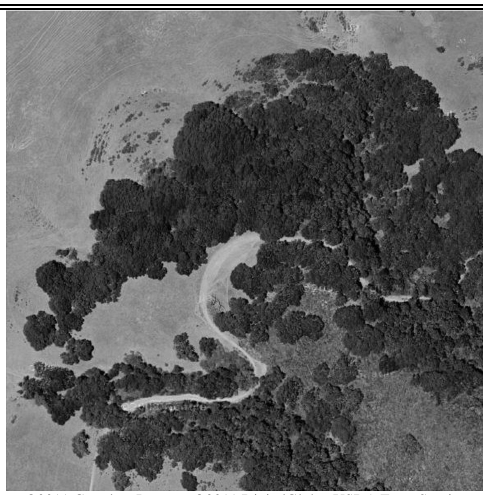
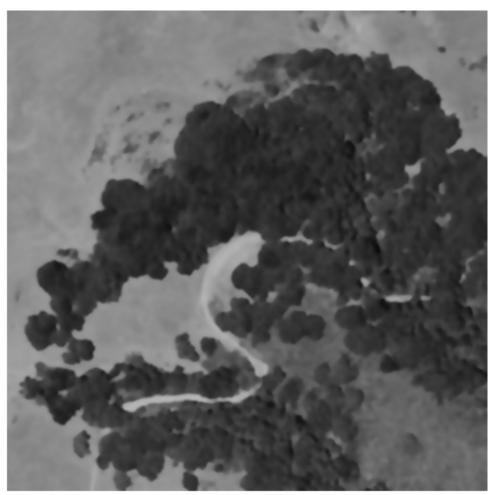
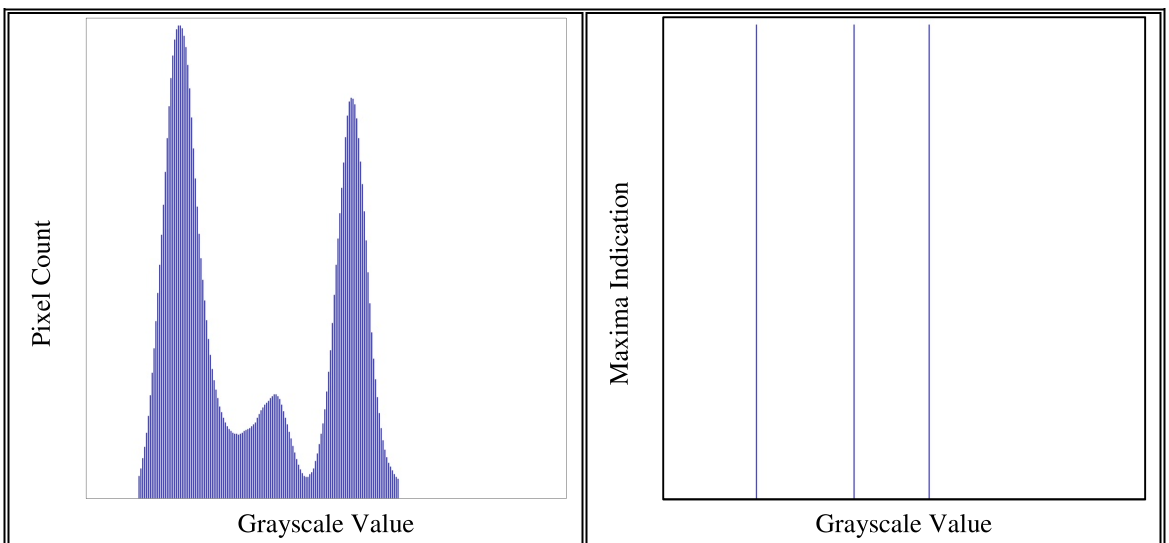
	
Original Image	Filtered Image

Figure 2: Filtering the Input Image

To dynamically determine the best number of clusters for k-means I used the histogram of the filtered image. I wanted to find the number of peaks in the histogram that were over a predetermined number of pixels and were of visible difference in color. The histogram is first smoothed with a simple Gaussian filter to remove spurious peaks. Then the global maximum is located and set to one. All values in the histogram between the point and the closest local minima higher and lower in gray-scale value are set to zero. This process is repeated until either the peaks are under a determined number of pixels (defined in a fraction of total image pixels) or the maximum peak count is reached. The maximum peak count was set to seven. This is because only five different terrain types are classified in the program. I chose seven because a cluster count over the number of terrain types in the image can improve clustering results and multiple clusters can later be identified and labeled as the same type of terrain. Figure 3 shows the maxima counting algorithm at work.



Histogram of Mt. Madonna Image	Maxima Located

Figure 3: Histogram Maxima Location

After k-means clustering the image clusters are classified using template matching. First template matching matrices are formed to indicate the match quality in the image. In these matrices a low gray-scale value or black indicates a good match and high gray-scale value or white indicates a bad matches. Then a mask is created for the target cluster. Using the mask the mean value of each of the matching matrices is calculated for the target cluster. These mean values are then compared and the lowest value indicates the best match. Therefore the cluster is labeled as the terrain type that produces the lowest mean value. The matching matrices are show in Figure 4.

