Vision-based Terrain Classification and Classifier Fusion for Planetary Exploration Rovers

by

Ibrahim Halatci

B.S., Mechatronics Engineering
Sabanci University, 2004

Submitted to the Department of Mechanical Engineering
In Partial Fulfillment of the Requirements for the Degree of
Master of Science in Mechanical Engineering

at the

Massachusetts Institute of Technology

September 2006

Signature of Author ....
Department of Mechanical Engineering
August 15, 2006

Certified by ....
Karl Iagnemma
Principal Research Scientist
Thesis Supervisor

Accepted by ....
Lallit Anand
Professor of Mechanical Engineering
Chairman, Department Committee on Graduate Students

Vision-based Terrain Classification and Classifier Fusion for Planetary Exploration Rovers

by

Ibrahim Halatci

Submitted to the Department of Mechanical Engineering
on August 15, 2006 in Partial Fulfillment of the
Requirements for the Degree of
Master of Science in Mechanical Engineering

ABSTRACT

Autonomous rover operation plays a key role in planetary exploration missions. Rover systems require more and more autonomous capabilities to improve efficiency and robustness. Rover mobility is one of the critical components that can directly affect mission success. Knowledge of the physical properties of the terrain surrounding a planetary exploration rover can be used to allow a rover system to fully exploit its mobility capabilities.

Here a study of multi-sensor terrain classification for planetary rovers in Mars and Mars-like environments is presented. Supervised classification algorithms for color, texture, and range features are presented based on mixture of Gaussians modeling. Two techniques for merging the results of these "low level" classifiers are presented that rely on Bayesian fusion and meta-classifier fusion.

The performances of these algorithms are studied using images from NASA's Mars Exploration Rover mission and through experiments on a four-wheeled test-bed rover operating in Mars-analog terrain. It is shown that accurate terrain classification can be achieved via classifier fusion from visual features.

Thesis Supervisor: Karl Iagnemma
Title: Principal Research Scientist

ACKNOWLEDGEMENTS

I would like to thank Dr. Iagnemma for his valuable advice and support during my research and the preparation of this thesis. His guidance and critical evaluation formed the backbone of this work. I would also like to thank Prof. Dubowsky for giving me the chance to be a part of this wonderful and inspiring group at the Field and Space Robotics Laboratory.

I would like to extend my gratitude to Christopher A. Brooks for his continuous help during the field experiments. His comments and our discussions improved the maturity of this work. Also, thanks to all members of the Field and Space Robotics Laboratory who one way or another affected my life and work for the last two years.

I would also like to thank my family, without their support and confidence I would not be here. Finally, I would like to thank my wife, Ozge, for her constant love and encouragement.

This work was supported by the Mars Technology Program at the NASA Jet Propulsion Laboratory.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1: INTRODUCTION

1.1 Motivation

One of the major recent developments in planetary rover technology is increasing autonomy. Future missions will require higher degrees of autonomy, since autonomous operation plays a key role in mission success and efficiency. Navigating vehicles autonomously allows faster command execution cycles and therefore allows effective use of limited system lifetimes (Brooks, 2004).

During the operation of the Sojourner rover, high-level rover control was performed in an open loop fashion: operator commands such as "Move straight 5 feet" were executed by monitoring the wheel rotation based on encoder count. After command execution, the rover was asked to send back images of the environment which were used by the operators to evaluate if the rover had moved as requested. Due to low data transmission rates, this control methodology required long execution periods and is inefficient for long-term mission performance.

The latest MER mission, on the other hand, has utilized a moderate amount of autonomy, employing path planning algorithms and visual odometry (Biesiadecki and Maimone; 2006). With the availability of such tools, operators have performed high-level control by pointing at a target in the local map and expecting the rover to reach the desired point in a reasonable period of time. This means rovers are capable of detecting geometric obstacles like large rocks or steep slopes, avoiding obstacles through local planning algorithms, and visually assessing their positions relative to the target so that they do not have to rely solely on encoder odometry for position estimation.

Although MER navigation algorithms have performed relatively well, there were several incidents that caused Opportunity to become immobilized for several sols. On April 29, 2005 and on May 30, 2006 Opportunity became trapped in loose drift material (NASA/JPL). It has been observed that although the environment was geometrically benign (i.e. flat), the highly deformable nature of the terrain made this region untraversable. Researchers have shown that terrain physical properties can strongly influence rover mobility, particularly in deformable and sloped terrain (Iagnemma and Dubowsky, 2002).

Scientific goals of future Mars missions include investigating surface geology, climate history, and potential for past and present life. Possible targets of scientific interest are highly rough terrains such as impact craters, rifted basins, and gullies and outflow channels (Urquhart and Gulick, 2003). Such regions are often uneven, sloped and covered with loose drift material causing rover slippage and sinkage.

As a result, the ability to detect and estimate terrain physical properties would allow a rover to intelligently predict its mobility performance and thus autonomously avoid potential obstacles and terrain regions that are likely to be non-traversable. These abilities would also let a robotic system adapt its control strategies for higher mobility performance. Thus, to ensure rover safety and robust navigation, there is a need to distinguish surrounding terrain types by means of on-board rover sensing elements.

1.2 Purpose

The purpose of this thesis is to present a study of vision-based terrain classification for planetary exploration rovers in Mars and Mars-like environments. These vision-based algorithms can provide crucial information about terrain in front of the rover that can be used to improve performance and robustness of path planning and navigation algorithms.

This work presents several low level classifiers and classifier fusion methods. The low level classifiers are based on Maximum Likelihood Estimation and employ color, texture, and geometric features of the terrain. Their outputs are combined in classifier fusion algorithms. Two techniques for merging the information from these three low level classifiers are presented.

The performance of these algorithms is evaluated using images from NASA's Mars Exploration Rovers mission and through experiments on a four-wheeled test-bed rover operating on Mars-like terrain. These results demonstrate that accurate classification of the distant terrain can be achieved via fusion of vision-based classifiers.

1.3 Background and Related Work

Methods that employ remote sensor features such as color, texture, and range have been proposed for robot perception in the context of unmanned ground vehicles operating in outdoor environments on Earth (Ramussen, 2001; Kelly *et al*, 2004; Manduchi *et al*, 2005). However, few of the methods focus specifically on terrain classification. Common problems that are addressed are obstacle detection for off-road navigation, autonomous science target detection for planetary surfaces, and road detection in unstructured environments (Dima *et al*, 2003; Castano *et al*; 2005).

Color-based classification is attractive for terrestrial applications since many major terrain types (i.e. soil, vegetation, rocks) possess distinct color signatures. Therefore, numerous researchers have proposed accurate and computationally inexpensive color-based methods for classification and segmentation of natural terrain. It is also relevant for planetary exploration rovers since multi-spectral imagers are an integral part of rover sensor suites (Squyres, *et al.*, 2003). However, it should be noted that planetary surfaces present difficult challenges for classification since scenes are often near-monochromatic due to the fact that terrain surface cover consists mainly of sand of varying composition and rocks of diverse shapes and concentration.

In (Manduchi, 2005), a color-based method was presented to classify terrain cover based on mixture of Gaussians modeling. In this thesis, a similar classifier structure will be applied to color and other visual features. Details are presented in Section 2.2. Manduchi also proposed methods for illumination compensation to integrate with a mixture of Gaussians-based classification framework (Manduchi, 2004).

Kelly (2004) demonstrated the effectiveness of multi-spectral imaging, specifically at the near-infrared [NIR] range. In his work, red channel reflectance and NIR reflectance were used as color features to distinguish vegetation and soil in outdoor scenes. It was shown that the spectral absorption properties of chlorophyll provide a unique characteristic for vegetation and thus accurate color-based classification was possible.

Although red, green and blue color channels are common for most color-based applications, some researchers employ alternative color spaces such as LUV, which represents the colors according to their luminance [L] and chrominance [U, V] (Dima, *et al.*, 2004). Distribution statistics of the color features over ground patches were included in the classification feature space as well and accurate classifier performance was demonstrated.

Image texture is a widely used feature for object recognition and image segmentation applications. In the context of terrain classification, texture-based classification is often employed to analyze aerial or satellite imaging (Jolly and Gupta, 1996). These applications mostly aimed for detection of farmland and forest areas as well as residential regions.

In the context of robot vision and recognition, Ramussen (2001) used Gabor filters to detect "denseness" of textured surfaces to distinguish roads from surrounding vegetation. Again, unique properties of vegetation (in the form of large color variance and scattered spatial structure) allow robust performance of texture-based classification.

Other researchers have proposed methods to quantify texture information such as the Fast Fourier Transform (Dima, *et al.*, 2004). In (Manduchi and Portilla, 1999) Independent Component Analysis was utilized for feature representation of images that had been convolved with filterbanks. Although texture-based methods are often computationally expensive, they have been shown to be effective at segmenting natural scenes.

Geometric features derived from range data acquired through stereo cameras or range finders have been used extensively for terrain classification and/or obstacle detection. Note, however, that such methods lack the ability to detect non-geometric hazards or identify terrain classes that do not possess geometric variation. Results of these methods are often binary maps that define terrain as either traversable or non-traversable (Goldberg *et al.*, 2002).

One of the state-of-the-art range-based obstacle detection algorithms was presented in (Bellutta, 2000; Manduchi, 2005). These algorithms exploit the elevation data in a stereo point cloud and were able to detect positive and negative geometric obstacles. Manduchi (2005) also presented a LIDAR-based method for detecting obstacles that were occluded by vegetation and not visible by stereo cameras.

Vandapel (2004) defined a new set of features to represent 3D data points using their statistical distributions in space. The focus of this work was again classifying terrestrial outdoor scenes, specifically distinguishing flat road surfaces, tall structures such as tree trunks and telephone poles, as well as scattered tree leaves and thin linear elements like telephone wires. These features are used to verify certain assumptions made in this thesis and are detailed in Section 2.5.2.

Avedisyan (2004) presented a unique approach for detecting hazardous regions in a scene. The proposed method was designed for far-field navigation and employed color and texture features to detect discontinuities and anomalies in the scene. It utilized stereo processing to identify image range and the intersection of equidistant curves which pointed to anomalies on the ground plane caused by large negative obstacles.

Besides analyzing visual features individually, some researchers have proposed combining these different modes of information to improve classification results. The most common approach is data fusion which essentially combines all available features to generate a single "super" feature vector. However, none of these applications directly concentrated on terrain classification, but rather focused on road detection (Ramussen, 2001; Dima, *et al.*, 2003) or science target detection for planetary rovers (Gor, *et al.*, 2001; Castano, *et al.*, 2005; Thompson *et al.*, 2005; McGuire, *et al.*, 2005).

In summary, most remote sensing techniques developed for robot perception has primarily focused on terrestrial applications where the visual features possess a wide variance and allow robust classification. Planetary surfaces, in the contrary, have less variance in terms of observed visual features. Also, the main goal of previous research has often focused on detecting roads or obstacles where the object of interest presents high contrast with the rest of the scene.

1.4 Outline

This thesis has five chapters followed by four appendices. This chapter is the introduction, summarizing the motivation behind this research and explaining its purpose. Related research is also described.

Chapter 2 describes the Mixture of Gaussians [MoG] -based Maximum Likelihood classifier model, which is the standard architecture for all classifiers studied in this thesis (except for the Bayesian fusion presented in Section 3.2). Low level classifiers based on color, texture, and range, and feature selection/extraction methods for these classifiers are also explained in this chapter.

Chapter 3 describes the classifier fusion and data fusion methods. Bayesian fusion is the first classifier fusion method that assumes independence of visual features. Meta-classifier fusion has a similar structure to the low level classifiers, but utilizes low level classification confidences instead of visual features. Data fusion is presented for baseline comparison which requires combining all available visual features for classification.

Chapter 4 is the results section. The low and high level classification algorithms were tested on two different data sets: One data set was compiled from MER imagery and another is from rover experiments conducted at Wingaersheek Beach outside Boston. This chapter describes the experimental results and discusses classifier performances.

Chapter 5 is the conclusion of this thesis which discusses the contributions of this research and possible extensions for future work.

Background material and equipment descriptions are presented in the appendices. Appendix A presents details of the Expectation Maximization [EM] algorithm, the method used for off-line learning of classifier parameters. Appendix B explains the Wavelet Transform, which was used in the texture feature extraction algorithm. Appendix C contains details about the MER image set. It also summarizes how to access the image database, interpret PDS labels and access stereo data. Appendix D describes the FSRL Technology Test-bed Rover, TORTOISE, which was used to collect experimental data.

Chapter 2: LOW LEVEL CLASSIFIERS

2.1 Introduction

In this thesis, low level classifiers refer to classification methods that rely solely on a single visual feature type. The features that are used in this work are color, texture, and range (i.e. geometric properties of terrain). Some of these visual features are directly available from raw sensor signals, whereas other features require preprocessing before being fed to the classifier. However for all sensing modes, a forward-looking color stereo pair is sufficient to capture the raw data.

In unstructured environments such as Mars, unique visual features provide unique pieces of information about a scene. However, none of these visual features are alone sufficient for robust classification. These scenes include rock formations that are buried or partially covered with drift material, and sandy plains that are accumulated to form dunes or lie flat between closely scattered discrete rocks. As explained in Section 1.3, classifiers utilizing visual features individually have been studied extensively for terrain classification in relatively structured tasks such as road detection or geometric obstacle detection.

Section 2.2 describes the Mixture of Gaussians-based [MoG] modeling and classification algorithm. The color-based classification approach is described in Section 2.3, the texture-based classification approach is described in Section 2.4, and the range-based classification approach is described in Section 2.5.

2.2 Mixture of Gaussians Classifier

A Mixture of Gaussians-based classification structure forms a basis for all the low level classifiers discussed in this thesis. This classifier is implemented as a framework that models a feature's probability density function as a mixture of several Gaussian functions. In this framework, the likelihood of the observed features, y, given the terrain class, x_i, is computed as a weighted sum of K number of Gaussian distributions (Bishop, 1995) as seen in Equation 1.

$$f(y \mid x_i) = \sum_k^K \alpha_k G(y, \mu_k, \Sigma_k) \tag{1}$$

Here α_k is the weighting coefficient of the k^{th} component, and G is a Gaussian distribution in a d - dimensional space with mean μ and covariance Σ:

$$G(y, \mu, \Sigma) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} e^{-\frac{1}{2}(y-\mu)^T \Sigma^{-1} (y-\mu)} \tag{2}$$

Mixture of Gaussians classification consists of the following two steps: 1) off-line training, 2) on-line classification. These will be described in Section 2.2.1 and 2.2.2, respectively. The flowchart for this algorithm is shown in Figure 2.1.

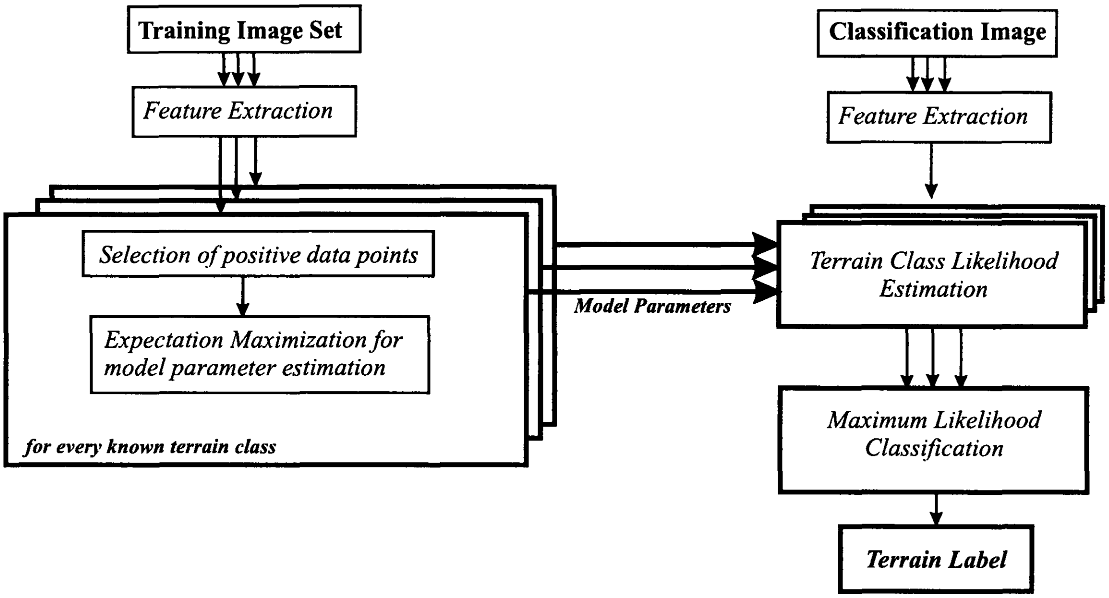
Training Image Set
Feature Extraction
Selection of positive data points
Expectation Maximization for model parameter estimation
for every known terrain class
Model Parameters
Classification Image
Feature Extraction
Terrain Class Likelihood Estimation
Maximum Likelihood Classification
Terrain Label

Figure 2.1: Mixture of Gaussians classifier structure

2.2.1 Off-line Training for Mixture of Gaussians Classifier

During off-line training, parameters of a features' probability distribution model describing each terrain class are computed separately. For this computation, training algorithm requires representative data for each class. For example, when training the "rock" class model, data points labeled as "rock" are positive data points and data points labeled as "sand" or "mixed" are negative data points.)

During this off-line process, the expectation maximization [EM] algorithm is used to estimate model parameters such as α, μ, and Σ. Besides the training data, this algorithm requires initial conditions for the parameters to be estimated. Initial conditions for the mean vectors, μ_i, are initialized by clustering the training data into K clusters using a K-means algorithm (Manduchi *et al,* 2005). The other initial conditions are chosen as:

$$\begin{aligned}\alpha_i &= \frac{1}{K} \\ \Sigma_i &= I_{dxd}\end{aligned} \tag{3}$$

Here, *d* is the dimension of the feature space and *K*, the number of Gaussian modes, must be predefined. Although methods exist for estimating *K* for optimal model fitting based on Monte Carlo Cross Validation (Smyth, 1996), they are computationally intensive. In this work, *K* is tuned based on empirical analysis. Similar to Manduchi (2005), reasonable results have been obtained with three to five Gaussians. Using a greater number of modes often causes over-fitting and classifier performance evolves as shown in Figure 2.2. In this case, increasing model complexity to yield improved fitting to training data would not necessarily yield the best representation of real (i.e. test) data.

The result of the EM algorithm is a set of Mixture of Gaussians model parameters based on the training data and initial conditions. Details of the EM algorithm are included in Appendix A.