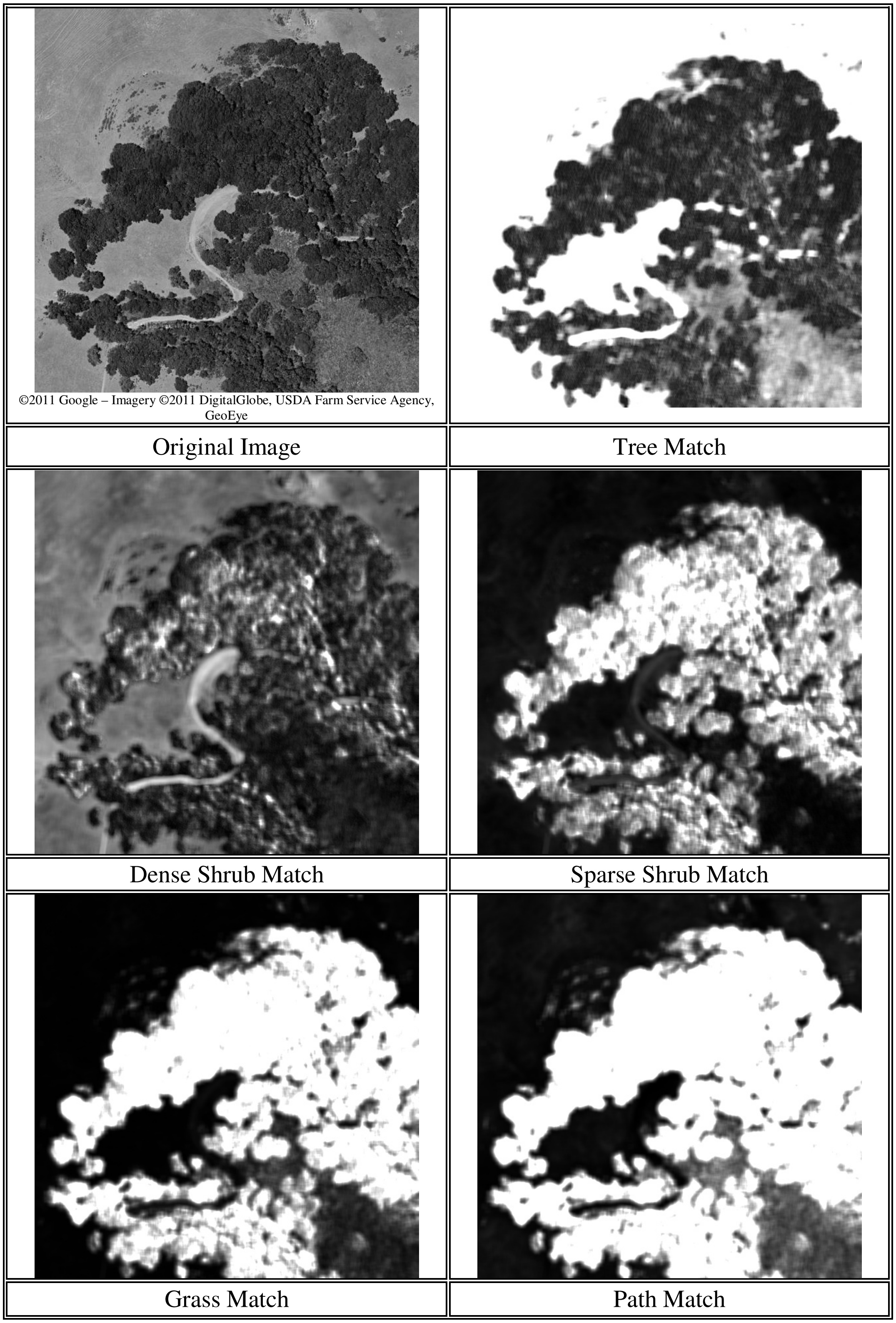


Figure 4: Match Matrices for Mt. Madonna

IV. Testing and Results

The results show strengths and weaknesses of the program. The image in Figure 5 was clustered well except for the grouping of the path with the grass. This was a problem that repeated itself because the gray scale values of grass, path, and rocks are very similar. However, all terrain was classified correctly.

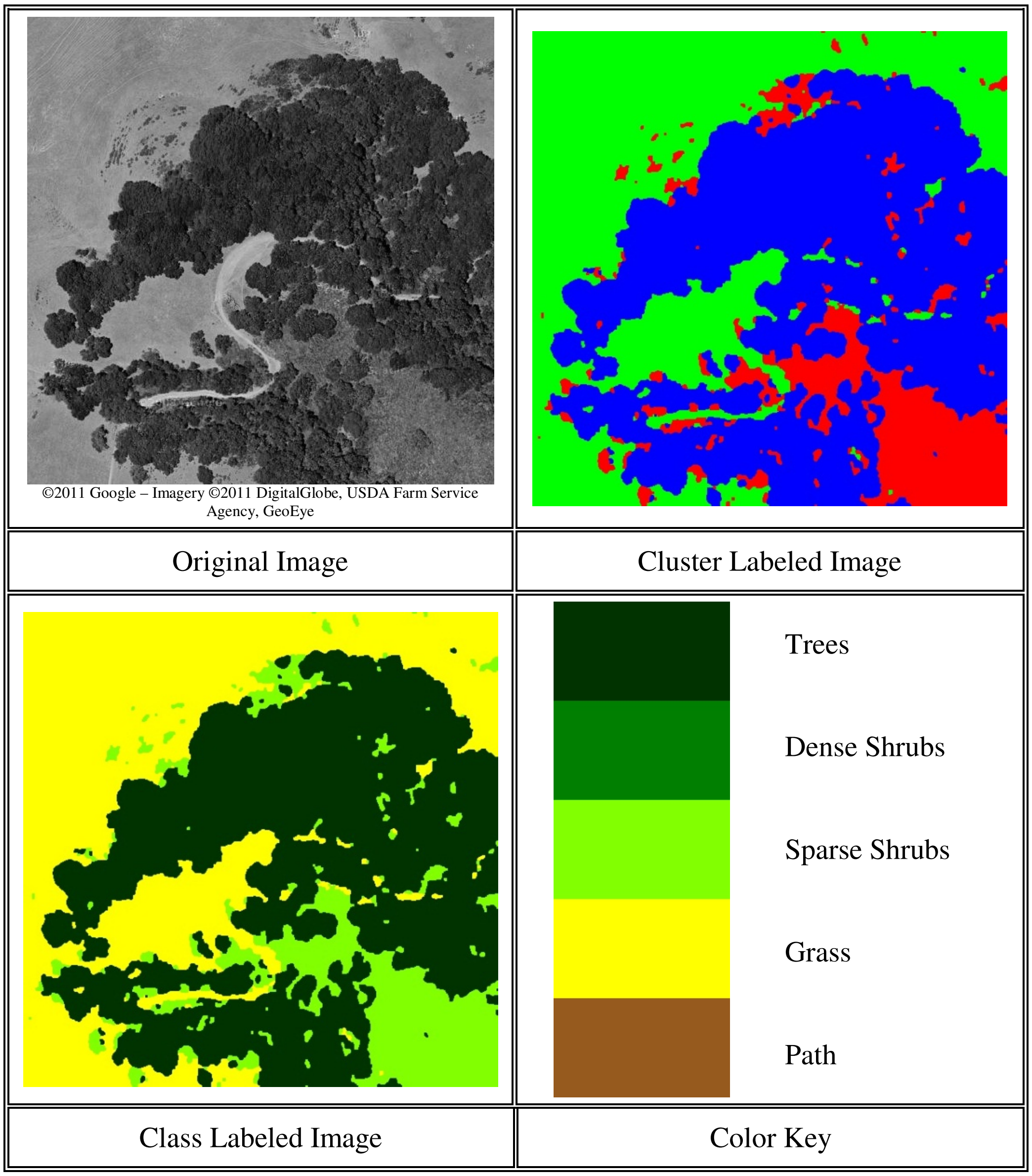


Figure 5: Mt. Madonna Trail Test Results

Again in Figure 6 the program clusters the path with grass but classifies correctly.