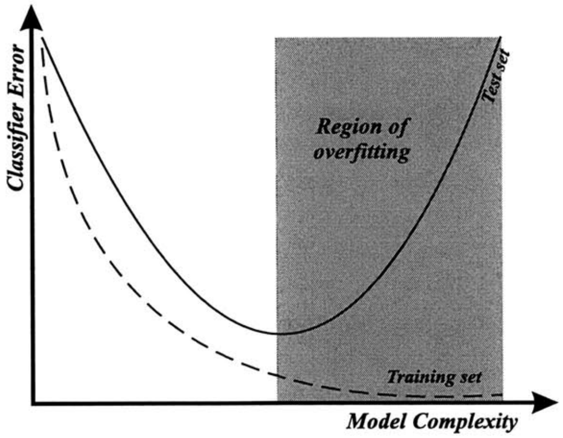

Figure 2.2: Over-fitting regime for model based classifiers

2.2.2 On-line Classification

Given the Mixture of Gaussians model parameters related to a visual feature of a terrain class, the online classification process is a simple matter of plugging in these feature vectors to the distribution model and computing its likelihood of belonging to a particular class. The final classifier outputs are class labels assigned according to the Maximum

Likelihood [ML] criteria (Equation 4) and the conditional likelihood of the assigned class. Note that it is also possible to assume terrain class priors are equally likely and use the Maximum a posteriori [MAP] classifier (Equations 5-6).

$$\hat{x} = \arg\max_{x_i}\left(f(y \mid x_i)\right) \tag{4}$$

$$\begin{aligned} P(x_i \mid y) &= \frac{P(x_i) f(y \mid x_i)}{\sum_i P(x_i) f(y \mid x_i)} \\ &= \frac{P(x_i) f(y \mid x_i)}{f(y)}; \quad i = 1,..,n \Rightarrow P(x_i) = 1/n \end{aligned} \tag{5}$$

$$\hat{x} = \arg\max_{x_i}\left(P(x_i \mid y)\right) \tag{6}$$

2.3 *Color-based Classifier*

Color is one of the major distinguishing characteristics of many terrain types. Although perceived color is a function of illumination and is affected by various physical factors such as shadowing and reflectance, color-based classification has been used to accurately identify natural terrain (Belluta *et al.*, 2000; Manduchi *et al.*, 2005). However, it should be noted that the color distribution on many planetary surfaces, including Mars, is somewhat limited. Specifically, the lack of moisture on Mars not only limits the possible classifiable terrain classes but also narrows the color variation of distinct terrain types.

Figure 2.3 shows a sample Mars scene where two major Mars terrain classes, rock and sand, are imaged at close range at various wavelengths. A large rock lies at the center of the scene. The rest of the scene shows drift material surrounding the rock. These series of images show how these two physically dissimilar terrain classes could appear similar through remote sensing and thus underlines the fact that Mars scenes are nearly monochromatic. Also note that the circular marks on the center of the rock are results of the Rock Abrasion Tool [RAT] on the Mars Exploration Rover, which removes the dust coating on the rock and thus reveals the actual color of the rock. It can be seen that drift material cover on the Martian surface can drastically degrade the performance of color based classifiers.

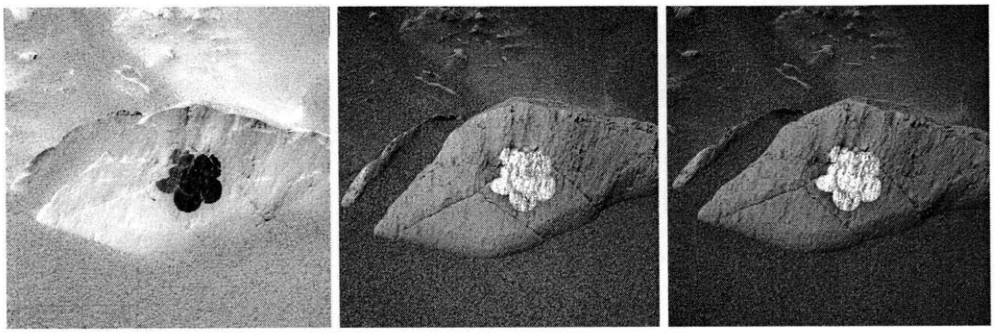

Figure 2.3: Sample scene from different filters (NASA/JPL, 2006)

In this work red, green and blue channel intensity values are selected to form a 3-dimensional feature vector for every image pixel. Construction of this feature vector for MER imagery is complicated by the nature of the rover imaging system and is explained in Section 4.2.2.

2.4 *Texture-based Classifier*

Another visual feature studied in this thesis is image texture. Image texture is a measure of spatial variation in intensity and is often used to detect structural, roughness or regularity differences in an image (Rajpoot, 2004). For images of natural scenes, different spatial irregularities exist at different scales. For example, the "grass" terrain class can present diverse textural appearances when observed at different distances, as seen in Figure 2.4. In this thesis, the texture length scale of interest is on the order of tens of centimeters. This scale allows observation of terrain surfaces in the range of four to thirty meters, which correspond to the range of interest for local navigation planning (Goldberg, *et al.*, 2002). Figure 2.5 shows an example of the texture signatures to be distinguished in a sample Mars scene.

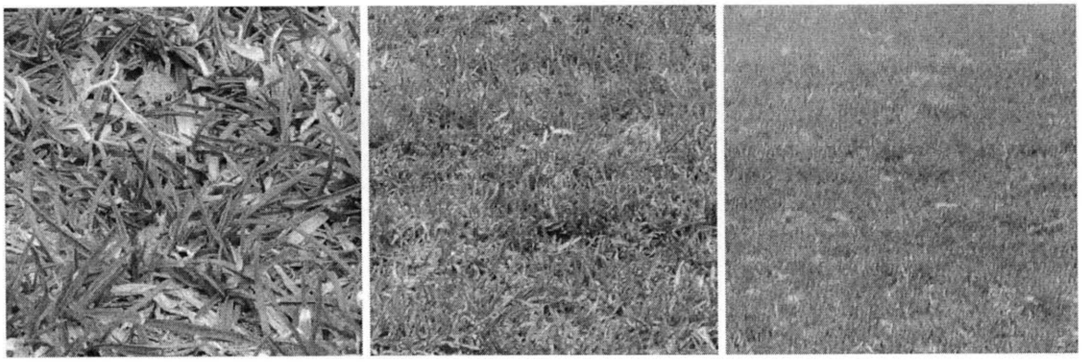

Figure 2.4: Grass terrain class at different scales

Methods for quantifying texture information have been studied extensively in the computer graphics and machine vision literature. These include methods such as model based techniques, statistical techniques like co-occurrence, and filtering approaches that employ Gabor or wavelet filters (Reed & Du Buf, 1993; Randen & Husoy, 1999). In this work, a wavelet-based fractal dimension method is employed that yields robust results in natural texture segmentation, as demonstrated by (Espinal, *et al.*, 1998).

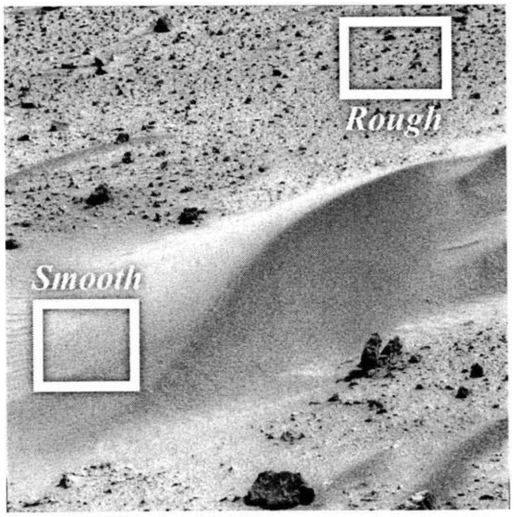

Figure 2.5: Sample textures for Mars terrain

2.4.1 Wavelet Transform for Feature Extraction

In texture-based classification, the wavelet transform is applied to continuous signals to decompose them into a family of functions which are generated from a "mother" wavelet. In the case of discrete wavelength transforms, the purpose is to compute the wavelet coefficients. This computation is similar to passing the signal from low and high pass filters and down-sampling with a factor of two (Mojsiliovic, 2000). Details on wavelet transforms are included in Appendix B.

The 2-dimensional wavelet transform is achieved by simply applying one-dimensional transforms in abscissa and ordinate directions, which yields a lower resolution sub-image (*LLx*) along with three wavelet coefficient images (*HLx, LHx,* and *HHx*). Additional levels of decomposition are computed by further transforming the sub-image. Figure 2.6 illustrates the wavelet transformation step-by-step. Low pass and high pass filters are first applied to an $2n \times 2n$ sized image in the horizontal direction to yield two sections of size $2n \times n$. Then these same filters are applied on the resulting images in the vertical direction which generats four sections of size $n \times n$. The upper left quadrant is then the result of two low pass filters and is called LL1, which will be used for the next level of transformation.

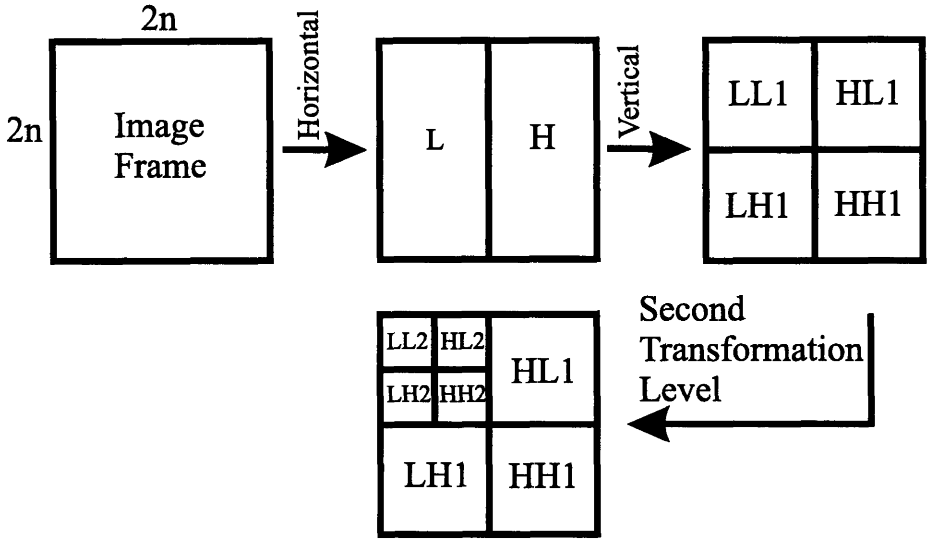

Figure 2.6: 2D Wavelet transform

The wavelet coefficients of every transformation level are used to extract the directional texture component for a scene. This method assumes that the texture in the scene is present in the high frequency components and thus high-pass filters preserve the texture information. Therefore, the upper right quadrant (HL1) contains horizontal components of the texture signature. Similarly, the lower right quadrant (HH1) contains diagonal components, and the lower left quadrant (LH1) contains vertical components. The same structure follows for the second and third levels of transformation. Note, however, that frequency bands for each higher transformation level are distinct and thus multi-level transformation allows detection of different spatial frequency components at different scales.

After transformation, the texture signature of every pixel is extracted for three dimensions. For every pixel, a directional feature coefficient is computed by averaging the amplitudes of wavelet coefficients in a predefined neighborhood. Every transformation level utilizes a different neighborhood window that scales with the transformation level. This feature extraction method yields the HDV (Horizontal, Diagonal, and Vertical) feature space. A 3-dimensional feature vector is generated for every pixel.

In this thesis, a Haar wavelength kernel is used for wavelet transformation with coefficients [1 -1] for high pass filters and [1 1] for low pass filters. The images are decomposed into 3 frequency levels, and neighborhood windows of 11, 9, and 7 pixels are used for 1st, 2nd, and 3rd transformation levels, respectively.

2.4.2 Texture Dependence on Distance

It is trivial for human beings to observe the effect of distance variation on visual texture. Essentially, increasing distance results in a decrease in the observable level of texture detail on a surface. Therefore, the variation in the texture metric described above as a function of distance was studied. As noted above, the MER images normally span distances from four to thirty meters, and observed features vary for a given terrain class.

In this analysis, the change in a texture feature vector coefficients as a function of distance was analyzed for various terrain classes. Numerical values of the HDV components of the texture feature normally take values in the range of [0 1], and since this texture metric can be viewed as a measure of roughness, a value of 0 represents a smooth surface.

It was observed that the coefficients in all three directions tend to diminish with increasing distance. In other words, terrain surfaces appeared smoother as the distance to the observed surfaces increased. The only exception was observed in the sand class. Here, it is hypothesized that the deformability of this terrain type causes intensity variations to be observed at lower spatial frequencies.

Table 2-1 presents the bounds in the HDV space in which the feature points were distributed. When similar analyses were performed on “rough’ and “smooth” classes of MER imagery, it was observed that the coefficients of the “rough” class decreased as observation range increased. On the other hand, the average values of the feature vectors for the terrain patches were still clearly clustered in the HDV space, suggesting that texture may still be valuable for classification at long ranges.

Table 2-1: Bounding box borders for texture features as a function of distance in the HDV space

Terrain Classes	Observation Distance		
	Less than 1 m	1 m to 4 m	4m to 10 m
Grass	$[0.83\ \ 0.51\ \ 0.84]^T$	$[0.61\ \ 0.38\ \ 0.60]^T$	$[0.18\ \ 0.22\ \ 0.49]^T$
Gravel	$[0.64\ \ 0.33\ \ 0.42]^T$	$[0.53\ \ 0.48\ \ 0.56]^T$	$[0.45\ \ 0.31\ \ 0.36]^T$
Sand	$[0.51\ \ 0.39\ \ 0.15]^T$	$[0.49\ \ 0.37\ \ 0.14]^T$	$[0.62\ \ 0.69\ \ 0.34]^T$

2.5 Range-based Classification

Range-based classification analyzes the terrain surface geometry information to distinguish between terrain classes that have distinct geometric properties. Unlike many obstacle detection algorithms that utilize elevation points, the range-based classifier in this work is designed to discriminate physical terrain classes according to their appearances in 3-dimensional space. Simply, the appearances are discriminated by the slope of a given terrain patch and the maximum step height. This feature selection is based on the observation that on the Mars surfaces, rocky terrain (in the form of either outcrop or large scattered rocks) projects from the ground plane while sandy terrain generally lies parallel to the ground plane. For accumulated cohesionless drift material, the slope cannot be greater than its angle of repose, while rocky terain has no such limit.

Figure 2.7 shows a typical Mars scene. The scene is composed of gently sloping sand in the foreground with a large outcrop in the mid range and scattered rocks mixed with sand in the background. There is also small patch of mixed terrain in the bottom right corner.

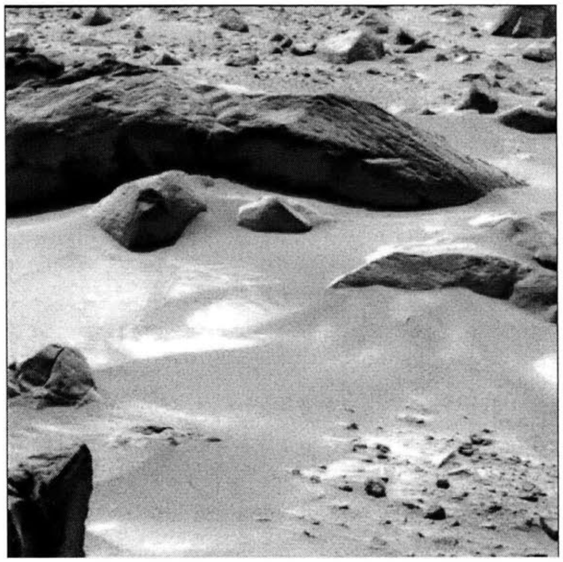

Figure 2.7: Sample Mars scene

2.5.1 Feature Selection

In this work, range data was acquired through stereo processing to yield a point cloud in Cartesian space. A sample elevation map for the scene in Figure 2.7 is seen in Figure 2.8. Range features are constructed from the normal vector of best-fit surfaces to 20 cm by 20 cm grid patches. The size of the grid patches was empirically selected to be equal to the rover wheel diameter in order to capture the minimum-sized terrain area that might significantly affect a rover's mobility. The surface fit representation of the terrain can be considered as a discrete approximation to the terrain cover. Note that this feature detection method was not trying to estimate the actual ground plane, but instead computes the slope and step height of the local patches.

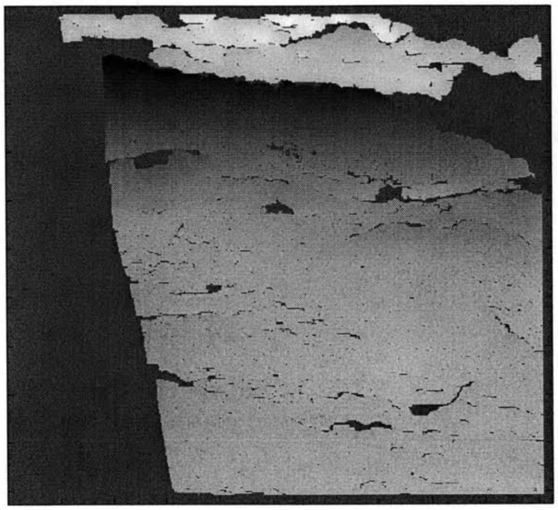

Figure 2.8: Elevation map for the scene in Figure 2.7

Computation of the feature vector for the terrain patches was performed as follows. For every grid patch, least squares estimation was used to compute the best fit surface to the point cloud. The equation of a planar surface in 3-dimensional Cartesian space is given as:

$$A_1x + A_2y + A_3z + A_4 = 0 \tag{7}$$

Assuming $A_4 \neq 0$ which means the terrain cover is not strictly on the *xy*-plane of the fixed world frame) gives:

$$\overline{A}_1x + \overline{A}_2y + \overline{A}_3z = -1 \tag{8}$$

Therefore the parameters to estimate are:

$$\mathbf{\Theta} = \begin{bmatrix} \overline{A}_1 & \overline{A}_2 & \overline{A}_3 \end{bmatrix}^T \tag{9}$$

They are also the components of the surface normal and the observations are

$$\mathbf{\Psi} = \begin{bmatrix} \vdots & \vdots & \vdots \\ \mathbf{x} & \mathbf{y} & \mathbf{z} \\ \vdots & \vdots & \vdots \end{bmatrix} \in R^{nx3} \tag{10}$$

Equation 8 can be rewritten as a combination of Equations 9 and 10:

$$\mathbf{\Psi}^{\mathbf{T}}\mathbf{\Theta} = \mathbf{Y} = \begin{bmatrix} -1 \\ \vdots \\ -1 \end{bmatrix} \in R^{nx1} \tag{11}$$

Here n is the number of data points in a particular grid patch. A least squares estimation algorithm computes the coordinates of the surface normal as:

$$\hat{\mathbf{\Theta}} = \left(\mathbf{\Psi}\mathbf{\Psi}^{\mathrm{T}}\right)^{-1}\mathbf{\Psi}\mathbf{Y} \tag{12}$$

Then the feature vector for a grid patch is constructed as:

$$\mathbf{v} = \begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\beta} \\ \hat{\mathbf{z}} \end{bmatrix} = \begin{bmatrix} a\tan\left(\frac{\overline{A}_1}{\overline{A}_2}\right) \\ a\tan\left(\frac{\overline{A}_3}{\sqrt{\overline{A}_1^{\,2} + \overline{A}_2^{\,2}}}\right) \\ \left|z_{\max} - z_{\min}\right| \end{bmatrix} \tag{13}$$

Here, the first two components are the azimuth and inclination angles of the surface normal vector as seen in Figure 2.9, while the third component is the step height within the grid.