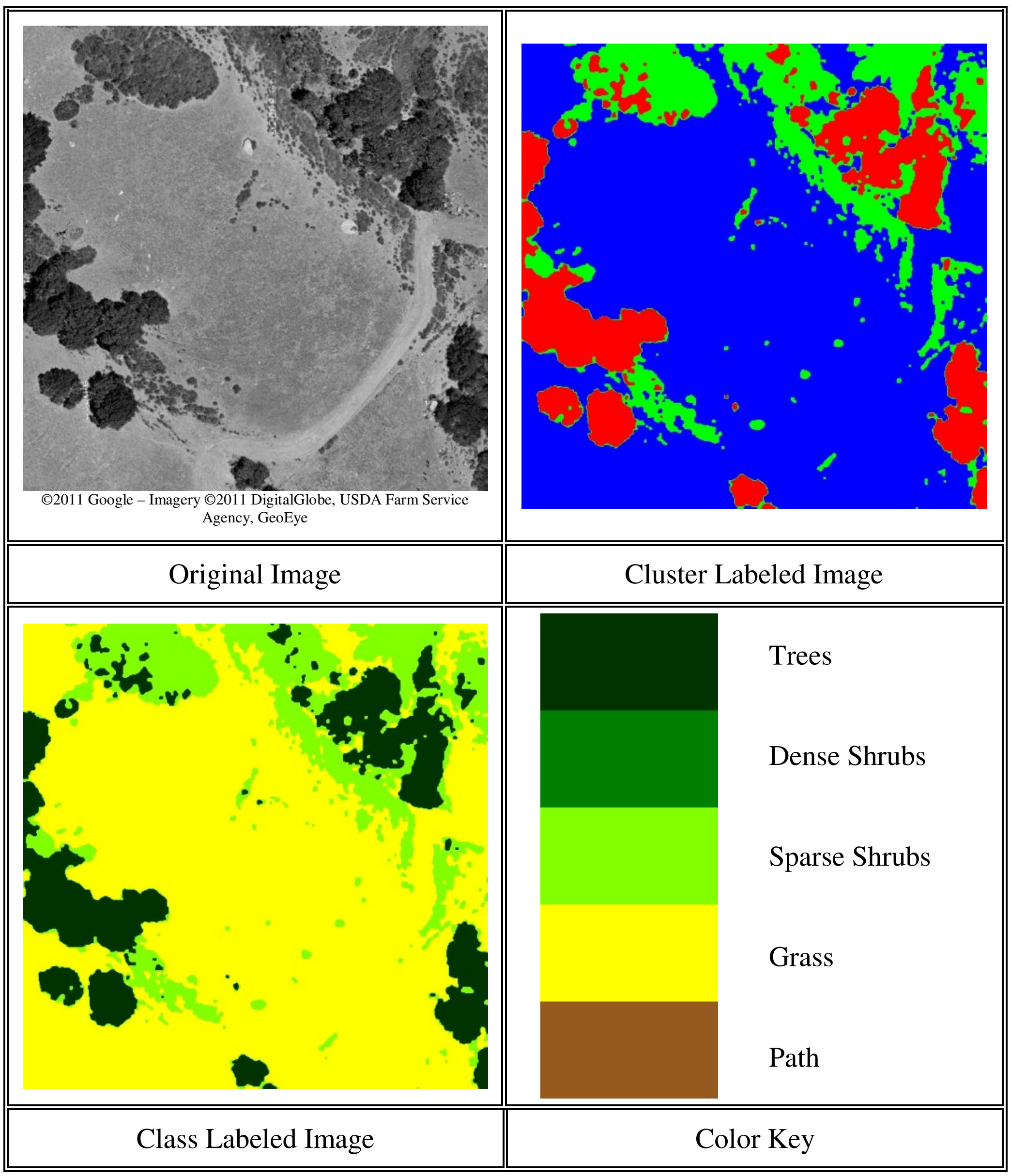


Figure 6: Mt. Madonna Base Test Results

Figure 7 shows one of the only results that labeled the path correctly in the image. However, in this image the path is clustered with rocks.

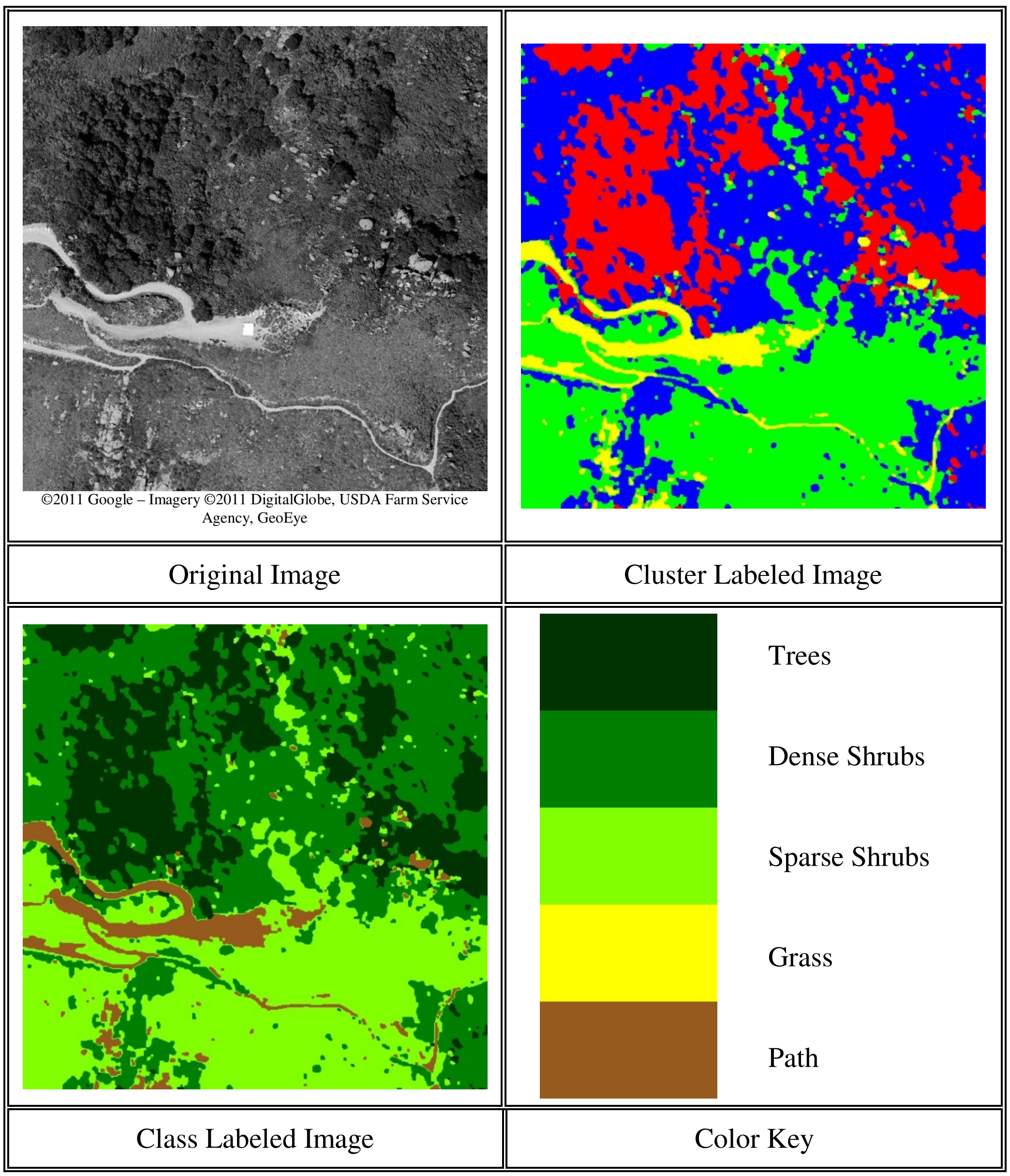


Figure 7: Mt. Madonna Top Test Results

The path in Figure 6: Mt. Madonna Base Test Results is clustered with the grass again. Otherwise, the program provided both good clustering and classification in this image.