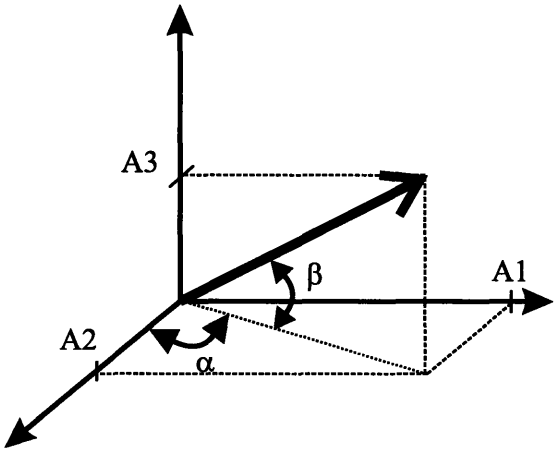

Figure 2.9: Range features from surface normal

2.5.2 Alternative Range Features

A feature extraction method has been presented by Vandapel (2004) for analyzing range data of natural scenes. In this approach, the covariance of the data points in a point cloud is used to estimate the 3-dimensional structure. These so-called "Vandapel – Hebert features" represent the "point-ness", "curve-ness", and "surface-ness" of a point cloud through the eigenvalues of the symmetric covariance matrix given as

$$\mathbf{M}_{3x3} = \frac{1}{N}\sum(\varphi - \overline{\varphi})^T(\varphi^T - \overline{\varphi}) \tag{14}$$

Here N is the number of data points in the region of interest and

$$\overline{\varphi} = \frac{1}{N}\sum \varphi \tag{15}$$

Vandapel (2004) claims that the eigenvalues are approximately equal ($\lambda_0 \sim \lambda_1 \sim \lambda_2$) for scattered points resulting from vegetation, one eigenvalue is much smaller than the others (λ_0, $\lambda_1 >> \lambda_2$) for linear structures like tree trunks; and one eigenvalue is much larger than the others ($\lambda_0 >> \lambda_1$, λ_2) for planar structures, where the principal direction indicates the direction of the surface normal. Therefore, the feature vector based on "Vandapel-Hebert features" is constructed as:

$$\begin{bmatrix} \text{point}-\text{ness} \\ \text{curve}-\text{ness} \\ \text{surface}-\text{ness} \end{bmatrix} = \begin{bmatrix} \lambda_2 \\ \lambda_0 - \lambda_1 \\ \lambda_1 - \lambda_2 \end{bmatrix} \tag{16}$$

The primary assumption of the feature selection method presented in Section 2.5.1 is that the terrain is dominated by planar surfaces. When Vandapel – Hebert features were applied to Mars scenes with a fixed 20 cm by 20 cm support region, it was observed that "surface-ness" feature was dominant and hence the planar assumption for the terrain features appears justified.

Chapter 3: HIGH LEVEL CLASSIFIERS

3.1 Introduction

In this chapter, two classifier fusion techniques are presented as tools for merging low level classifier results. Classifier fusion is designed to utilize low level classifier results as inputs and yield improved class decisions. The purpose of this fusion approach is to compensate for the weaknesses of each individual classifier and thus produce a more accurate and robust terrain map. For baseline comparison, a data fusion method is also presented.

3.1.1 Motivation for Fusion Algorithm

Low level classifiers can yield poor results when applied individually in certain problem domains. For example, color is sensitive to illumination effects; therefore wide lighting variations can give rise to poor color classification performance. Similarly, texture feature characteristics vary with distance, as presented in Section 2.4.2. Therefore, a texture-based classifier often cannot perform equally robustly for a scene with widely varying range properties. Range-based classification is also highly dependent on accurate stereo processing, as it requires accurate and dense stereo data for good performance. Also range-based classification sometimes results in misclassification due to geometric similarities between certain terrain shapes. For example, steep sandy slopes can be confused with large rock obstacles by range-based classifiers.

To apply classifier fusion, distinct data types that are to be merged must have the same resolution. In other words, color- and texture-based data should be represented in a patch-wise form to be combined with range-based data, due to the fact that range-based data is inherently spatial (i.e. range features are defined over a region rather than at a point or pixel, and therefore range-based classification cannot be performed in a pixel-wise manner.). For the following two classifier fusion methods, the necessary patch-wise conversion is applied to the outputs of low level classifiers by averaging the classifier confidences over the pixels in the patch.

$$P_{patch}(x_i \mid y_j) = \frac{1}{N}\sum_{k=1}^{N} P_k(x_i \mid y_j) \qquad (17)$$

Here, the likelihood of a patch belonging to class x_i given the visual features y_j is the average likelihood over N pixels within that patch. For the data fusion, a conversion is applied at the data level; therefore, RGB and HDV values are averaged to assign a single feature value for a terrain patch.

3.2 Bayesian Fusion

Bayesian classifier fusion has been proposed for classification of natural scenes with promising results (Manduchi, 1999; Shi and Manduchi, 2003). It is a simple approach to merge the results of probabilistic classifiers assuming that the visual features are conditionally independent. This fusion method ignores the class assignment decision made by low level classifiers and operates only on the computed class likelihoods. Bayesian Fusion can be represented by this relation:

$$P(x_i \mid y_1, \ldots, y_n) = \prod_{j=1}^{j=n} P(x_i \mid y_j) \qquad (18)$$

Here the probability of a patch belonging to class x_i given all visual features $\{y_1 \ldots y_n\}$ is the product of probabilities observed via individual visual features. Then, the class assignment decision is based on *maximum a posteriori* as given in Equation 6.

However, the above formulation implicitly requires that all classifiers function in the same class space. When the set *{x_i}* is not identical for all classifiers, Equation 18 is no longer valid and the merged model is defined as:

$$P(z_k \mid y_1, \ldots, y_2) = \prod_{i,j} P(x_i \mid y_j) \tag{19}$$

Here set *{z_k}* is the new class space after fusion. Since the new class space is formed by the Cartesian product of the low level classifier spaces, the fusion model ends up having a large number of non-physical terrain classes. Although previous researchers addressed this problem with a dimensionality reduction algorithm (Manduchi 1999), this method did not exploit physical class knowledge that could be inherited from supervised classifiers.

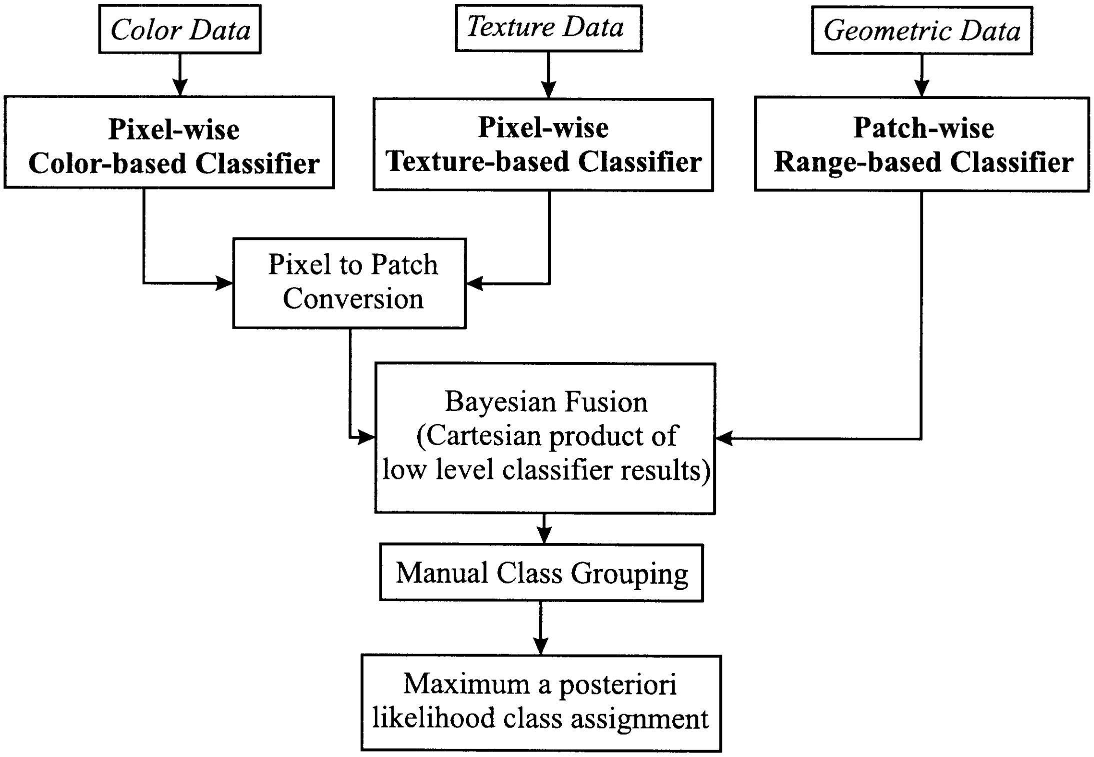
Color Data
Texture Data
Geometric Data
Pixel-wise Color-based Classifier
Pixel-wise Texture-based Classifier
Patch-wise Range-based Classifier
Pixel to Patch Conversion
Bayesian Fusion (Cartesian product of low level classifier results)
Manual Class Grouping
Maximum a posteriori likelihood class assignment

Figure 3.1: Flowchart for Bayesian Fusion approach

In his algorithm, Manduchi (1999) defined a metric called descriptiveness, which was a measure of information that a class model has for the segmented image. A dimensionality reduction algorithm merged the two least descriptive classes and continued iteration until a sufficient number of classes remained. This algorithm was presented for merging a supervised color classifier and an unsupervised texture classifier. However, all the low level classifiers defined in Section 2 are supervised classifiers. In this work, the fusion class space was manually grouped into a lower dimensional space of physically meaningful terrain classes based on physical class knowledge.

3.3 Meta-Classifier Fusion

The second classifier fusion method studied in this work is meta-classifier fusion. It utilizes the low level classifiers' outputs as "fusion features" to train a higher-level classifier. As illustrated in Figure 3.2, meta-classifier fusion is a patch-wise classifier and requires the construction of a new feature space based on the outputs of low level classifiers. Specifically, the combined feature is composed of the continuous class likelihood outputs of the low level classifiers. Equation 20 shows the output of a low level classifier and Equation 21 describes how multiple low level classifier outputs are combined as a single vector.

$$[low-level\ output] \ = P_k[y] = \begin{bmatrix} P(x_1 \mid y) \\ \vdots \\ P(x_k \mid y) \end{bmatrix}; \ \sum_i^k P(x_i \mid y) = 1 \tag{20}$$

$$[fusion\ feature] \ = \begin{bmatrix} P_{k-1}[y_1] \\ \vdots \\ P_{k-1}[y_n] \end{bmatrix} \tag{21}$$

Here y_i is the index for a low level classifier corresponding to an observable feature, and k is the total number of detectable classes for the low level classifier y_i. For instance, x low level classifiers with y detectable classes each generate $x*(y-1)$-dimensional feature vectors for the fusion level. In this work, for MER data, low level classifiers had two detectable classes, and the fusion vector was 3-dimensional. For other experimental data, this color-based classifier had 3 detectable classes while the other classifiers had two classes, thus a 4-dimensional feature vector was generated for meta-classifier fusion.

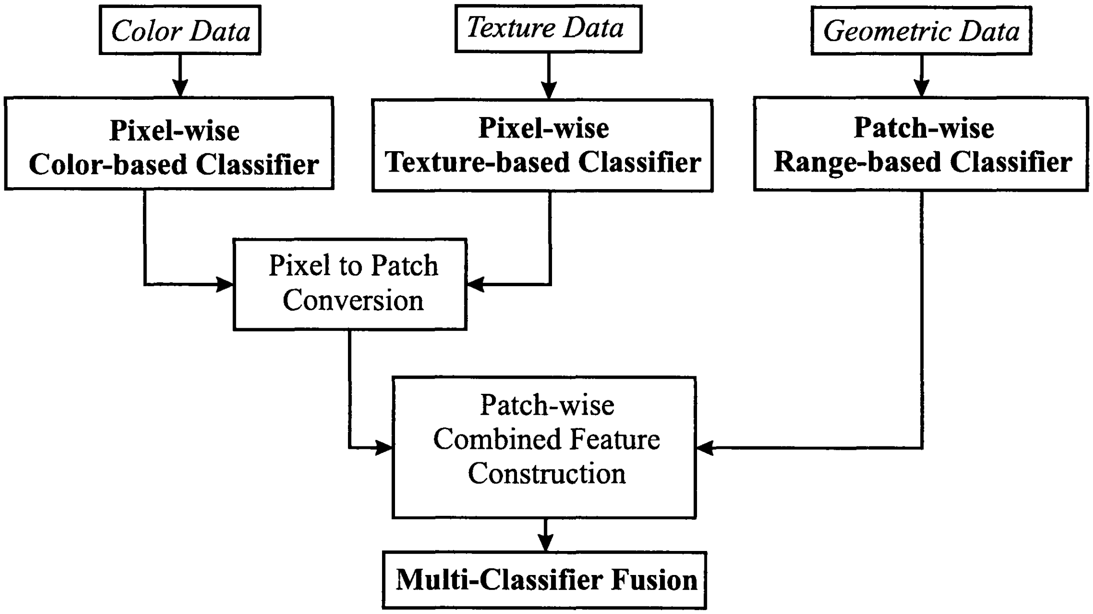
Color Data
Texture Data
Geometric Data
Pixel-wise Color-based Classifier
Pixel-wise Texture-based Classifier
Patch-wise Range-based Classifier
Pixel to Patch Conversion
Patch-wise Combined Feature Construction
Multi-Classifier Fusion

Figure 3.2: Flowchart for meta-classifier fusion approach

After the feature extraction process described above, meta-classifier fusion classifies the feature vectors into class labels. This classification utilizes the same Mixture of Gaussians classifier structure described in Section 2.2. Here, however, the mixture model describes the distribution of the low level classifier responses of a terrain patch.

3.4 Data Fusion

Data fusion was used as a baseline to compare the performances of the classifier fusion methods. This approach combines features from different sensing modes into a single feature vector and performs classification in this (high dimensional) feature space. In this thesis, the data fusion feature vector was constructed by appending the RGB vector of the color data, the HDV vector of the texture data, and the three-dimensional range vector. Figure 3.3 illustrates the algorithm structure.

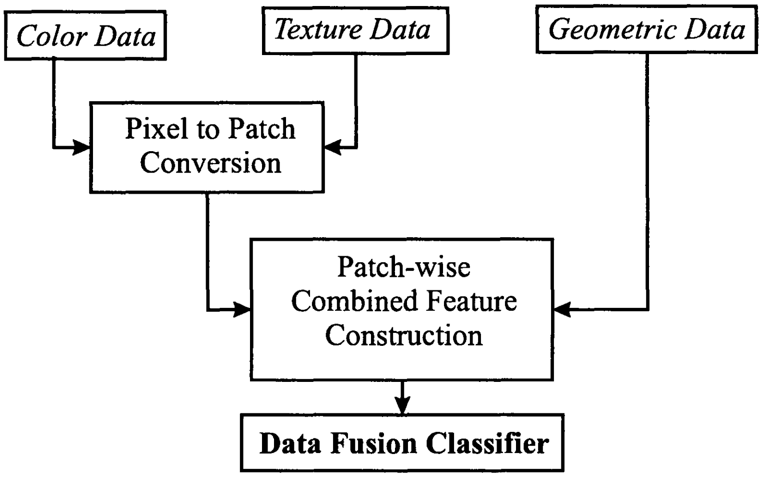
Color Data
Texture Data
Geometric Data
Pixel to Patch Conversion
Patch-wise Combined Feature Construction
Data Fusion Classifier

Figure 3.3: Flowchart data fusion approach

As described in Section 3.1.1, pixel-wise features of color and texture data are converted to patch-wise features since geometric data cannot be represented in a pixel-wise manner. For grid patches with *N* pixels, a patch-wise feature is computed as the mean vector:

$$c = \begin{bmatrix} \overline{R} \\ \overline{G} \\ \overline{B} \end{bmatrix} \quad t = \begin{bmatrix} \overline{H} \\ \overline{D} \\ \overline{V} \end{bmatrix} \tag{22}$$

$$c = \frac{1}{N}\sum_{i}^{N} c_i \quad , \quad t = \frac{1}{N}\sum_{i}^{N} t_i \tag{23}$$

Here c and t represent the patch-wise color and texture feature vectors, respectively. Then the data fusion vector becomes:

$$[Data\ fusion\ vector] = [\alpha \quad \beta \quad \hat{z} \quad \overline{R} \quad \overline{G} \quad \overline{B} \quad \overline{H} \quad \overline{D} \quad \overline{V}]^T \qquad (24)$$

A drawback of this approach and a motivation to investigate other fusion methods is that the statistical properties of different sensing modes are not consistent, and modeling the variance in a high dimensional feature space, even with higher number of Gaussian components, will likely not result in accurate approximations.

Chapter 4: Experimental Results

4.1 Introduction

The performances of the classifiers described in Chapters 2 and 3 were tested on two different image test sets. The first test set consists of images from NASA's Mars Exploration Rovers mission. High-resolution multi-spectral panoramic camera images were used to verify algorithm performances on Mars surface scenes. The second set of images was collected through a four-wheeled test-bed rover in an outdoor Mars-like environment. Detailed information on these images and performance analyses of the classifiers on these two sets are presented in this chapter.

4.2 MER Imagery

The performance of the low level classifiers and the fusion methods were studied on publicly available images from the MER mission's Spirit and Opportunity rovers. Fifty representative images were selected from the Mars Analysts' Notebook Database (2006); details about the image sources are given in Appendix C. Ten images were used for training of the supervised classifiers and the remaining forty images were used to evaluate algorithm accuracy and computation time. A sample image is shown in Figure 2.7 and Figure 4.1.

For Mars surface scenes, three primary terrain types that are believed to posses distinct traversability characteristics were defined: rocky terrain, composed of outcrop or large rocks; sandy terrain, composed of loose drift material and possibly crusty material; and mixed regions, composed of small loose rocks partially buried or lying atop of a layer of sandy terrain. Examples of these terrains are shown in Figure 4.1

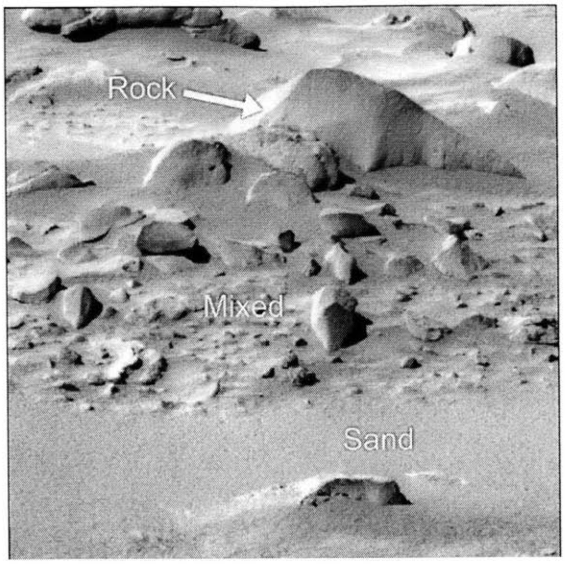

Figure 4.1: Terrain classes for Mars

4.2.1 Ground Truth

Ground truth images are used to quantify classifier accuracy by comparing classifier class assignments to the true class labels. They are also utilized for selecting representative data points during training.

Ground truth images are generated by manually labeling the pixels in a given image. However, since classifiers operate in both pixel-wise and patch-wise manners, labeling image pixels is not sufficient. A patch-wise ground truth image was created automatically by combining the manually labeled image (pixel-wise) with the range data of the corresponding scene. Range data was used to assign each pixel to a patch, and every patch was assigned to a class label according to the following rule:

$$class\ label = \begin{cases} \text{rock} & \text{if} \quad r/N > .25 \\ \text{mixed} & \text{if} \quad (m+r)/N > .25 \\ \text{unknown} & \text{if} \quad u/N > .25 \\ \text{sand} & \text{if} \quad \text{none of the above} \end{cases} \tag{25}$$

Here lower case letters *r, m,* and *u* represent the number of pixels in the patch labeled as rock, mixed and unknown, respectively, and *N* is the total number of pixels. The order of the conditional checks in Equation 25 also represents the priority list for class assignment.

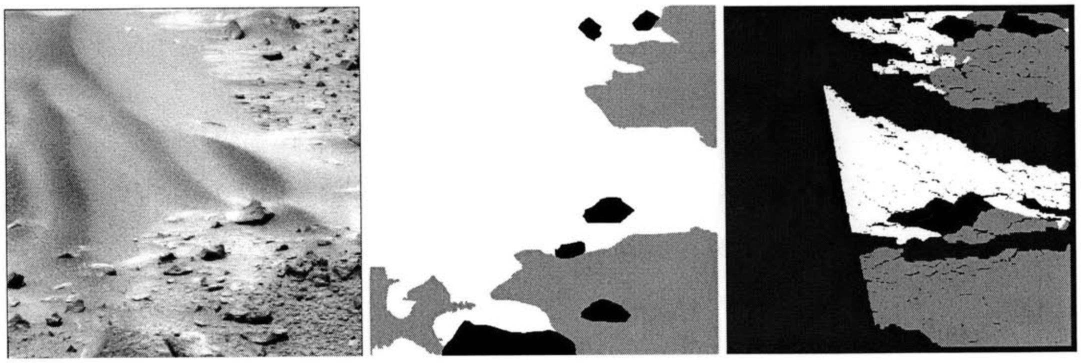

Figure 4.2: Mars scene (left), hand labeled ground truth (middle), auto-generated patch-wise ground truth (right). Black label represents rock class, white label represents sand class, light gray label represents mixed class, and dark grey label represents unknown regions.

Figure 4.2 shows a sample Mars scene recorded from Spirit's left panoramic camera, a corresponding hand-labeled ground truth image in the middle, and the right image is the patch-wise ground truth image. Note that the left side of the patch-wise ground truth image is labeled "unknown" since this part of the image is out of the field of view of the stereo pair, and thus geometric properties are unobservable. In the results presented here, portions of the images that did not yield stereo data were not classified since they would not yield direct comparisons of color, texture, and range classifier performance.

4.2.2 Color-based Classification

The MER panoramic camera pair has eight filters per camera, with the left filters distributed mostly in the visible spectrum and the right camera filters located in the infrared spectrum (with the exception of filter R1 at 430 nm). For color feature extraction, a combination of images from the left camera filters was used to generate a near-true color image. The 4th filter (601 nm), 5th filter (535 nm), and 6th filter (492 nm) intensities were chosen since they are near to the red, green, and blue wavelengths respectively.

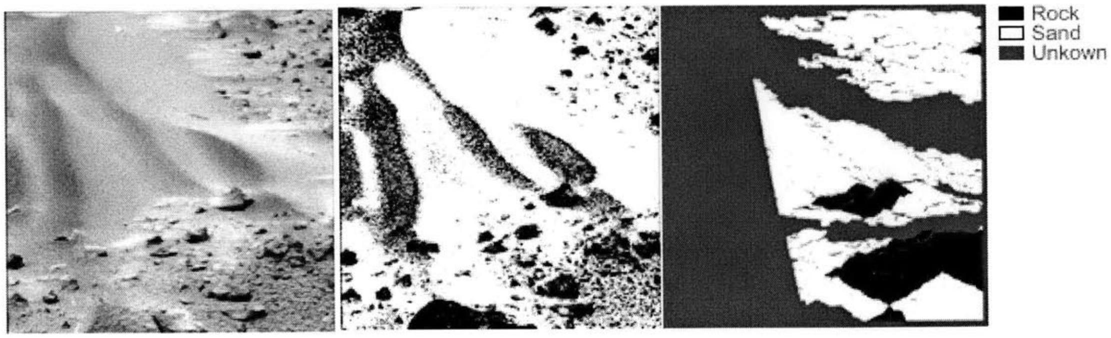

Figure 4.3: Sample color-based classification: Original scene (left), pixel-wise color classifier (middle), conversion to patch-wise (right)

Figure 4.3 shows two representations of a result from the color-based classifier. The left image is the original Mars scene while the middle image shows an example of pixel-wise class assignment (i.e. each individual pixel was assigned to a class), and the right image shows an example of patch-wise class assignment (i.e. each grid cell was assigned to a class). It should be noted that for color-based classification, classes of interest are “rock” and “sand” terrain classes. The “mixed” terrain class is not represented since this class is essentially a mixture of “rock” and “sand” classes and does not have a unique pixel-wise color signature.

In general, the color-based classification had difficulty distinguishing between “rock” and “sand” classes due to the small variation of color information between the classes. It was observed that most Mars scenes exhibit a uniform brownish-red tone, where rocks generally appear darker than sandy plains. However, as mentioned previously, color-based features are highly dependent on illumination, and therefore when the scene exhibited bright or dark regions this method resulted in high error rates. As presented in Figure 4.4, the color-based classifier produced results close to expectation of random choice between two classes on average, 57%.

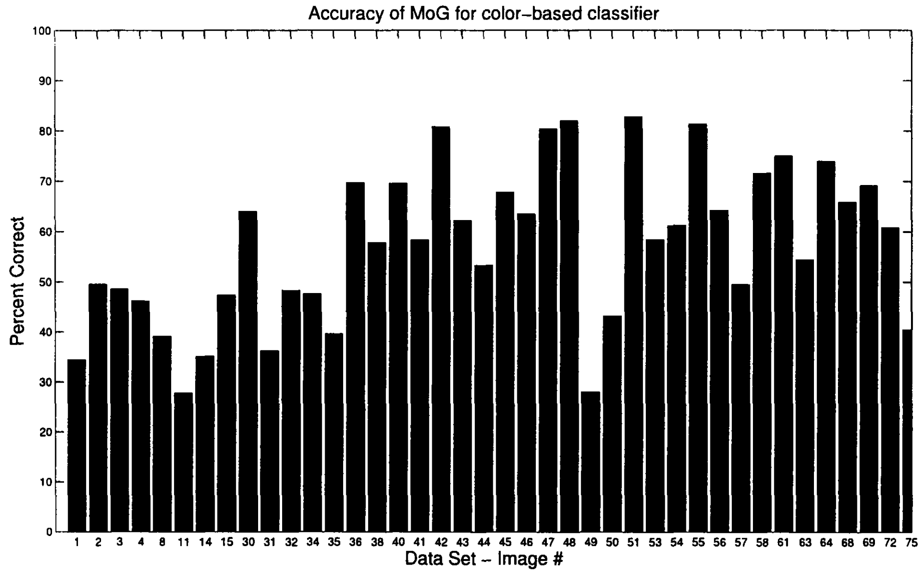

Figure 4.4: Color-based classification results on MER imagery

Similar results were also obtained when classification was applied through a single filter image only. As shown in Table 4-1, classification of images from the left eye filters were studied individually, in order to detect the direction of highest variation in the visible spectrum. It can be seen that there is little variation in all available filters which did not allow accurate classification.

Table 4-1: Grayscale classification results for left eye filters

Filter #	Accuracy (%)
L2	45
L3	46
L4	51
L5	46
L6	42

Using IR data for color-based classification

It is also possible to use right eye filters (which are out of the visible spectrum) for color-based classification. As shown in Figure 4.5, classification results using right-eye filters have qualitatively different characteristics in terms of rock detection. Although overall accuracy was close to the results obtained from classifying visible spectrum data, it was observed that infrared data allows more robust results for dust-covered rock regions. Although merging visible and infrared data seems to be a promising approach, current MER data did not allow such fusion since a direct relation between left and right eye images was not easily available. This relation, however, could potentially be extracted through stereo processing.

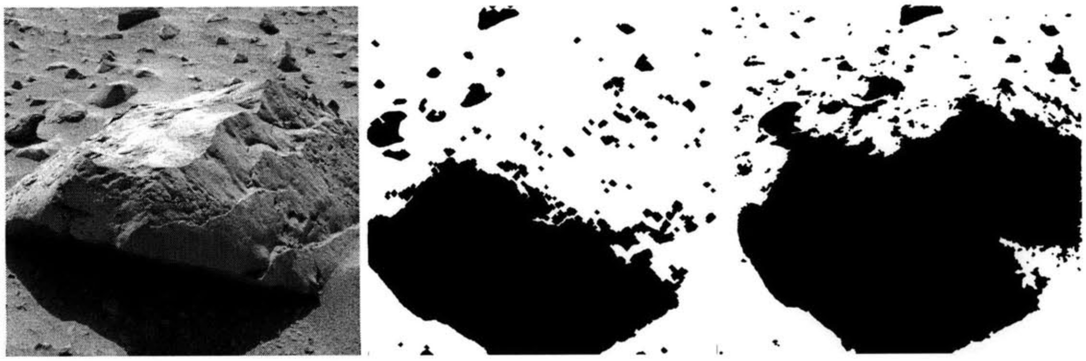

Figure 4.5: Original scene (left), Classification results: left eye filter combination (middle), right eye filter combination (right). Black label represents rock and white label represents sand class.

To summarize, color-based classification, although it produces accurate results for terrestrial applications, was not found to perform sufficiently robustly for Mars environments even with a wide range of spectral data available. On the other hand, potential future work lies in merging information from different wavelengths (including from the thermal range) to improve color-based results.

4.2.3 Texture-based Classification

Texture feature extraction operates on a single intensity image. For MER imagery, the 2nd filter of the left camera (753 nm) was used. For texture classification, the classes of interest are "smooth" and "rough" classes which correspond to sand and mixed terrain classes in the Mars scenes. The physical "rock" class presented both smooth and rough surfaces, therefore it was not selected as a unique texture-based class.

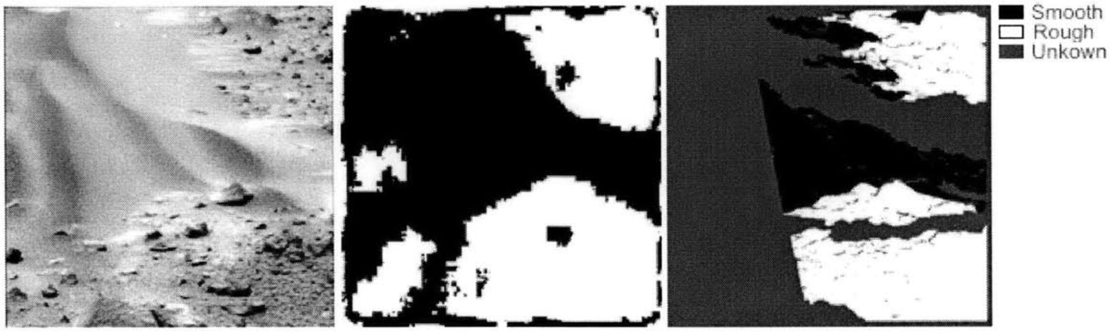

Figure 4.6: Sample texture-based classification: Original scene (left), pixel-wise texture classifier (middle), conversion to patch-wise (right)

Figure 4.6 shows two representations of the texture classifier. As in Figure 4.4, the left image is the original scene while the middle image shows the pixel-wise classifier result and the right image shows the patch-wise result.

The texture-based classifier performed better than the color classifier with an average accuracy of 60%, as seen in Figure 4.7. This accuracy, however, is insufficient for use as a stand-alone classifier in most applications. The major drawback of texture-based classification was the range dependence described in Section 2.4.2: for images that spanned large distances, texture-based classification did not produce robust results. In particular for features at close range (i.e. 2 to 4 meters) this sensing mode exhibited very poor results.

To overcome this problem, two methods for introducing distance as a part of texture feature were studied. The simplest way to introduce a distance metric into texture features was to augment the texture feature vector with the distance from the rover to the feature position in Cartesian space. A more complicated method maintained the original texture feature vector, but trained different classifiers for use in different ranges. These variations were found to result in 2%-4% performance improvement, and therefore increasing the overall classifier complexity was considered undesirable.

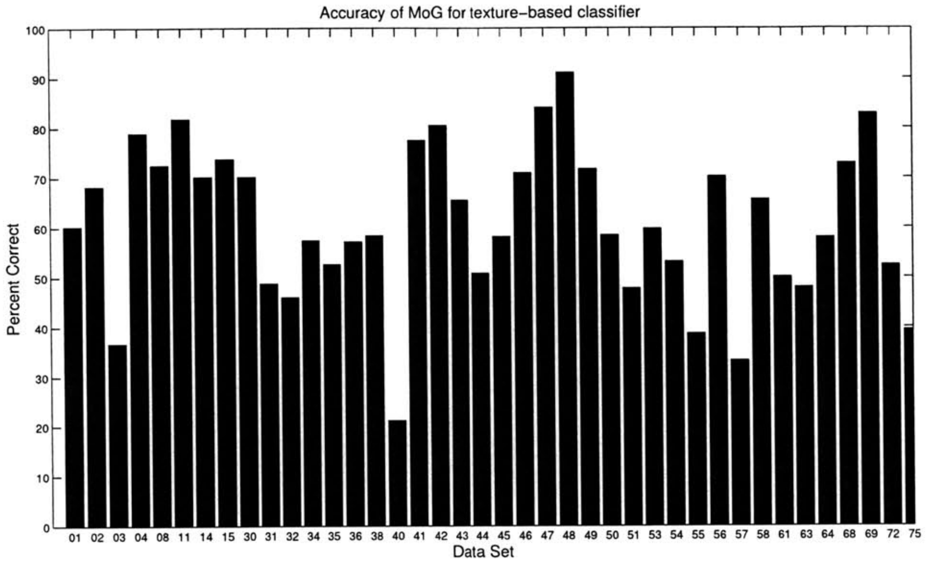

Figure 4.7: Texture-based classifier results on MER imagery

4.2.4 Range-based Classification

Range data was extracted by processing stereo pair images using the stereo libraries developed at NASA/JPL (Ansar, et al., 2004). The stereo software supplied by JPL was treated as a black box where inputs were the left and right image pair and corresponding camera model parameters. Search window sizes and disparity ranges were defined. The output of the package was a binary file with Cartesian coordinates of every pixel. Details on the usage of the stereo software package, including parameter selection, are presented in Appendix C.

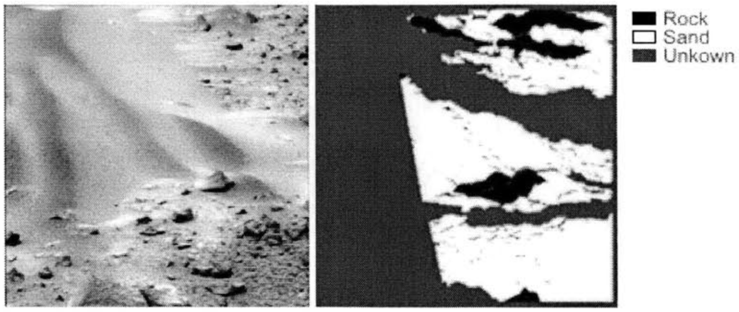

Figure 4.8: Sample range-based classification: Original scene (left), range-based classification (left)

Figure 4.8 shows a sample scene and classification results for the range-based classifier. As explained in Section 2.5, range-based classification is possible only for a portion of the image where range data exists. Classes of interest were "rock" and "sand" terrain classes. Although range-based classification presented the highest mean accuracy among low level classifiers with an average accuracy of 75% (see Figure 4.9), there were certain terrain features that range data did not reliably classify. In particular, sloped sand in the form of ridges and dunes were systematically confused with rock. In other areas, low, flat outcrops were difficult to distinguish from surrounding sandy terrain.