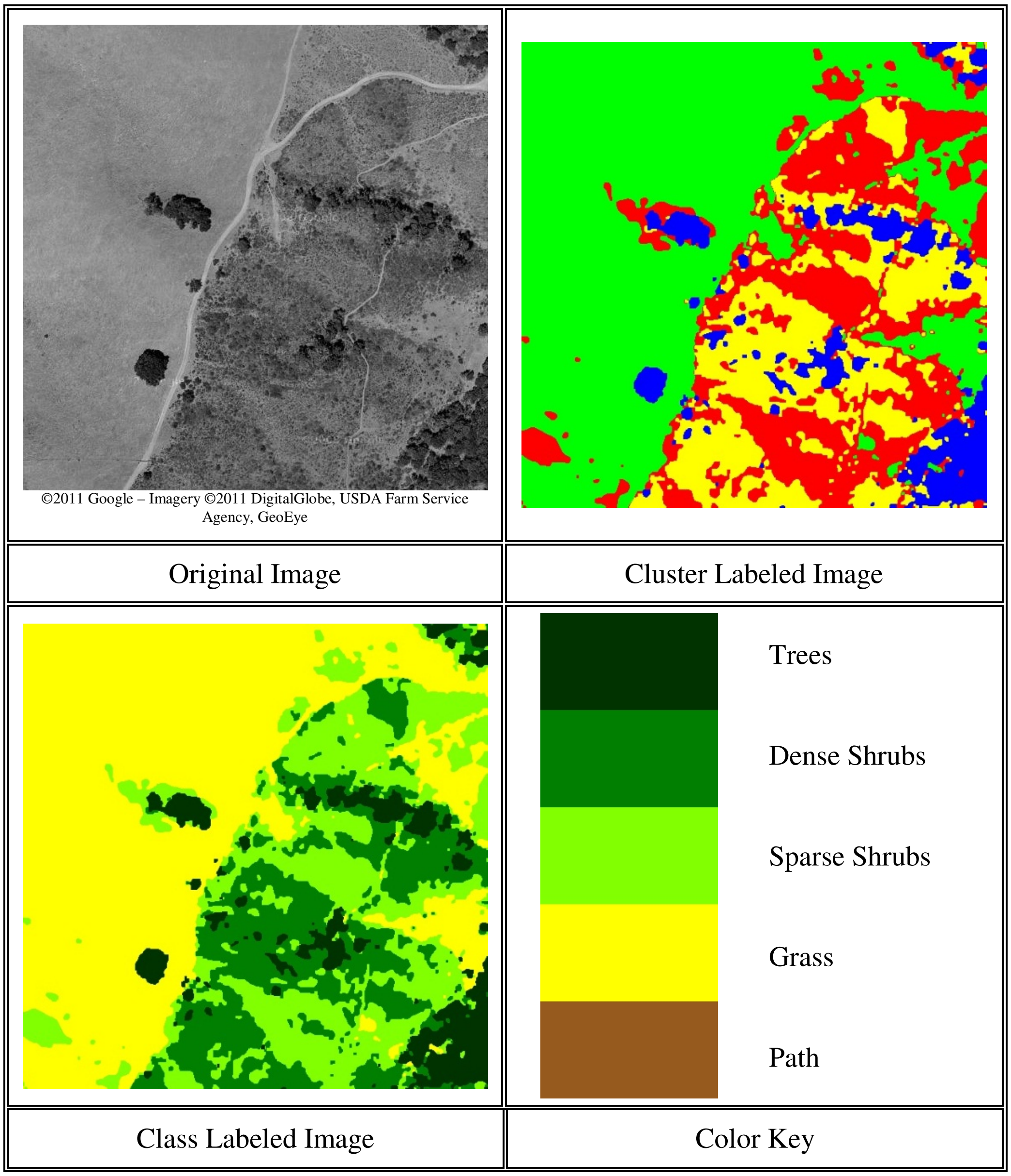


Figure 8: Bishop's Base Test Results

Figure 9: Bishop's Peak Trail Test Results also shows good results of clustering and classification.