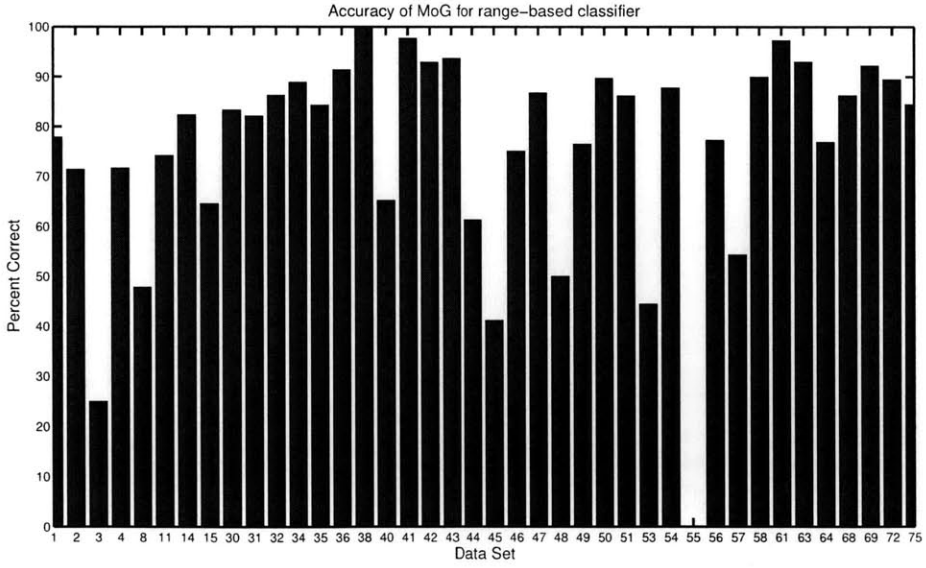

Figure 4.9: Range-based classification result on MER imagery

4.2.5 Classifier Fusion Results

As described in Section 3.1, classifier fusion methods generate final class decisions based on the low level classifier results. Here, image regions that possessed valid stereo data were classified and the rest of the image was considered to be unknown. The color- and range-based classifiers distinguish rock and sand terrain classes while the texture-based classifier distinguishes the mixed class. Merging the results of these classifiers allows to generation of a three class terrain map for the Mars scenes.

As discussed in Section 3.2, final class assignment for Bayesian fusion method requires a manual grouping of the non-physical classes into physically meaningful groups. This grouping in the case of MER imagery was formed as:

$$P(R \mid c,t,r) = P(R \mid c) * P(R \mid r) * P(Sm \mid t) \tag{26}$$

$$P(S \mid c,t,r) = P(S \mid c) * P(S \mid r) * P(Sm \mid t) \tag{27}$$

$$P(M \mid c,t,r) = P(R \mid c) * P(R \mid r) * P(Ro \mid t) + P(S \mid c) * P(S \mid r) * P(Ro \mid t) \tag{28}$$

Here *R, S, M* represents the final terrain classes “rock”, “sand”, and “mixed” respectively, and *c, t, r* represents the observed visual features color, texture, and range respectively. *Ro* and *Sm* refer to the texture-based classes of “rough” and “smooth”.

These relations explicitly encode physical knowledge in the final class decisions. Rocky terrain was generally observed to be darker than sandy or mixed terrain, with pronounced vertical features, and often appeared relatively smooth. Low level classifier outputs related to these attributes were thus mapped to the final “rock” terrain class. Sandy terrain was generally observed to be smooth and low-lying relative to the ground plane. Mixed terrain exhibited uneven geometric properties and very rough textural properties at the scale of interest.

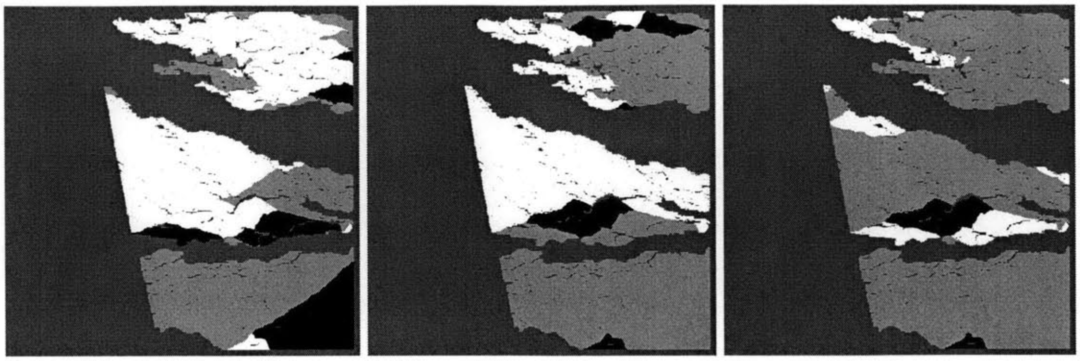

Figure 4.10: Resulting images for fusion methods: Data fusion (left), Bayesian fusion (middle), meta-classifier fusion (right). Black label represents rock class, white label represents sand class, light gray label represents mixed class and dark gray label represents unknown regions.

Figure 4.10 shows sample resulting images for the same Mars scene shown in Figure 4.3. Figure 4.11 shows ROC curves for each classifier, illustrating the accuracy of the classifiers across a range of confidence thresholds. As described (in Sections 4.2.2, 4.2.3, and 4.2.4) previously, low level classifiers performed relatively poorly. The numerical results for all the images are included in Table 4-2.

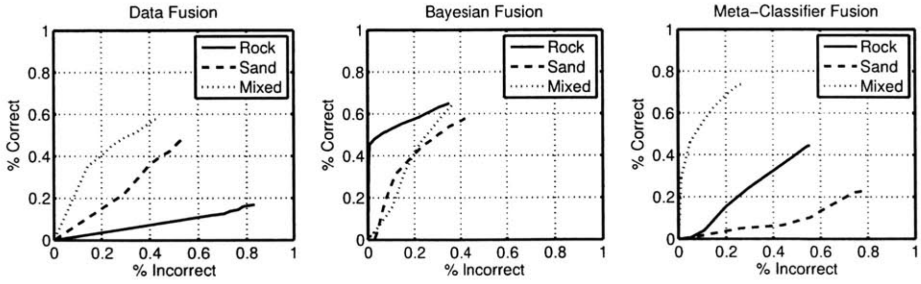

Figure 4.11: ROC curve for classifier methods applied on MER Imagery

The fusion methods improved overall classification performance in two ways. First, classifier fusion yielded a more descriptive class set than any of the low level classifiers (i.e. there were two detectable classes for each low level classifier but three for each fusion result.). Second, classification accuracy increased for rock and sand class assignments. It was observed that classifier fusion allowed the color- and range-based classifiers to compensate for each other's weaknesses and resulted in a better performance for detecting "rock" and "sand" classes.

Comparing classifier fusion methods, it was observed that Bayesian and meta-classifier fusion performed more accurately than data fusion. This result was expected due to the complexity of fitting a distribution in the high dimensional data fusion space. However, meta-classifier fusion lacks the robustness of Bayesian fusion, likely due to an insufficient amount of training data. Since meta-classifier is trained in the patch-wise manner, the number of training points per image is in the order of tens, whereas pixel-wise training employs at least thousand of data points per image. Therefore, it is expected that meta-classifier fusion requires a higher number of training images to perform comparably to Bayesian fusion.

Table 4-2: Classifier results for MER imagery

Image #	Low Level Classifier Accuracy (%)			High Level Classifier Accuracy (%)		
	Color	Texture	Range	Data F.	Bayesian	Meta-C.
1	34.4	60.2	77.9	31.4	63.5	54.7
2	49.6	68.2	71.4	36.8	63.8	56.5
3	48.6	36.7	25.0	0.0	83.1	70.9
4	46.2	78.9	71.7	64.2	81.8	76.0
8	39.1	72.5	47.8	9.2	81.2	76.1
11	27.8	81.8	74.2	42.1	66.4	46.8
14	35.1	70.2	82.4	19.8	62.9	42.6
15	47.3	73.8	64.5	21.7	81.1	72.5
30	64.0	70.2	83.3	54.0	71.4	23.4
31	36.2	48.8	82.1	60.6	55.1	56.3
32	48.2	45.9	86.2	43.1	59.5	68.2
34	47.5	57.4	88.9	42.7	34.8	34.7
35	39.6	52.6	84.3	60.7	62.2	62.4
36	69.7	57.2	91.4	56.5	61.4	50.1
38	57.8	58.4	100.0	44.7	73.5	49.2
40	69.6	21.3	65.2	33.7	40.2	75.2
41	58.2	77.5	97.7	64.1	74.6	44.5
42	80.7	80.5	92.9	30.7	88.6	67.5
43	62.2	65.5	93.7	12.4	70.7	67.4
44	53.3	50.8	61.3	42.4	53.7	44.2
45	67.7	58.2	41.2	38.7	66.9	69.0
46	63.5	71.0	75.1	32.9	73.8	62.5
47	80.4	84.1	86.8	18.7	67.5	65.6
48	81.9	91.1	50.0	22.6	82.0	80.5
49	27.9	71.8	76.5	43.0	56.2	58.0
50	43.1	58.5	89.7	55.0	50.5	39.4
51	82.8	47.8	86.2	25.0	77.4	60.0
53	58.4	59.8	44.4	45.2	39.2	42.8
54	61.1	53.2	87.8	35.2	75.8	66.1
55	81.3	38.7	0.0	12.0	20.6	58.0
56	64.1	70.3	77.3	75.0	92.4	54.7
57	49.5	33.3	54.3	27.4	44.0	56.0
58	71.6	65.7	90.0	34.9	56.6	68.5
61	75.0	50.1	97.3	8.0	61.5	62.8
63	54.4	48.0	93.0	44.4	65.9	73.3
64	74.0	58.0	76.9	45.9	72.7	36.4
68	65.8	73.0	86.3	55.0	84.4	75.9
69	69.0	82.9	92.2	22.4	53.4	65.3
72	60.7	52.4	89.5	42.0	51.2	55.0
75	40.5	39.4	84.5	65.0	67.7	65.4
Mean	57.2	60.9	75.5	38.0	64.7	60.0
Std. Dev.	15.6	15.6	21.2	17.8	15.5	13.3

4.3 Wingaersheek Beach Experiments

Additional experiments were performed using a four wheeled mobile robot developed at MIT, named TORTOISE (all-Terrain Outdoor Rover Test-bed for Integrated Sensing Experiments), shown in Figure 4.12. Details on TORTOISE and its sensor suite are presented in Appendix D.

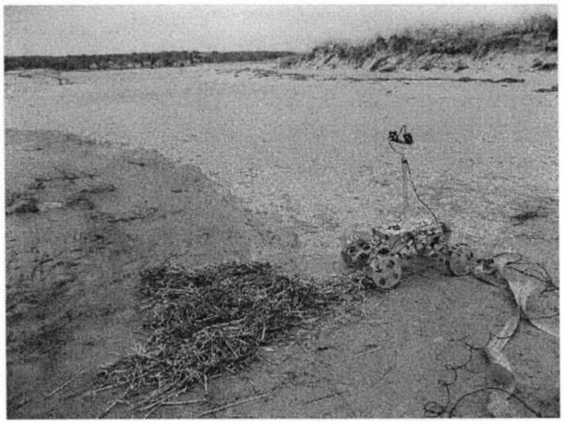

Figure 4.12: TORTOISE starting a multi-terrain traverse at Wingaersheek Beach

Experiments were performed at Wingaersheek Beach in Gloucester, MA. As seen on Figure 4.13, this is an ocean-front environment that has regions dominated by large rock outcrops and scattered rocks over sand. Neighboring areas exhibit sloped sand dunes and sandy flats mixed with beach grass. Figure 4.14 shows a typical scene from the experiment site. This scene shows a large rock in the foreground and scattered, partially buried rocks in the mid range. Sand appears in grayish color while rock features vary from gray to light brown and dark brown. This test site was chosen because of its visual and topographical similarities to the Mars surface. For the following experiments, the terrain classes of interest are "rock", "sand" and "beach grass".

Six days of experiments were conducted, with a total of approximately 50 traverses and a total distance traveled of 500 meters. Performances of the low and high level classifiers were studied with the images recorded by TORTOISE. Results are shown in Figure 4.15 as a series of ROC curves. Note that the sand class line lies on the vertical axis for the data fusion and Bayesian fusion plots.

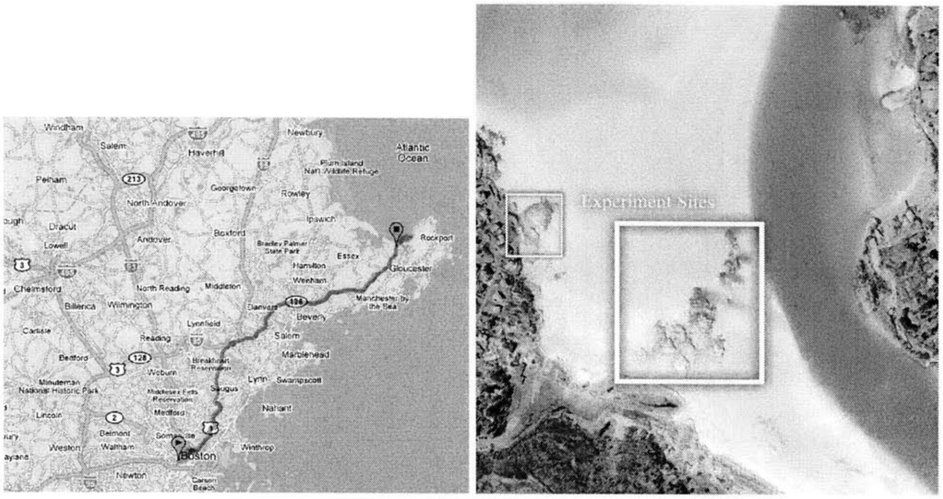

Figure 4.13: Wingaersheek Beach, Transportation map from Boston (left), satellite image of experiment site (right)

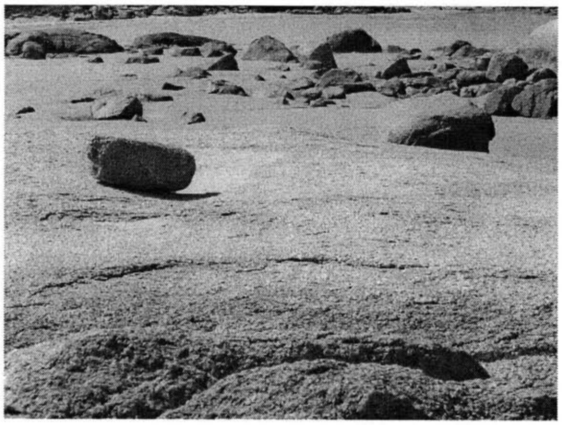

Figure 4.14: A scene from Wingaersheek Beach

Performance of the low level classifiers was relatively poor. Compared to experiments on MER imagery, it was observed that the color classifier performance was improved, no doubt due to the greater color variation present in an average beach scene. Relatively poor results were observed from the geometry-based classifier. The reason for this decrease in performance is likely due to poor accuracy and resolution of stereo-based range data for these experiments relative to MER imagery data, which used state-of-the-art JPL stereo processing software operating on high-quality images. This performance decline points to the sensitivity of range-based classification to data quality, and strengthens the motivation for classifier fusion.

Among classifier fusion methods, Bayesian fusion was found to exhibit better performance than meta-classifier fusion or data fusion. Meta-classifier fusion was biased towards the "sand" class, resulting in poor accuracies for "rock" class assignments. Data fusion yielded poor results for beach grass but relatively accurate results for other classes. As with MER imagery, classifier fusion in general performed better than low level classifiers by providing greater accuracies and more descriptive class assignment.

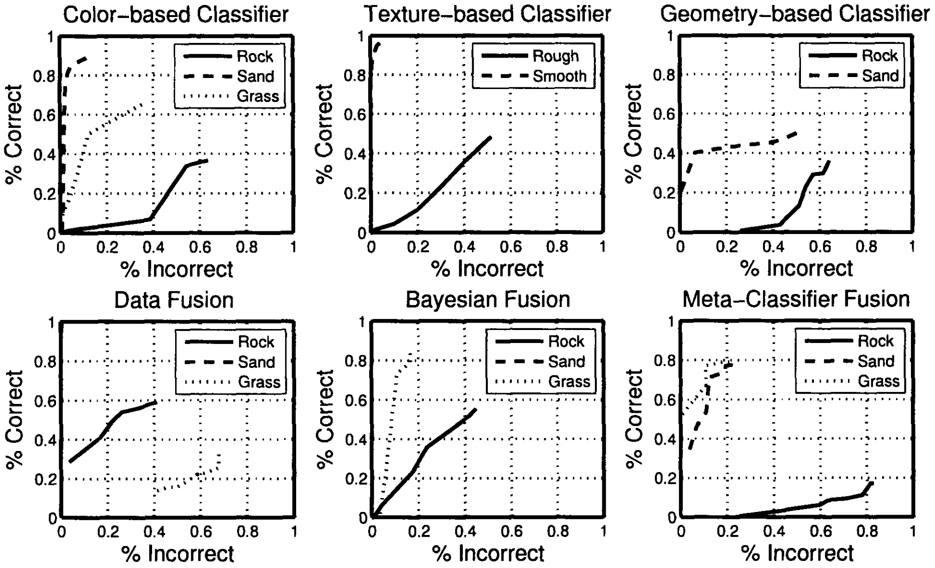

Figure 4.15: ROC curves for classifiers methods applied on the Beach images

4.4 Extensions to Vision-based Classification

Classifier fusion approaches can be applied in two distinct scenarios. The first method is when multiple distinct classifiers yield class assignments for a certain region. This method has been presented earlier in this thesis. Here the low level classifiers' class assignments for a terrain patch are merged to yield more robust results. A second potential fusion scenario is when a single classifier produces multiple class assignments for a certain region. This second case occurs, for example, when a classifier generates different results for the same terrain patch when viewed from different angles.

For the data set collected at Wingaersheek Beach, the rover captured images while traversing a straight line, so successive images contain overlapping terrain regions. If the observed features are subject to a change over time (such as from illumination and shadowing) the class assignments at different times would be different, and thus a fusion method may be necessary to produce consistent class decisions throughout the traverse.

4.4.1 Successive Frame Fusion

Classification results from successive, closely-spaced (spatially) images can potentially be merged to generate more robust class estimates than those from a single frame. This relies on accurate frame-to-frame spatial registration of images. Ideally, GPS or algorithms such as visual odometry can be used for rover localization and position registration. This problem is outside the scope of this thesis. Here experiments were performed in which the rover traversed a straight line at constant velocity and the captured images were time-stamped. Accurate registration was assumed through the use of dead reckoning. This was practical due to the relatively short traverses in the beach experiments used to study successive frame fusion. Using this approach, stereo data of every frame is converted to the inertial frame from the rover-fixed frame via appropriate rotation and translation matrices.