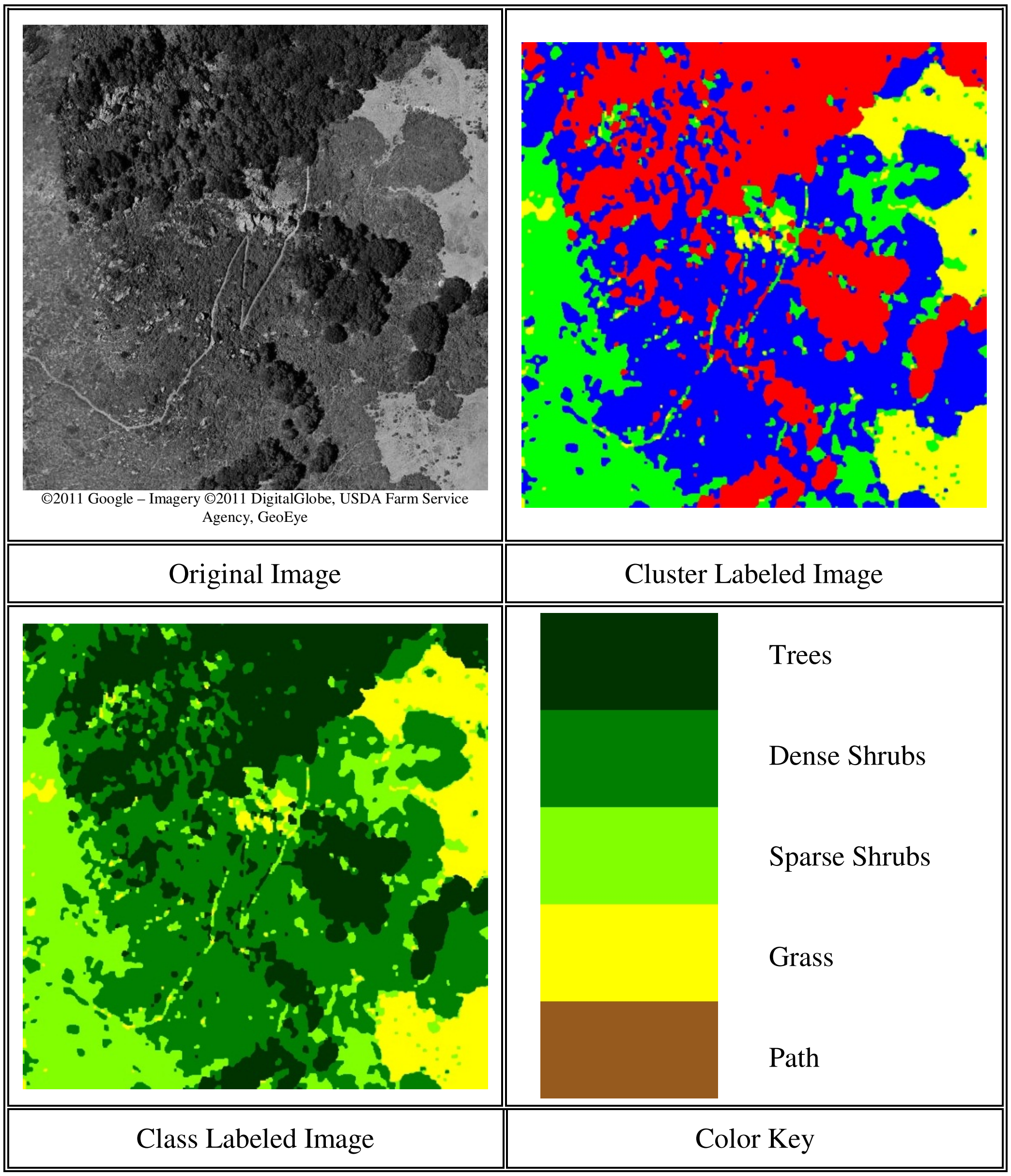


Figure 9: Bishop's Peak Trail Test Results

The canyon road image in Figure 10 exposed weakness in the classification method. The image is significantly darker than many of the other test images. The classification method does not account for variation in lighting conditions and thus labeled most of the clusters incorrectly. The clustering is very good in this image though.

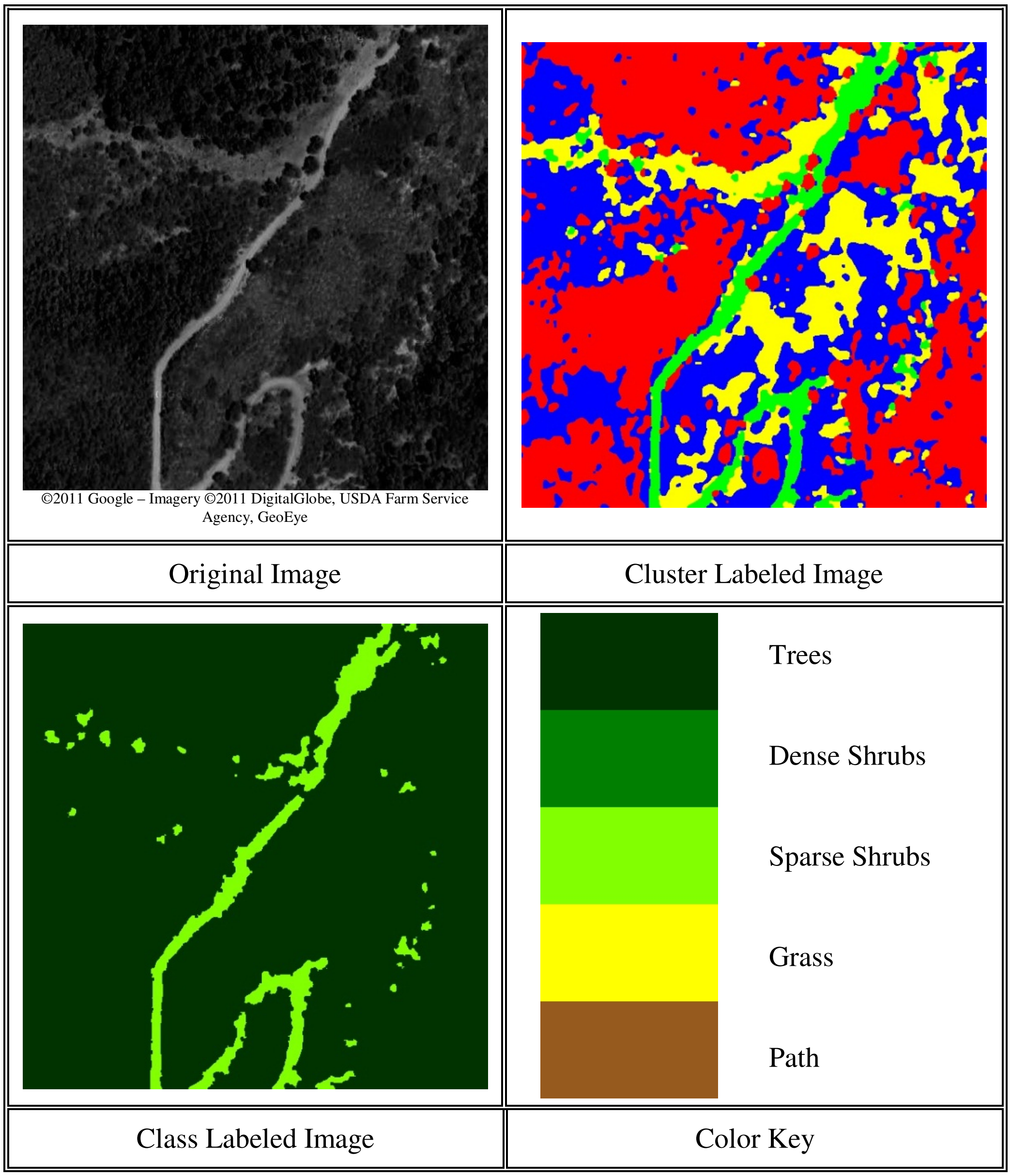


Figure 10: Canyon Road Test Results

Figure 11 displays the classification method combining clusters that are similar in texture accurately.

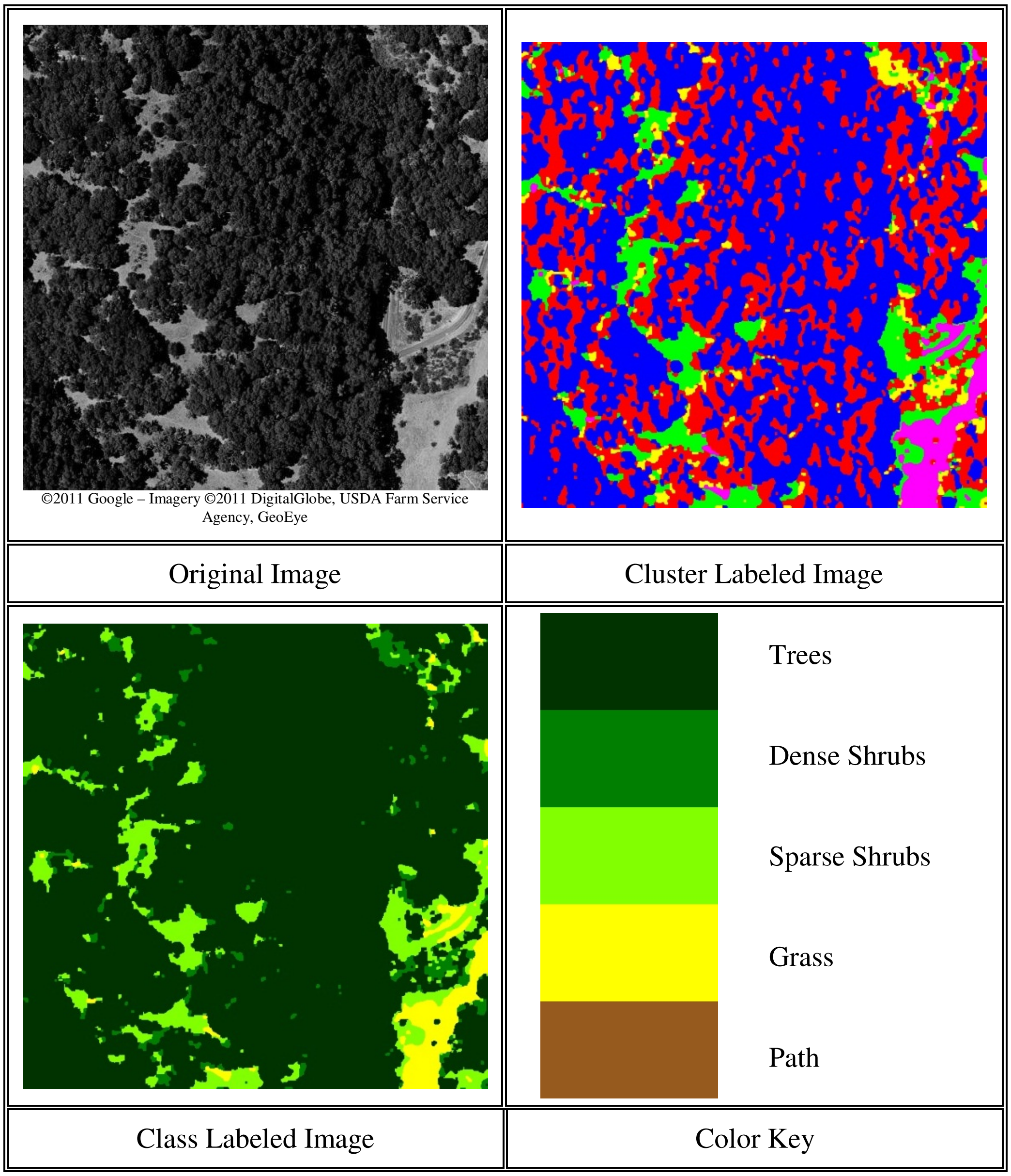


Figure 11: Atascadero Test Results

The clustering in Figure 12: Cayucos Test Results groups grass and path once again and the classification identified the grass as sparse shrubs.

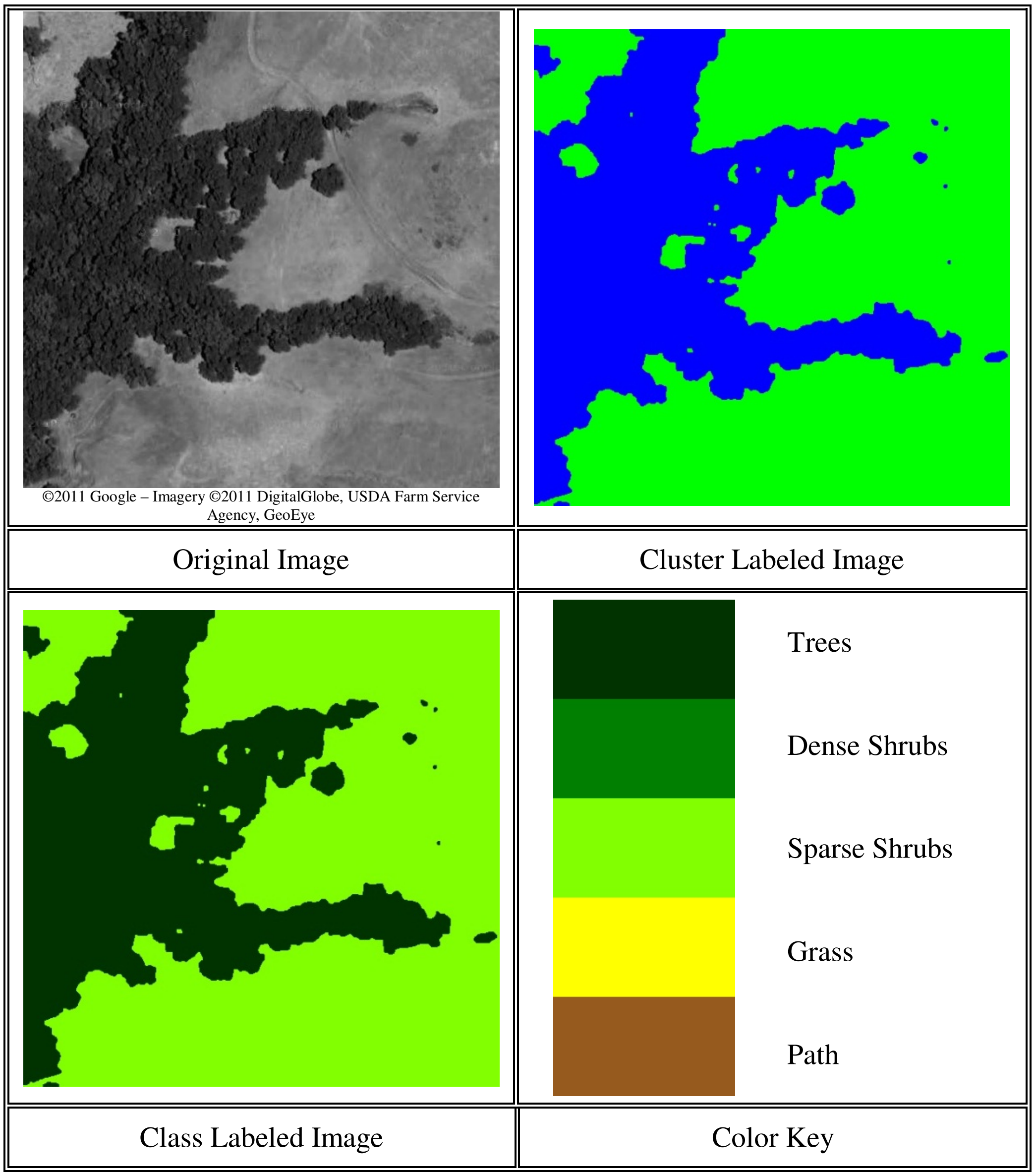


Figure 12: Cayucos Test Results

Figure 13 and Figure 14 display the programs inability to deal with new textures. The water in Laguna Lake is clustered and classified incorrectly. Although the program is not meant to detect buildings or roads, Figure 13 and Figure 15 show how these constructions disrupt segmentation and classification of the terrain in the image.