Three methods for successive frame fusion were investigated:

Majority Voting Here, a voting scheme merges the most recent classification result with previous classifier decisions. At every time interval, classifier fusion assigns a class label to every grid patch on the ground plane. This assignment is interpreted as a positive vote of that class for the given patch in that time interval, as shown in Equation 29, where w is the vote from the patch j for class C_i, and $D_{j,t}$ is the classifier decision at time t. The terrain class with the maximum votes W_j (see Equation 30) is assigned as the final class label at that time t. In the case of ties, the most recent class assignment is given precedence. This method results in crisp terrain class labels and does not preserve class likelihood information from the low level classifiers.

$$w_{j,t}(C_i) = \begin{Bmatrix} 1; & if & D_{j,t} = C_i \\ 0; & if & D_{j,t} \neq C_i \end{Bmatrix} \tag{29}$$

$$W_j(C_i) = \sum_{t=0}^{T} w_{j,t}(C_i) \tag{30}$$

Weighted Voting This approach is similar to majority voting in the sense that class assignment decisions are based on voting. However, here votes are scaled based on various properties. One weighting coefficient that was investigated was the distance of the grid cell relative to the rover. As shown in Equation 31, a positive vote for a cell is scaled by the distance (x_{rover} - $x_{feature}$). This method relies on the assumption that classifier outputs derived from close-range cells are likely to be more accurate than those derived from distant cells, since texture and range feature quality degrades as distance increases.

$$w_{j,t}(C_i) = \begin{Bmatrix} \dfrac{1}{\left|x_{rover}(t) - x_{feature}(t)\right|}; & if & D_{j,t} = C_i \\ 0 \quad ; & if & D_{j,t} \neq C_i \end{Bmatrix} \tag{31}$$

Another weighting method that was investigated was based on the class likelihood output from the classifier fusion algorithm as shown in Equation 32, where $P(C_i|y)$ is the posterior probability of class C_i given the observed features, y. This method emphasizes the confidence level of the class assignments.

$$w_{j,t}(C_i) = \begin{cases} P(C_i \mid y); & if \quad D_{j,t} = C_i \\ 0 \quad ; & if \quad D_{j,t} \neq C_i \end{cases} \tag{32}$$

Confidence Method This approach uses the total probability theorem assuming that the class decisions at different time intervals are disjoint. Equation 34 computes the combination of class decisions from different times, using the total probability theorem. The first term represents the confidence of the classifier model on the observed feature, resulting in higher weighting coefficients for the decisions that are based on well defined features in the MoG model. The second term is the conditional likelihood of the terrain class given the observation, the output of the MoG classifier at time *t*, as in Equation 33.

$$P(C_i \mid y) = \frac{f(y \mid C_i)P(C_i)}{\sum_i f(y \mid C_i)P(C_i)} \tag{33}$$

$$P(\overline{C_i}) = \sum_t^T P(y_t)P(C_i \mid y_t) \tag{34}$$

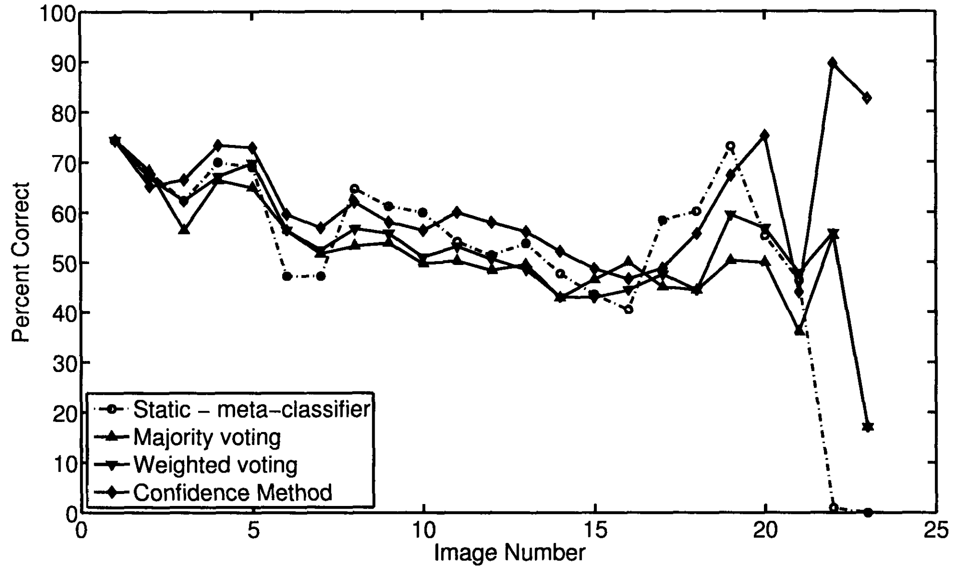

Figure 4.16: Successive frame fusion results

The performance of these algorithms was studied using five rover traverses from the beach data set described above. For a single traverse, a series of images from the TORTOISE mast-mounted stereo pair was recorded as the rover moved in a straight line at constant speed. For these simple traverses, rover position estimation was highly accurate and thus accurate spatial image registration was possible.

Figure 4.16 shows accuracy results of successive frame fusion during a single experiment. Note that the figure shows the distance-based Weighted Voting algorithm. Accuracies of every fifth image are plotted with respect to image number, which corresponds to the time sequence of the traverse.

These results suggest that successive frame fusion decreases variation in the average classifier accuracy, and thus yields improved robustness to environmental changes. For example, toward the end of the traverse data presented here, TORTOISE's view angle was modified by due a sharp negative pitch while traversing a rock and classification performance decreased drastically. Successive frame fusion allowed the inheritance of previously observed features to "smooth" classifier output, thereby compensating partially for environment effects. Table 4-3 shows the average accuracy and standard deviation for each classifier for full-frame classification

Table 4-3: Successive frame fusion statistics

Fusion Method	Average Accuracy (%)	Std. Deviation (%)
Static frames	49.3	21.3
Majority voting	51.3	11.5
Weighted voting	53.3	11.7
Confidence method	62.1	11.6

4.5 Summary and Conclusions

Experimental analyses of low level classifier performances and classifier fusion methods have been presented in this chapter. It has been shown that low level classifier performance was not robust when applied to challenging outdoor settings such as planetary surfaces. Classifier fusion methods show improvement over low level classifier results in the form of more descriptive class assignments and improved accuracies.

Another scenario for classifier fusion was introduced to update class assignments for terrain patches as new decisions become available from the terrain classifier. It has been shown that merging class assignment information for a certain terrain patch from different time intervals increases classification robustness, since inheriting previous measurements decreases sensitivity to physical changes in the environment.

Chapter 5: CONCLUSION

5.1 Contributions of This Thesis

Knowledge of the physical properties of terrain surrounding a planetary exploration rover can be used to allow a rover system to fully exploit its mobility capabilities. The ability to detect or estimate terrain physical properties would allow a rover to intelligently predict its mobility performance and thereby autonomously avoid terrain regions that are potentially non-traversable. Knowledge of terrain properties could also allow a system to adapt its control and planning strategies to enhance performance, by maximizing wheel traction or minimizing power consumption. Therefore, there is a need to identify terrain classes and predict physical properties to achieve better rover performance. This thesis has focused on utilizing remote sensing methods for detecting surrounding terrain classes on planetary surfaces.

Low level classifiers were presented that utilize single visual features only. A Mixture of Gaussians based classifier was discussed for these low level classifiers. Classification performance was evaluated on MER mission images as well as experimental data sets collected in a Mars-like environment. Although these sensing modes can perform well for terrestrial applications, features on planetary surfaces do not present sufficient variation for these classifiers to perform accurately.

Fusion methods were proposed to improve the performance of the low level classifiers. Two techniques for merging the results of the low level classifiers were presented that rely on Bayesian fusion and meta-classifier fusion. A data fusion approach was presented to compare the classifier fusion results. It was shown that classifier fusion yielded terrain class assignments that were typically more accurate and descriptive than those from low level classifiers. In addition, a method for successive frame fusion was introduced. Successive frame fusion is a potential method to improve classifier decisions when visual data of a single terrain patch is available from different angles or time instants. This fusion method aims to increase classifiers robustness to physical changes in the environment such as occlusions, shadowing and illumination changes. It was observed that fusing information from different time instants improved classification accuracy compared to the results obtained from static frames.

5.2 Suggested Future Work

Future work related to remote sensing could investigate alternative classifier architectures (such as support vector machines) to yield better classification accuracies. Decreasing computation time and required processing power by means of alternative algorithms is also important.

Another promising avenue would be to associate local sensing with remote sensing to relate terrain traversability properties to the distant terrain. As presented in the motivations section, estimating mobility performance of a rover on a distant terrain is crucial. Achieving this prediction through analyzing physical properties of near terrain is an important step. This analysis could be done using visual and tactile sensing methods such as wheel sinkage measurement (Brooks, 2004) and wheel vibration analysis (Brooks and Iagnemma, 2005). This will enable improved prediction of the physical properties of distant terrain, and lead to generation of safe, feasible and traversable routes.

REFERENCES

Addison, P.S. (2002). *The illustrated wavelet transform handbook.* Bristol: Institute of Physics Publishing.

Ansar, A., Castano, A., and Matthies, L. (2004, September). *Enhanced real-time stereo using bilateral filtering.* 2nd International Symposium on 3D Data Processing, Visualization, and Transmission.

Avedisyan, A., Wettergreen, D.,Fong, T., and Baur, C. (2004). *Far-field terrain evaluation using geometric and toposemantic vision.* 8th ESA Workshop on Advanced Space Technologies for Robotics and Automation.

Bellutta, P., Manduchi, R., Matthies, L., Owens, K. and Rankin,K. (2000, October). *Terrain perception for Demo III.* Proceedings of the Intelligent Vehicles Symposium.

Biesiadecki, J. J., and Maimone, W.M. (2006). *The Mars Exploration Rover Surface Mobility Flight Software:Driving Ambition,* Proc. of IEEE Aerospace Conference.

Bilmes, J. (1997). *A Gentle Tutorial on the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models.* Technical Report, University of Berkeley

Bishop, C.M., (1995). *Neural networks for pattern recognition.* New York: Oxford University Press.

Brooks, C. (2004). *Terrain identification methods for planetary exploration rovers.* Master's thesis. MIT

Brooks, C. and Iagnemma, K. (2005). *Vibration-based Terrain Classification for Planetary Rovers.* IEEE Transactions on Robotics, 21, 6, 1185-1191.

Castano, R., et al. (2005). *Current Results from a Rover Science Data Analysis System.* Proceedings of 2005 IEEE Aerospace Conference, Big Sky.

Crotty, D. (2006*). Mars Exploration Rover multispectral color imagery.* Retrieved May 21, 2006 at http://www.lyle.org/~markoff/.

Dima, C.S., Vandapel, N., and Hebert, M. (2003). *Sensor and classifier fusion for outdoor obstacle detection: an application of data fusion to autonomous road detection.* Applied Imagery Pattern Recognition Workshop, 255- 262.

Dima, C.S., Vandapel, N., and Hebert, M. (2004). *Classifier fuiosn for outdoor obstacle detection.* ICRA'04.

Espinal, F., Huntsberger, T.L., Jawerth, B., and Kubota T. (1998). *Wavelet-based fractal signature analysis for automatic target recognition.* Optical Engineering, Special Section on Advances in Pattern Recognition, 37(1), 166-174.

Goldberg, S., Maimone, M., and Matthies, L. (2002). *Stereo vision and rover navigation software for planetary exploration.* IEEE Aerospace Conference, Big Sky.

Gor,V., Castaño, R., Manduchi, R., Anderson, R., and E. Mjolsness (2001). *Autonomous Rock Detection for Mars Terrain.* Space 2001, AIAA.

Iagnemma, K. and Dubowsky, S. (2002, March). *Terrain estimation for high speed rough terrain autonomous vehicle navigation.* Proceedings of the SPIE Conference on Unmanned Ground Vehicle Technology IV,

Jolly, M.-P.D.; Gupta, A. (1996). *Color and texture fusion: application to aerial image segmentation and GIS updating*. Applications of Computer Vision, WACV'96. 2-7.

Kelly, A., et al. (2004, June). *Toward Reliable Off-Road Autonomous Vehicle Operating in Challenging Environments.* International Symposium on Experimental Robotics.

Manduchi, R. (1999). *Bayesian Fusion of Color and Texture Segmentations.* In the Proc. of Int. Conf. on Computer Vision.

Manduchi, R, and Portilla, J. (1999). *Independent component analysis of textures.* In the Proc. of Int. Conf. on Computer Vision.

Manduchi, R. (2004). *Learning outdoor color classification from just one training image.* In the Proc. of European Conf. on Computer Vision, 4.

Manduchi, R., Castano, A., Thalukder, A., and Matthies, L. (2005, May). *Obstacle detection and terrain classification for autonomous off-road navigation.* Autonomous Robots.

Mars Analyst's Notebook (2006). Retrieved May 24, 2006, from http://anserver1.eprsl.wustl.edu/.

McGuire, P. C. *et al*, (2005). *The Cyborg Astrobiologist: scouting red beds for uncommon features with geological significance.* International Journal of Astrobiology, 4: 101-113.

Mojsilovic, A., Popovic, M.W., Rackov, D.M., (2000). *On the selectopn of an optimal wavelet basis for Texture characterization,* IEEE Trnsactions on Image Processing, 4, 12

NASA/JPL, (2006). *Mars Exploration Rovers Mission* Retrieved at July 27, 2006 from http://marsrovers.jpl.nasa.gov/home/index.html

Rajpoot, K.M., and Rajpoot, N.M. (2004). *Wavelets and support vector machines for texture classification,* In Proceedings 8th IEEE International Multitopic Conference.

Randen, T., and Husoy, J.H. (1998). *Filtering for texture classification: A comparative study,* IEEE Transactions of Pattern Analysis and Machine Inteligence, 21, 4

Rasmussen, C., (2001, December). *Laser Range-, Color-, and Texture-based Classifiers for Segmenting Marginal Roads.* Proc. Conf. On Computer Vision & Pattern Recognition Technical Sketches.

Reed, R.T., and Buf, J.M.H, (1993). *A review of recent texture segmentation techniques,* CVGIP: Image Understanding, 57, 359-372

Shi, X., and Manduchi, R. (2003). *A Study on Bayes Feature Fusion for Image Classification.* Proceedings of the IEEE Workshop on Statistical Algorithms for Computer Vision.

Smyth, P. (1996). *Clustering using Monte-Carlo cross validation*, Proc. of Int. Conf. on Knowledge Discovery and Data Mining, 126-133.

Squyres, S. W., et al, (2003). *Athena Mars rover science investigation.* J. Geophys. Res., 108(E12), 8062, doi:10.1029/2003JE002121.

Thompson, D. R., Niekum, S., Smith, T. and Wettergreen, D. (2005). *Automatic Detection and Classification of Features of Geologic Interest.* Proc. of IEEE Aerospace

Urquhart, M. and Gulick, V (2003). *Lander detection and identification of hydrothermal deposits*, abstract presented at First Landing Site Workshop for MER.

Vandapel, N., Huber, D.F., Kapuria, A., Hebert, M. (2004). *Natural Terrain Classification using 3-D Ladar Data.* Proceedings of the International Conference on Robotics and Automation (ICRA'04), 5, 5117- 5122.

Videre Design (2006) (May 25), 2006 at http//www.videredesign.com/index.htm

Appendix A: EM Learning

Expectation maximization is a standard approach for estimating parameters of mixture models. It can be utilized to estimate the missing values in a data set as well as computing the parameters underlying the distribution of the data (Bilmes 1998). In (Bishop, 1995), expectation maximization is given as a solution to the maximum likelihood estimation problem of mixture models. An error function is defined to be minimized which corresponds to maximizing the logarithm of the likelihood, Γ:

$$E = -\ln\Gamma = -\sum_{n=1}^{N} \ln p(x^n) = -\sum_{n=1}^{N} \ln \sum_{j=1}^{M} \alpha_j p(x^n \mid j) \tag{A.1}$$

Here the summation over index j is the mixture model as in Equation 1. Then, an iterative procedure is introduced as:

$$E^{new} - E^{old} = -\sum_n \ln\left\{\frac{p^{new}(x^n)}{p^{old}(x^n)}\right\} \tag{A.2}$$

Here, superscripts refer to parameter values before (old) and after (new) the iteration. It is also possible to extend Equation A.2 as:

$$E^{new} - E^{old} = -\sum_n \ln\left\{\frac{\sum_j \alpha_j^{new} p^{new}(x^n \mid j)}{p^{old}(x^n)} \frac{P^{old}(j \mid x^n)}{P^{old}(j \mid x^n)}\right\} \tag{A.3}$$

Using the Jensen inequality which is given as:

$$\ln\left(\sum_j \lambda_j x_j\right) \geq \sum_j \lambda_j \ln(x_j); \quad \text{given} \quad \lambda_j > 0 \text{ and } \sum_j \lambda_j = 1 \tag{A.4}$$

updates Equation A.3 as:

$$E^{new} - E^{old} \leq -\sum_n \sum_j p^{old}(j \mid x^n) \ln\left\{\frac{\sum_j \alpha_j^{new} p^{new}(x^n \mid j)}{p^{old}(x^n) P^{old}(j \mid x^n)}\right\} \tag{A.5}$$

Here the purpose is to minimize E^{new} with respect to new parameters; therefore one should minimize the right hand side of Equation A.5. For the specific case of the Gaussian mixture model the derivative of the right hand side becomes:

$$\tilde{Q} = -\sum_{n}\sum_{j} P^{old}(j \mid x^n)\left\{\ln\alpha_j^{new} - d\ln\sigma_j^{new} - \frac{\left\|x^n - \mu_j^{new}\right\|^2}{2\left(\sigma_j^{new}\right)^2}\right\} + const \qquad \text{(A.6)}$$

Setting Equation A.6 to zero the update equations are identified as follows:

$$\alpha_j^{new} = \frac{1}{N}\sum_{n} P^{old}(j \mid x^n) \qquad \text{(A.7)}$$

$$\mu_j^{new} = \frac{\sum_{n} P^{old}(j \mid x^n)x}{\sum_{n} P^{old}(j \mid x^n)} \qquad \text{(A.8)}$$

$$\left(\sigma_j^{new}\right)^2 = \frac{1}{d}\frac{P^{old}(j \mid x^n)\left\|x^n - \mu_j^{new}\right\|^2}{\sum_{n} P^{old}(j \mid x^n)} \qquad \text{(A.9)}$$

For high dimensional spaces and full covariance matrices Equation A.9 is generalized as:

$$\Sigma_j^{new} = \frac{\sum_{n=1}^{N} P^{old}(j \mid x^n)(x^n - \mu_j^{new})(x^n - \mu_j^{new})^T}{\sum_{n=1}^{N} P(j \mid x^n)} \qquad \text{(A.10)}$$

Implementation follows these three update equations.