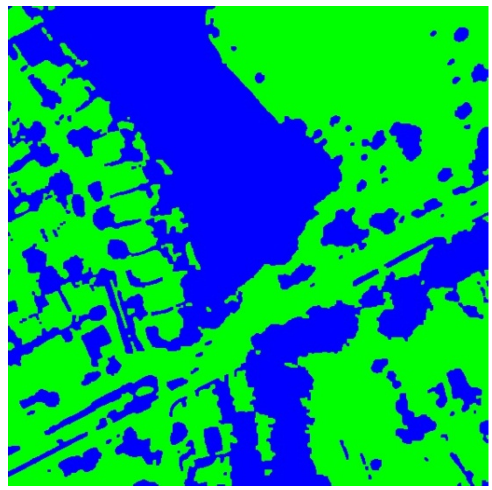
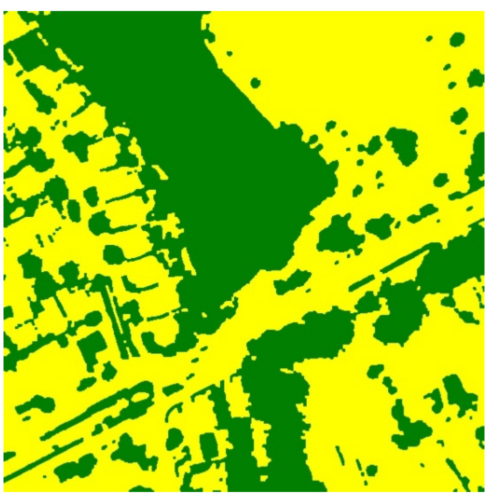
 ©2011 Google – Imagery ©2011 DigitalGlobe, USDA Farm Service Agency, GeoEye	
Original Image	Cluster Labeled Image
	 Trees Dense Shrubs Sparse Shrubs Grass Path
Class Labeled Image	Color Key