Pseudo Code:

- Initialize parameters (described in Section 2.2.1)
    - Choose equal values for the weighting coefficients of Gaussian modes. (i.e. $\frac{1}{n}$ for n number of Gaussians in the mixture model)

  - Precluster the data with K-means to identify the means of the Gaussians in the mixture model (i.e. segment the data into *n* clusters for *n* number of Gaussians in the mixture model)
  - Assign the initial covariance of the Gaussians as identity matrices.
- Compute the initial model error (E^{old}) based on initial parameter values.
  - Apply Equation A.1, where *N* is the number of pixels in the image and *M* is the number of Gaussian modes in the model.
- Compute new parameter values.
  - Apply Equations A.7,8,10
- Compute model error (E^{new}).
  - Apply Equation A.1 with new parameters
- Iterate until the new error increment is less than 1% of the old error (simply the local minima) or 20 maximum iterations is achived.

$$\left|E^{new} - E^{old}\right| < 0.01E^{old} \tag{A.11}$$

Appendix B: Wavelet Transform

This section presents a basic background for wavelet transforms and is intended to develop practical intuition into the use of this tool. Wavelet transformation is a signal analysis tool for aperiodic/nonstationary signals. The transformation is actually a decomposition of signals into frequency components where the locations of the frequency components are also preserved.

For example, the aperiodic continuous signal, shown in Figure B.1, when analyzed through Fourier transform, would yield only the frequency components within the signal. Another method called Gabor filters employs a windowed Fourier transform where the transform is evaluated in separate windows to detect frequency components in different locations. However a drawback of this approach is the lack of flexibility of the window size.

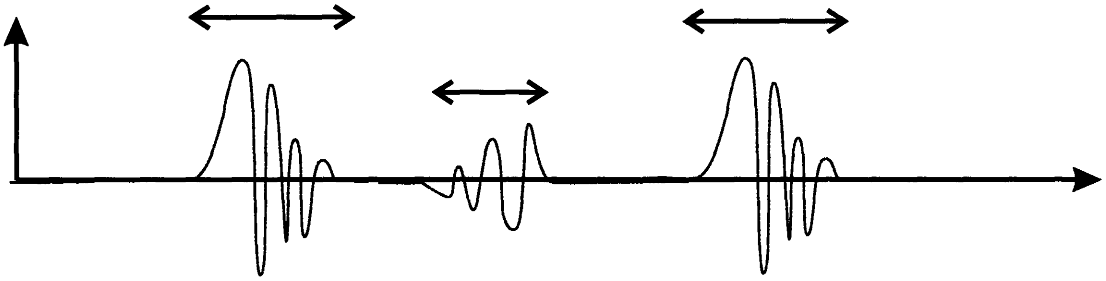

Figure B.1: A non stationary signal with different effective windows

To resolve this constraint on signal analysis, wavelet transform utilizes a window function, called a mother wavelet, that is defined by dilation (scaling) and location (translation) parameters. The general form of a "mother wavelet" is

$$\Psi_{a,b} = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right); \quad a \neq 0, \quad a,b \in R \tag{B.1}$$

Here the function ψ is the wavelet kernel which can take various form such as Mexican hat wavelet, Daubechies family wavelets etc. (Addison, 2002). Different applications utilize various kernels and there is no central model for selecting a kernel for specific applications. In this thesis for the texture feature extraction method the Daubechies family of wavelets was analyzed and D1 (also known as Haar wavelet) was selected.

The wavelet transformation is defined as:

$$W_{\Psi}[f]_{(a.b)} = \frac{1}{\sqrt{|a|}} \int_{-\infty}^{\infty} f(t)\psi(\frac{t-b}{a})dt \qquad \text{(B.2)}$$

It is a convolution of the window function with the signal. This convolution is repeated as a function of location and dilation so that wavelet analysis is able to detect the locations of different frequency components (see Figure B.2).

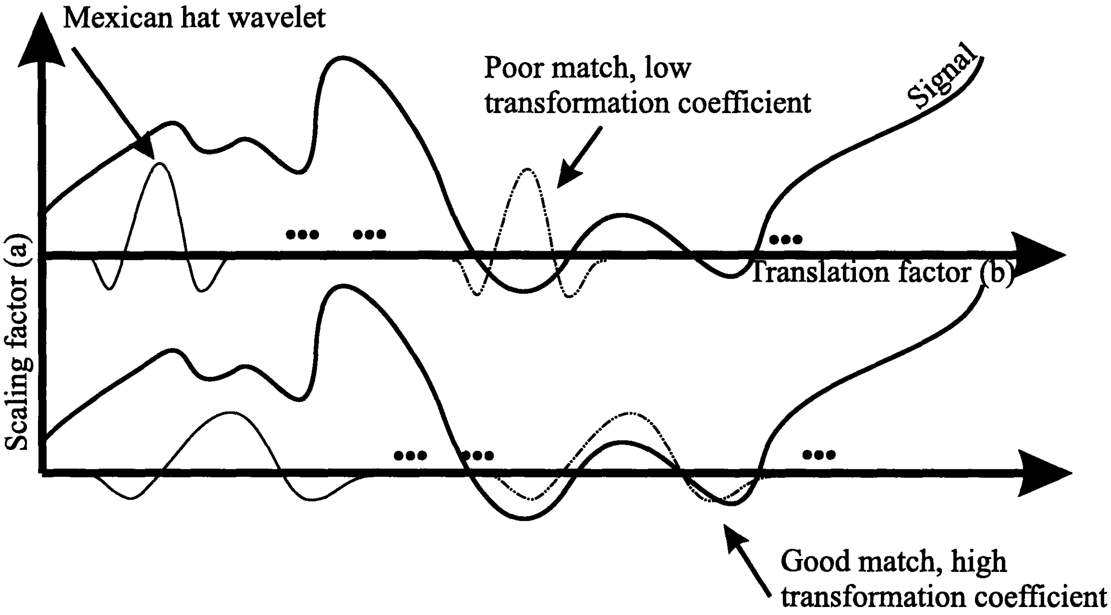
Mexican hat wavelet
Poor match, low transformation coefficient
Signal
Scaling factor (a)
Translation factor (b)
Good match, high transformation coefficient

Figure B.2: An example of wavelet transform processing on a continuous signal

In practice, the transformation scans the continuous signal with the mother wavelet with certain set of scaling parameters and represents the signal in a location vs.

scale map, where the scale axis represents the frequency components and the location axis represents the position of the corresponding component. The amplitude is a measure representing how good the signal matches with the mother wavelet at that particular location and scaling factor.

For image processing applications, this tool is utilized as a discrete transform of a finite discrete signal, which is composed of the pixel values of a row/column of an image. The transformation convolves the wavelet filter with the image pixel values. For simplicity, the 2D transformation is applied in horizontal and vertical directions separately using two 1D transforms. For instance, the filter coefficients for the Haar wavelet is [1 1] for the low-pass filter and [1 -1] for the high-pass filter. When applied on a row of pixel values such as [10 5 4 3 8 0 7 3], the first level of transformation is [7.5 3.5 4 5 2.5 0.5 4 2] where the first half of the row vector is the result of convolution with a low-pass filter and the second half is computed through convolving with a high-pass filter. The second level of transformation will be applied to the first half of the resulting vector, since that section includes the lower frequency contents.

In short, texture analysis is a method to identify variations in intensity where large variations result in highly textured surfaces while lower variation appears smoother. The purpose of using Haar wavelets is to identify regions of high and low variation and represent these regions through amplitudes of transformation coefficients.

Appendix C: MER Image Set

This appendix presents details on the MER image set used in algorithm validation for the vision-based classifiers. In this section, instructions on how to gather new image data or locate the original image data used in this thesis is presented. Details on PDS labels for identifying images are included. Also, a description of how to construct stereo pair and camera models and how to use JPL stereo software is included in this section.

Scientific and engineering data collected by Spirit and Opportunity, the Mars Exploration Rovers, are publicly available from <http://anserver1.eprsl.wustl.edu/>. At the moment, data collected during the first 630 sols are accessible through the web site. The ninth data release – data from sols 631-720 – is planned for July 27, 2006. New data releases generally occur every three months and 80 sols of data become available with every new release.

The Mars Analyst's Notebook allows users to browse through sols and view thumbnail images before downloading high resolution versions. The easiest way to search for full size panoramic camera (PanCam) images is to select either of the rovers on the main page and go to the "navigator" section. This section has a menu on the side bar that stores data sorted by instrument, which gives quick access to all PanCam imagery.

An alternative method to find representative Mars scenes is to browse through Crotty's (2006) web site where radiometrically corrected versions of the images are presented. One can browse through these color Mars scenes to identify representative images and then PDS labels can be used to find the original data with the Analyst's Notebook.

The PDS label is a unique labeling system for planetary data. For MER data, this label consists of 27 digits. The following description of PDS labels for MER data is adapted from NASA/JPL (July 10, 2006).

Digit #1: Denotes spacecraft id – see Table C-1

Table C-1: Spacecraft ID

Identifier	Spacecraft
1	Opportunity
2	Spirit

Digit #2: Denoted camera id – see Table C-2.

Table C-2: Camera ID

Identifier	Meaning
F	Forward HAZCAM
R	Rear HAZCAM
N	NAVCAM
P	PANCAM
M	Microscopic Imager
E	EDL-cam

Digits #3 to #11: Denotes the spacecraft clock. This is the number of seconds since January 1, 2000 at 11:58:55.816 UTC.

Digits #12 to #14: Denotes the product type. Product types are differentiated as having camera-induced distortion removed geometrically "linearized") or not removed (nominal), and as being thumbnail-sized or not. Four special flag characters follow (see Table C-3):

- Beginning with E - Type of EDR, which is raw with no camera model linearization or radiometric correction. If not beginning with E, then it is a Reduced Data Record (RDR).
- Ending with T - EDR or RDR that is thumbnail sized.

- Ending with L - If not beginning with E, denotes an RDR that is linearized, except for thumbnail sized RDRs.
- Ending N - If not beginning with E, denotes an RDR that is thumbnail-sized and linearized.

Table C-3: Valid values for MER Camera instrument input

Identifier	Data Product
EFF	Full frame EDR
ESF	Sub-frame EDR
EDN	Down-sampled EDR
ETH	Thumbnail EDR
ERS	Row Summed EDR
ECS	Column Summed EDR
ERP	Reference Pixels EDR
EHG	Histogram EDR

Digits #15 to #16: Denotes the site location count which indicates the rover's location where the image was acquired. Valid values include: 00, 01, 02,...99, A0, A1... A9, AA, AB, etc.

Digits #17 to #18: Denotes position-within-site count which indicates the rover's position within a site. Valid values include: 00, 01, 02,...99, A0, A1,..., A9, AA, AB...etc.

Digits #19 to #23: Denotes a group of images that were obtained within a single command sequence. Command sequence number consists of one letter and 4 numbers – see Table C-4 and Table C-5

Table C-4: Valid values for character (position 1) in field

Identifier	Command Sequence (Letter component)
C	Cruise
D	IDD & RAT
E	Engineering
F	Flight Software (Sequence rejected)
G	Spare
K	Spare
M	Master (Surface only)

N	In-Situ instrument (APXS, MB, MI)
P	PMA & Remote Sensing Instrument
R	Rover Driving
S	Submaster
T	Test
W	Sequence triggered by a Comm. Window
X	Contingency
Y	Spare
Z	SCM Sequence

Valid values for integers (unless the character in position 1 is P): 0000 thru 4095

Table C-5: Valid values for integers (position 2 thru 5) if P in character position 1

Identifier	Command Sequence (number component)
0000 through 0499	Misc. imaging setup/parm. Sequences
0500 through 0999	Unallocated, for possible future use
1000 through 1499	HAZCAM sequences
1500 through 1999	NAVCAM sequences
2000 through 2899	PANCAM sequences
2900 through 2999	MI sequences
3000 through 3999	Mini-TES sequences
4000 through 4095	Misc. PMA actuation sequences

Digit #24: Denotes camera eye – see Table C-6

Table C-6: Camera eye

Identifier	Meaning
R	Right
L	Left
M	Monoscoping (non-stereo)
N	Not applicable

Digit #25: Denotes the camera filter – see Table C-7 and Table C-8.

Table C-7: Valid values for PANCAM

Identifier	Left Camera	Right Camera
1	739nm (338nm bandpass)	436nm (37nm Short-pass)
2	753nm (20nm bandpass)	754nm (20nm bandpass)
3	673nm (16nm bandpass)	803nm (20nm bandpass)
4	601nm (17nm bandpass)	864nm (17nm bandpass)
5	535nm (20nm bandpass)	904nm (26nm bandpass)
6	482nm (30nm bandpass)	934nm (25nm bandpass)

7	432nm (32nm Short-pass)	1009nm (38nm Long-pass)
8	440nm (20) Solar ND 5.0	880nm (20) Solar ND 5.0

Table C-8: Valid values for Microscopic Imager "Filter" positions

Identifier	Filter position
1	MI window cover closed (500-700 nm response)
2	MI window cover open (400-700 nm response)

Digit #26: Denotes the product producer – see Table C-9.

Table C-9: Valid values for product producer

Identifier	Producer
A	Arizona State University
C	Cornell University
F	USGS at Flagstaff
J	Johannes Gutenberg Univ. (Germany)
M	MIPL (OPGS) at JPL
N	NASA Ames Research (L. Edwards)
P	Max Plank Institute (Germany)
S	SOAS at JPL
U	University of Arizona
V	SSV Team (E. De Jong) at JPL
X	Other

Digit #27: Denotes the product version number. Version identifier provides uniqueness for book keeping. Valid values: 1, 2, 3, ... 9, A, B, ... Z

As an example: A photo with the file name "2P126471340EDN0000P2303L5M1" would be from Spirit (2), taken by the left PANCAM through the 535nm (green) filter. By matching the spacecraft times (in this case 126471340) with other pictures that have a similar time, you can find sets that were taken at the same time of the day. Similar sets can be used to build stereograms and color composites. This picture was downsampled

(EDN) at site 0 (00) and position 0 (00) (On the lander in this case!). It was taken with command sequence P2303 for MIPL (OPGS) at JPL and is version 1.

Table C-10 gives PDS labels of the MER image collection used in this thesis along with the purpose of the image.

Table C-10: PDS labels of the image set used in this thesis

Referred image #	PDS label	Purpose
1	2P133201599EFF2224P2573L3C1	Test
2	2P133215675EFF2232P2571L3C1	Test
3	2P133474961EFF2232P2587L3M1	Test
4	2P134351518RAD2500P2536L2C1	Test
5	2P131863377EFF1200P2454L4C1	Training texture
8	2P134260970EFF2400P2534L2C1	Test
11	2P137021717EFF40D0P2412L2C1	Test
14	2P134970336EFF2700P2760L2C1	Test
15	2P134970703EFF2700P2760L2C1	Test
17	1P159876032EFF40ELP2598L2C1	Training range
18	2P157970579EFFA269P2592L2C1	Training range
21	2P127604277EFF0309P2543L2C1	Training color
23	2P131152910EFF1124P2579L2C1	Training texture
24	2P131954806EFF1300P2532L2C1	Training color
25	2P130014432EFF0514P2533L4C1	Training texture/color
26	2P134632976EFF2700P2540L2C1	Training texture
27	2P135241228EFF2702P2544L2C1	Training texture
30	2P152107503EFF8987P2542L2C1	Test
31	2P156723977EFFA201P2571L2C1	Test
32	2P156812738EFFA201P2572L2C1	Test
34	2P156817229EFFA201P2575L2C1	Test
35	2P157703334EFFA269P2590L2C1	Test
36	2P157703673EFFA269P2591L2C1	Test
38	2P158503999EFFA2AFP2531L2C1	Test
40	2P165606114EFFA928P2537L2C1	Test
41	2P129914742EFF0506P2398L2C1	Test
42	2P129915002EFF0506P2398L2C1	Test
43	2P130265292EFF0700P2543L2C1	Test
44	2P133106987EFF2200P2567L2C1	Test
45	2P133215948EFF2232P2571L2C1	Test
46	2P134984977EFF2700P2761L2M1	Test
47	2P140389803EFF6800P2591L2C1	Test
48	2P173773703ESFACAQP2536L2C1	Test
49	2P167114339EFFA9F4P2411L2C1	Test
50	2P168264740EFFA9I4P2564L2C1	Test

51	2P171816160EFFAAJCP2569L2C1	Test
53	2P168447258EFFAA04P2570L2C1	Test
54	2P171463735EFFAAJCP2558L2C1	Test
55	2P168886713ESFAAACP2576L2C1	Test
56	2P172349635EFFABAKP2577L2C1	Test
57	2P167116059EFFA9F4P2412L2C1	Test
58	2P172615014EFFABEJP2578L2C1	Test
61	2P166847115EFFA9DWP2550L2C1	Test
63	2P169242473EFFAAB2P2591L2C1	Test
64	2P169243516EFFAAB2P2592L2C1	Test
68	2P135679621ESF3000P2548L2C1	Test
69	2P131242200EFF1145P2589L2C1	Test
71	2P158414638ESFA2AFP2599L2C1	Training fusion
72	2P158503999EFFA2AFP2531L2C1	Test
74	2P133020015ESF2100P2564L2C1	Training range/fusion
75	2P157263098ESFA225P2581L2C1	Test
77	2P157349336ESFA225P2584L2C1	Training fusion

Stereo Processing of MER imagery

Stereo libraries developed for the MER mission was made available to MIT/FSRL by the NASA/JPL due to collaboration through the Mars Technology Program. Libraries were delivered as an executable package that required stereo pair images, corresponding camera models and computation specific parameters. The package then produced output files such as a binary file for coordinates of the point cloud, "result.ran," rectified versions of the stereo pair images, "l-rect.pic," and "r-rect.pic," and an image of the elevation map, "elev.ppm." A sample prompt line command is "./jpl_stereo -pyrlevel 1 -hwindowsize 19 -vwindowsize 19 -maxdisp 508 im89_L7.png im89_R1.png im89_l.cahvor im89_r.cahvor"

-pyrlevel: Refers to the down-sampling level. For a 1024 by 1024 image, -*pyrlevel 1* results in a 512 by 512 image for all output files. The zero level is not applicable for the MER image set and level one was maintained for all images to keep image sizes consistent. The trade off is losing resolution for more stereo coverage.

-Xwindowsize: Refers to horizontal and vertical window size in which the feature matching will take place. This setting should be an odd integer and define a square window. This setting directly affects the density of the computed stereo points. Normally, one should try values in the range [11 21] and choose the value that results in maximum coverage. Different images would require different values for the best stereo result.

-maxdisp: Refers to maximum disparity. Unless a minimum border is defined separately with *–mindip*, the disparity range is assumed to be [0 maxdisp#]. The valid disparity range is defined to be in the range of [0 254], but user should scale this range with the *pyrlevel#* setting as:

$$\max disp\# < 254 \times 2^{pyrlevel\#} \qquad (C.1)$$

Since *pyrlevel 1* is used as default in this work, *maxdisp# 504* is also used as default parameter value for all images.

Image pair: Next to the arguments are the file names of the stereo image pairs, left image and right image respectively. Images should be grayscale and in png format as stored in the PDS system.

For the stereo computation, L2 – R2 or L7 – R1 pairs are utilized since these pairs have optical filters close to each other. However, for generating rectified images for L4, L5, and L6 filters, the accompanying right eye filter is irrelevant, since image rectification is calculated through the camera model only and no matching between left and right images are done.

Camera model: The last two arguments are the corresponding camera models of the stereo pair images, left and right camera model respectively. JPL stereo libraries employ a *"cahvor"* camera model which is an abbreviation for six camera parameters:

center, axis, horizontal, vertical, optical, and *radial* that defines the camera to rover transformation of the coordinate system.

A cahvor camera model file is a text file that includes these six parameters along with five other parameters that are computed from cahvor vectors. Cahvor vectors are to be found in the PDS label of every image. This label can be accessed by clicking to the PDS tab in the "Product View Window" in Mars Analyst's Notebook. A PDS label consists of constants and parameter values for that specific product. In this work, a PDS label is only used to identify the camera parameters. The camera model parameters are given in the PDS label as:

```
/* CAMERA_MODEL DATA ELEMENTS */
GROUP        = GEOMETRIC_CAMERA_MODEL
CALIBRATION_SOURCE_ID       = "178"
^MODEL_DESC      = "GEOMETRIC_CM.TXT"
MODEL_TYPE       = CAHVOR
MODEL_COMPONENT_ID = ("C", "A", "H", "V", "O", "R")
MODEL_COMPONENT_NAME    = (CENTER, AXIS, HORIZONTAL,
        VERTICAL, OPTICAL, RADIAL)
MODEL_COMPONENT_1  = (0.321503, -0.0442795, -1.23716)
MODEL_COMPONENT_2  = (0.0232831, -0.964824, 0.261863)
MODEL_COMPONENT_3  = (3567.51, -422.512, 104.283)
MODEL_COMPONENT_4  = (23.4174, 451.151, 3563.0)
MODEL_COMPONENT_5  = (0.0247597, -0.96764, 0.251121)
MODEL_COMPONENT_6  = (0.022997, 0.024872, -0.407146)
FILTER_NAME       = PANCAM_L3_673NM
REFERENCE_COORD_SYSTEM_INDEX = (22, 24, 24, 65, 44)
REFERENCE_COORD_SYSTEM_NAME = ROVER_FRAME
END_GROUP = GEOMETRIC_CAMERA_MODEL
```

A required section is the six model components which are given as 3-dimensional vectors. They represent the locations or directions of the components in the rover frame. These values are plugged into the following equations:

$$H_s = \|A \times H\| \tag{C.2}$$

$$H_c = \langle A, H \rangle \tag{C.3}$$

$$V_s = \left\| A \times V \right\| \tag{C.4}$$

$$V_c = \langle A, V \rangle \tag{C.5}$$

$$\theta = \frac{\langle (V \times H), A \rangle}{\langle (A \times V), (A \times H) \rangle} \tag{C.6}$$

Then the camera model file format is as follows:

```
Dimensions = 1024 1024

C = 0.321503 -0.0442795 -1.23716
A = 0.0232831 -0.964824 0.261863
H = 3567.51 -422.512 104.283
V = 23.4174 451.151 3563.0
O = 0.0247597 -0.96764 0.251121
R = 0.022997 0.024872 -0.407146
S = 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
```

```
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000

Hs   =  3556.4265
Hc   =  518.0203
Vs   =  3556.7915
Vc   =  498.2818
Theta =  -1.5703

 S internal =
0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000
```

Here, dimensions are the horizontal and vertical pixel counts of the original image. *C, A, H, V, O,* and *R* vectors are copied from the PDS label. *S* is a null vector of size *9 by 36.* It is followed by horizontal and vertical components of the center and scale parameters along with the theta angle which are computed through Equations C.2-6. The last parameter is $S_{internal}$ which is a 5 by 5 null vector.

When the software is executed with these arguments, a process message is also printed on the screen during computation. This message also warns the user if a fault is detected with any of the arguments (i.e. if they are out of valid range or files cannot be read due to formatting error etc.). a final message includes the size of the range image (it will be down-sampled based on *pyrlevel#*) and what percent of the image has good range data. In this work, 35% is set to be a minimum limit for an image to be included in the collection. This value is also used to set *Xwindowsize* to the optimum value as explained above.

One of the outputs of the executable is the binary file that contains 3D coordinates of the pixel. The format of the file is designed to imitate the image where every pixel location contains three floating point numbers assigned *x, y,* and *z* coordinates. These values are used for range-based feature extraction as explained in Section 2.5.1.

Appendix D: TORTOISE EXPERIMENTAL ROVER

This appendix presents the details on TORTOISE (all-Terrain Outdoor Rover Test-bed for Integrated Sensing Experiments), the four wheeled test-bed rover of the FSRL. TORTOISE was designed to conduct wheel-terrain interaction experiments in outdoor settings during the 2002-2003 academic year. Later in the Spring'05, the wheel transmissions were updated with a 2:1 gear ratio instead of 1:1. Finally during Fall'05, sensory devices required for remote sensing experiments were added.

TORTOISE Main Design

This section is directly adapted from Appendix-D of Brooks (2004). Any changes that are made to the base system after Summer'04 are be noted.

The FSRL Technology Test-bed Rover (a.k.a. TORTOISE, for all-Terrain Outdoor Rover Test-bed for Integrated Sensing Experiments, see Figure D.1) is one of the test platforms available in the Field and Space Robotics Laboratory. It was specifically designed to study terrain interaction and sensing issues affecting planetary rovers.

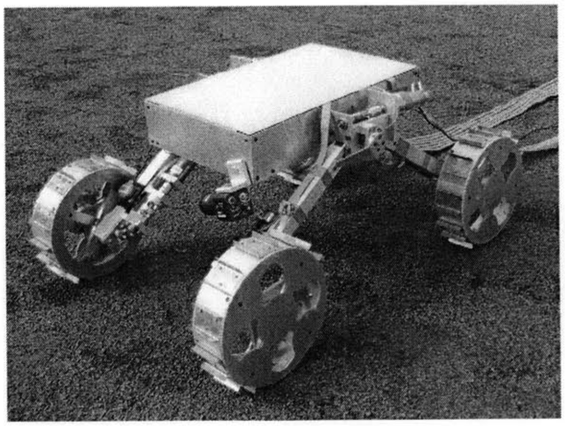

Figure D.1: Technology Test-bed Rover as of Summer'04

TORTOISE is a four-wheeled rover with an actively reconfigurable suspension. It is 80 cm long in its longest configuration, is 50 cm wide at its widest point, and has a mass of 12.2 kg. The angles between the two legs on each side can be controlled independently. The two shoulders are connected to the main body with a differential so that one can rotate while the other remains stationary. The main body maintains an angle midway between the two sides. For example, if the right pair of legs rotates 20° with respect to the left pair, the main body will rotate 10°. The direction of the rover is controlled using skid steering. Rover dimensions are summarized in Table D-1.

Table D-1: FSRL Technology Test-bed Rover dimensions

Dimensions	
Length	80 cm.
Width	50 cm.
Mass	12.2 kg.
Wheel diameter	20 cm.
Wheel width	5.1 cm.
Grouser length	1 cm.

The four wheels are made of rigid aluminum tubing. Each wheel is 20 cm in diameter and 5.1 cm wide, with 20 stainless steel grousers extending 1.0 cm from the surface of the wheel. The wheels are powered by 12-watt DC brush-type motors with 246:1 planetary gearboxes. The shoulder joints are powered by 10.5-watt DC brush-type motors, with 134:1 planetary gearboxes and a 20:1 worm/worm-gear pair. The motion of all six motors is sensed using magnetic encoders. Motor and transmission details are presented in Table D-2.

Table D-2: FSRL Technology Test-bed Rover motors and transmissions

	Motors and Transmissions	
Wheel Motor	12 Watt DC brush type	Faulhaber 2342S012CR
	Wheel Transmition	492:1 total
	2:1 bevel gear	Custom made
Shoulder Motor	10.5 Watt DC brush type	Faulhaber 2842S012C
Shoulder Transmission	268:1 total	Faulhaber 38/1 134:1
	20:1 worm gear	Custom made

The front right wheel of the rover (on the left in Figure D.1) is equipped with several sensors to study the terrain it is traversing. A 5.6 N-m torque sensor measures the torque the motor is applying to the wheel (see Figure D.2). A contact microphone is mounted to the leg of the rover near the front right wheel (see Figure D.3) for vibration sensing. Additionally, a color CCD camera with a 3.5mm-8.0mm variable focal lens is mounted to the rover body where it can maintain a view of the inside of the front right wheel.

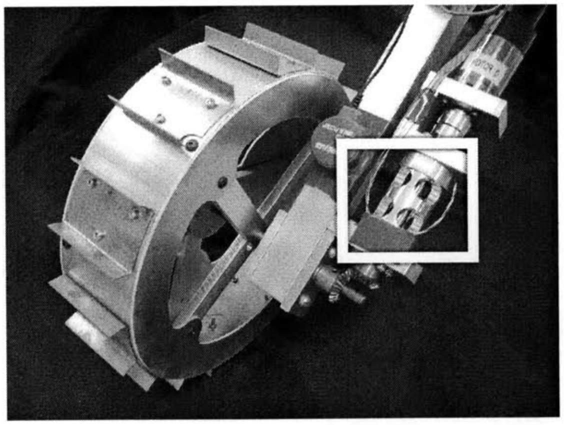

Figure D.2: Torque sensor mounted on TORTOISE

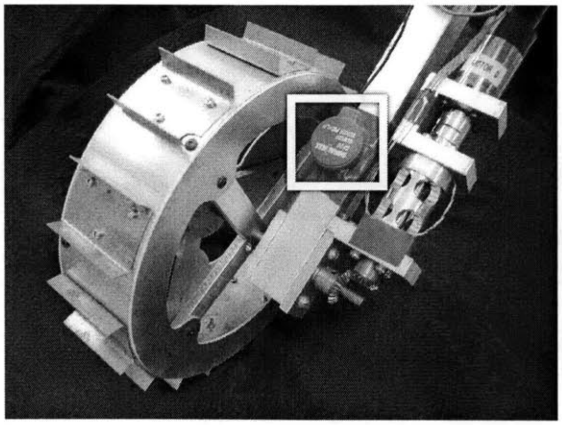

Figure D.3: Vibration sensor mounted on TORTOISE

The rover is also outfitted with sensors to fully estimate its configuration. A two-axis tilt sensor measures the pitch and roll of the rover body. The angles of the two shoulder joints are measured with potentiometers, as is the angle between the right shoulder and the body. Model numbers for all sensors are shown in Table D-3.

Table D-3: FSRL Technology Test-bed Rover sensors

	Sensors	
Motor rotation	Magnetic encoder	Faulhaber HEM2342S16
Torque	5.6 Nm. Torque sensor	Futek 5160
Vibration	Contact microphone	Signal Flex SF-20
Vision	1/3” CCD camera	Genwac GW-202B
	3.5 mm. – 8.00 mm. Lens	Edmund Optics NT55-255
Configuration	2 axis Tilt sensor	Crossbow CXTA02
	Potentiometers	Vishay/Spectrol 65700103

All control and data sampling is done off-board. Motor power is sent to the rover via a tether and sensory signals are returned the same way. Motor control is done using an off-board PC104 computer. Sampling of wheel torque and rover configuration is done by the same PC104 system. Image capture from the CCD camera and vibration signal recording is done off-board on a tethered laptop computer. See Figure D.4 for a schematic of the rover communications.

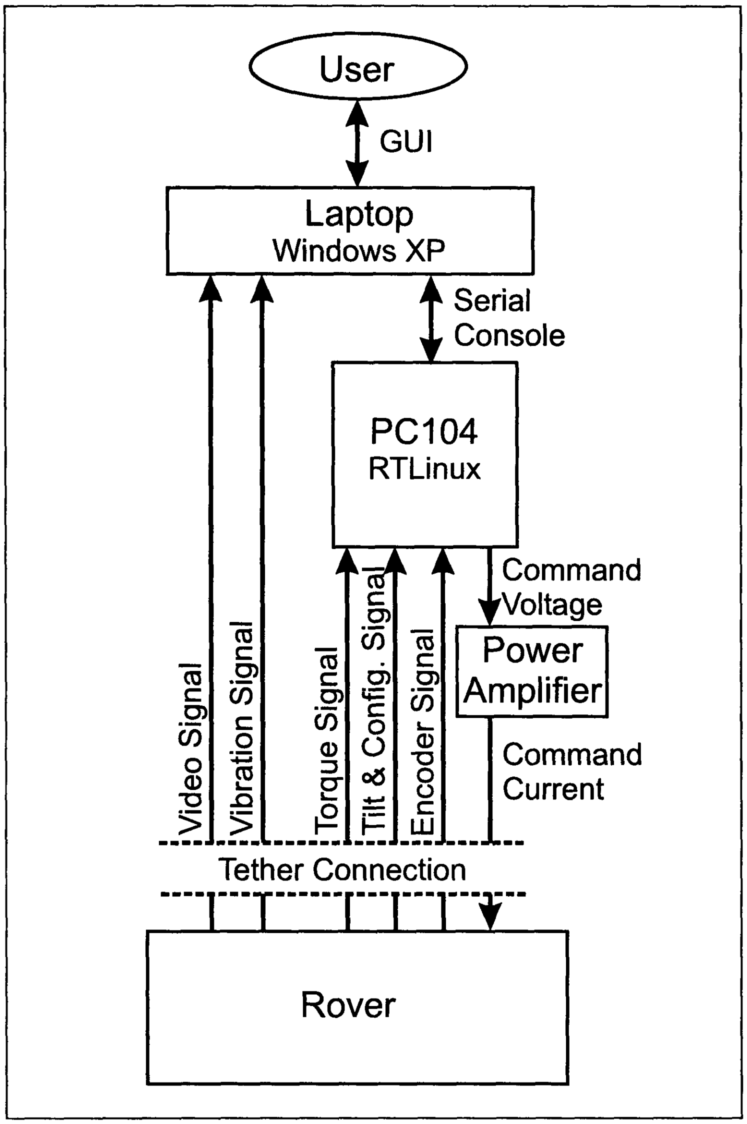

Figure D.4: Rover communications schematic

The PC104 system is a Diamond Systems Prometheus LC, a ZFx86 processor equivalent to a 486-100MHz. Analog signals, including signals from the tilt sensor, the torque sensor, and the potentiometers, are sensed using an analog input card. Encoder signals are received by a quadrature decoder card. Control signals for the motors are sent as voltage outputs from an analog output card. These voltage signals are then translated into current signals by a custom power amplifier board, based on the National

Semiconductor LMD18245 full-bridge motor driver. The current signals are sent to the motors via the tether.

The PC104 system is running Linux with an RTLinux microkernel, for real-time control and data sampling. The control and sampling software was developed specifically for this rover. User interaction with the PC104 system is done via a serial console connection to the laptop computer.

The laptop computer, a PC running Windows XP, interacts with the PC104 system, the CCD camera, and the vibration sensor. It connects to the PC104 system using a null modem serial cable. It connects to the CCD camera with a USB video capture box. The connection to the vibration sensor is via a standard audio cable which plugs into the laptop's microphone port. Model information for the input/output boards is shown in Table D-4.

Table D-4: FSRL Technology Testbed Rover I/O boards

I/O Boards	
PC104 Analog Input, Digital I/O	Diamond MM-AT
PC104 Quadrature Decoder	Microcomputer Systems MSI-P400
PC104 Analog Output	Ruby MM-4XT
USB Video Capture	ProVideo PV321CE

The entire system is run using battery power, so it can be taken to remote locations where electrical outlets are unavailable. Conveniently accessible locations include terrains such as gravel, concrete, grass, sand, and a mixture of sand, silt and clay.

Expansions for Remote Sensing

Remote sensing experiments required a forward looking stereo pair imager to be integrated to TORTOISE. A 50 cm long mast is mounted to the top cover and a Videre design stereo pair set was positioned on the mast, see Figure D.5.

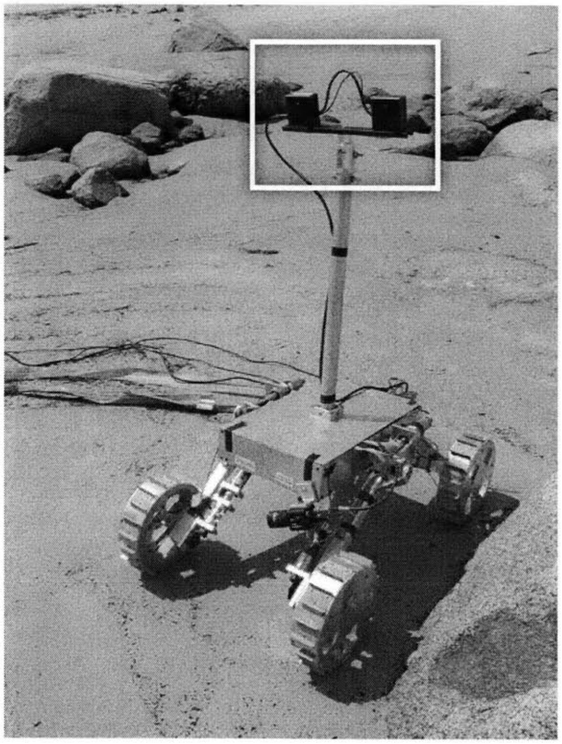

Figure D.5: TOROTISE stereo pair

The Stereo pair installed on TORTOISE is a Videre Design STH-D1-10204-W which is a version of DCAM series. Other details are shown in Table D-5.

Table D-5: Remote sensing system properties

	Stereo Pair Parameters	
Mast height	63 cm.	Fixed
Stereo baseline	19 cm.	Variable
Lens	12x0.5 mm miniature	Changeable
Focal length	3.6 mm.	Fixed
Sensor	Sony HAD ¼"	CCD
Resolution	640x480	Single frame
	320x240	15 Fr/sec