Figure 13: Madonna Road Test Results

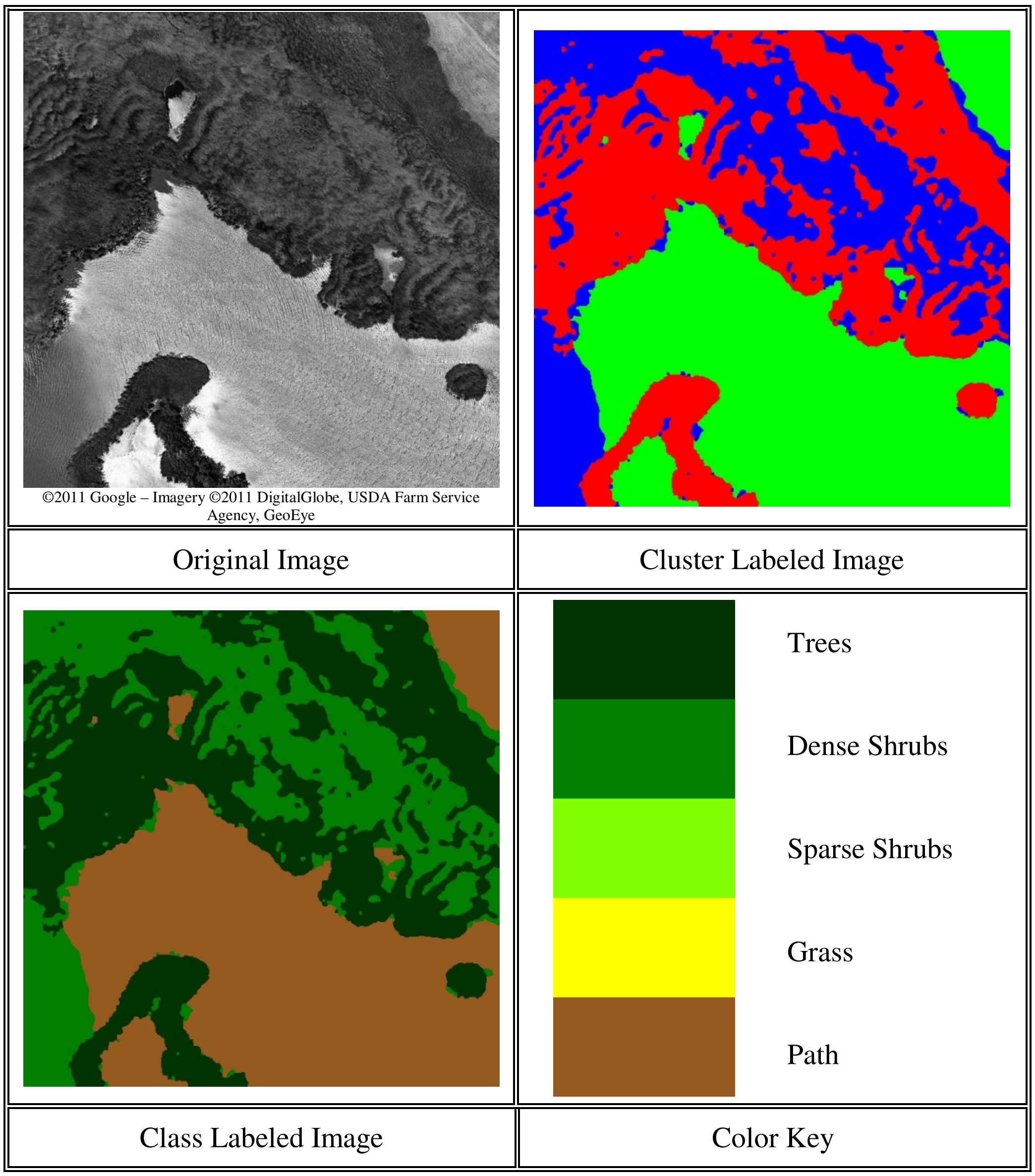


Figure 14: Laguna Lake Test Results

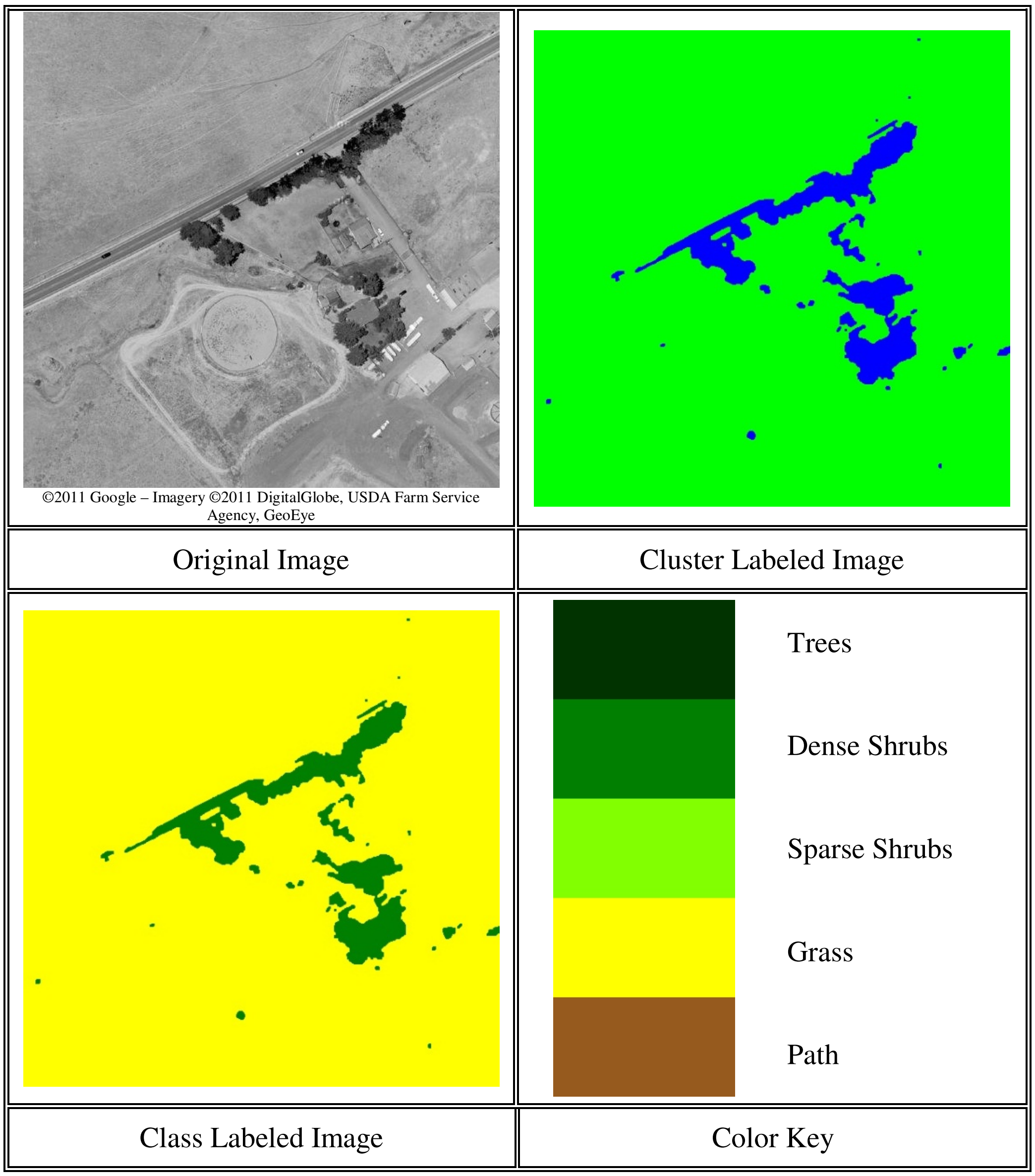


Figure 15: Foothill Blvd Test Results

V. Conclusions and Recommendations

The method of terrain classification chosen has strengths and weaknesses. Some of these weaknesses could be eliminated easily while others may require a new method to remedy.

Clustering is performed well on terrain that the program is meant to classify with the exception of the path and grass clustering problems. However, with roads, buildings or water in the image clustering and classification accuracy degrades. Identifying roads and buildings and masking them before clustering would improve clustering drastically in such images. It is possible that paths could be identified in this way as well. However, these tasks may cost computing power that will slow the application. This method is worth researching to improve performance. The same method could possibly be used to mask large bodies of water as well.

Identifying paths or roads in an image could be done with template matching if a proper template was chosen. The image could then be compared to the match image in different orientations and sizes. The problem with the path identification in my project was the focus on the texture of the path itself. A better template would search for the change in color on either side of the path.

Classification is fast but unreliable because the identification relies on only one template image. I attempted to use multiple template images and combine their mean

values for a given cluster to improve classification, but this seemed to decrease the accuracy of the classifier. A reason for this could be that the template images as well as the match images differed in brightness. I produced the template match matrices before clustering. I recommend generating match matrices after clustering and adjusting both the clusters and the template to make them gray-scale invariant. This could improve classification in images with varied brightness.

The template matching classifier is closest to a Nearest Neighbor classifier. With more research and time a more adequate classifier could be chosen. I suggest replacing the current classifier entirely.

VI. Bibliography

Forsyth, David, and Jean Ponce. *Computer Vision: a Modern Approach*. Upper Saddle River, NJ: Prentice Hall, 2003. Print.

Google Maps. Web. 20 Apr. 2011. <http://maps.google.com>.

"OpenCV 2.1 C++ Reference — Opencv V2.1 Documentation." *Welcome - OpenCV Wiki*. Willow Garage. Web. 30 May 2011. <http://opencv.willowgarage.com/documentation/cpp/index.html>.

OpenCV 2.2 Cheat Sheet (C++). Willow Garage, June 2010. PDF.

OpenCV Reference Manual V2.2. Willow Garage, Dec. 2010. PDF.

Slivovsky, Lynne. "Segmentation." Computer Vision Course. Cal Poly, San Luis Obispo. Fall 2010. Lecture.

APPENDIX A

ANALYSIS OF SENIOR PROJECT DESIGN

• Summary of Functional Requirements
The project is a terrain classification program. This program was designed to run on the RMAX helicopter being equipped for search and rescue. The program is meant to analyze images from a DragonFly2 camera from Point Grey Research that will be mounted on the helicopter. Terrain recognition is important for the autonomous operation of the search and rescue helicopter.

• Primary Constraints
The gray scale imagery was a limiting factor. The similar gray scale values for different terrain types made clustering difficult.

Another limiting factor is the ability for this program to run in real-time. Approaches that were more processor intensive were avoided in the interest of speed.

• Economic
No component parts required.
Final cost: $0
Original estimated development time: No estimation made
Actual development time: Approximately 130 hours

• If manufactured on a commercial basis:
Project will not go into manufacturing phase. It was developed for the specific application.

• Environmental
The program does not have great environmental impacts. The impacts of the project are related to the operation of the helicopter.

• Manufacturability
Project will not go into manufacturing phase.

• Sustainability
The robustness of this program must be improved before it is used on an autonomous helicopter. Images could be made gray-scale invariant to improve classification. In addition identifying and masking roads and buildings before clustering would improve performance.

The program does not impact the sustainable use of resources.

• Ethical
The use of this project is ethically good. The helicopter will be used to locate and provide aide to any person lost and/or wounded in and around San Luis Obispo County.

• Health and Safety
There is not any health concerns associated with this project. The helicopter would be a great asset to SLO Search and Rescue and therefore impact the health and safety of San Luis Obispo residents positively.

• Social and Political
The RMAX helicopter is meant for search and rescue. This project has a good social impact.

• Development
During the course of this project I learned to use the OpenCV 2.2 library. I also learned more about computer vision, specifically clustering and classification.