Classification Experiments on Real-World Texture

Rebecca Castaño
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109
Rebecca.Castano@jpl.nasa.gov

Roberto Manduchi
Department of Computer Engineering
University of California at Santa Cruz
Santa Cruz, CA 95064
manduchi@soe.ucsc.edu

Justin Fox
California Institute of Technology
Pasadena, CA 91109
jfox@its.caltech.edu

Abstract

Many papers have been published concerning the analysis of visual texture and yet, very few application domains use texture for image classification. A possible reason for this low transfer of the technology is the lack of experience and testing in real-world imagery. In this paper, we assess the performance of texture-based classification methods on a number of real-world images relevant to autonomous navigation on cross-country terrain and to autonomous geology. Texture analysis will form part of the closed loop that allows a robotic system to navigate autonomously. We have implemented two different classifiers on features extracted by Gabor filter banks. The first classifier models feature distributions for each texture class using a mixture of Gaussians. Classification is performed using Maximum Likelihood. The second classifier represents local statistics using marginal histograms of the features over a region centered on the pixel to be classified. We measure system performance by comparison to ground truth image labels.

1 Introduction

The use of visual features from imaging sensors for the control of autonomous robots is well established. Vision algorithms (most notably, stereo matching) are often deployed for inferring geometric information, such as the presence and the location of obstacles and the relative position of the robots in the environment. While the geometric representation of the world is extremely important for autonomous navigation, in several instances one is interested in a richer description of the surfaces in the scene. For example, two ongoing projects at the Jet Propulsion Laboratory (JPL) have among their goals the development of algorithms for classifying visible surfaces into a selected set of materials, based on the surface's appearance.

The first of such projects will enable a terrestrial military vehicle to drive safely and efficiently cross-country by day and night. In this environment, the notion of "obstacle" based on pure geometric considerations is not sufficient. For example, a small bush or a patch of tall grass may be considered "obstacles" from a purely geometric standpoint, yet the vehicle could drive safely over them. Therefore, for the robot to be able to drive in vegetated environments, the geometric scene description should be enriched with information about the terrain cover. Color analysis has been used for terrain typing, for example in the DEMO III project [1], where the surfaces in the scene are classified into a small number of classes (green vegetation, dry vegetation, soil, rock). However, color (or, in general, multispectral analysis in the reflective spectrum) has a number of shortcomings. First, it cannot be used at night. Second, it won't allow robust separation of

dry vegetation or tree bark from certain kinds of soil. This is a well-known problem in remote sensing data analysis [19]. Third, the perceived color is a function of both the reflectivity properties of the surface and of the spectrum of the illuminant, which makes color classification a rather hard problem in certain circumstances (for example, with changing weather conditions or when parts of the scene are in the shade). Therefore, we are currently studying the use of other visual features (as well as of other sensors, such as spectrometers in the thermal infrared domain or laser rangefinders) to complement color analysis for a more robust terrain cover classification.

The second project deals with the processing of visual information for rovers on Mars. Due to the long communication delays between Earth and Mars, as well as the scarce up-link opportunities, it is desirable that Mars rovers have a high degree of autonomy. Since the foremost goal of Mars exploration is the collection of surface science data, it is important that the amount of "interesting" data collected during a mission be maximized. Therefore, a rover should be able to autonmously select geological targets within its field of view, and to make motion planning decisions which maximize the science value of the targets that will be approached and sampled. For example, one may seek features that indicate the former presence of water and thus locations with higher than average opportunities for finding signs of life, if any exist. Examples of such indicators include layered sedimentary deposits and carbonate rocks, each of which has distinctive geolocal surfaces characteristics. Thus, the "importance" of a rock surface as a science target is a function of its visual appearance (the shape of the rock as well as the characteristics of its surface). Targets deemed interesting based on visual inspection may then be tested with a spectrometer to detect the actual rock composition. If "interesting" areas are selected in an image, a strategy of prioritized compression, buffering and transmission can also be used to best exploit the scarce onboard memory and communication bandwidth resources [15].

An important image feature that we plan to use for both projects, which is the object of this work, is visual texture. A quick look at the data sets in Tables 8-16 should convince an observer that texture is indeed be powerful cue for classifying different kinds of terrain cover and different kinds of rock surfaces. Visual texture analysis has been studied for over two decades, and a variety of algorithms and statistical methods have been proposed (see for example, [3], [4], [6], [8], [12], [18]). Yet, surprisingly few application domains use texture to classify images. Among them, we may count biomedical inspection [10], surface quality control [9], remote sensing [13], and content-based indexing of image databases [5], [11], [16]. We conjecture that one possible reason for not using texture in other domains is the lack of practical experience in real-world imagery. The vast majority of papers in the vision literature consider very simple test images, namely the Brodatz set or the like. Brodatz-like texture are rather homogeneous and suffer from very few of the problems of real-world images (e.g., foreshortening or illumination changes). On the other end of the spectrum, some work in the image databases literature considers completely unconstrained sets of image, albeit often concentrating on texture patches characterized by polka-dot or striped patterns. The problem with this kind of imagery is that the assessment of the system performance is rather difficult, due to the large variability within unconstrained data sets and to the difficulty of objective scoring criteria.

In this paper we assess the performance of a selected ensemble of texture-based classification algorithms on three data sets characteristic of the applications of our projects: autonomous navigation on cross-country terrain and autonomous geology. The image sets are therefore rather constrained (in both cases, they have been acquired by cameras on a robot) yet infinitely more realistic than Brodatz collages. The problem we are interested in is pixel-based supervised classification. This is quite a different task than unsupervised segmentation (or clustering) which is studied by most authors. We argue that in the vast majority of practical applications, classification is a more important issue than clustering. It also offers a higher hope to be implemented in real time, since clustering often requires time-consuming iterative procedures. Finally, classification allows for a more objective performance assessment, since the results of the classifier can be compared directly with the manual labeling performed by a human operator.

The ideal outcome of our work is a set of indicative performance scores, which ultimately should help an individual to make decisions about using of texture classification in a practical system. Given the large number of algorithms available in the literature, we had to pre-select a small number of candidate techniques for our experiments. We chose to implement two different classifiers operating on features extracted by multiscale-multioriented Gabor filter banks. Such feature operators have proven to be very effective and have become a standard choice in most of the recent algorithms in the literature.
The first classifier we implemented models the probability distribution function (pdf) of texture features using mixtures of Gaussian, and performs a Maximum Likelihood (ML) classification. This technique is simple and general; the Expectation Maximization (EM) algorithm can be used for training, and classification is rather fast. The second classifier represents local statistics by marginal histograms over small image squares. This technique has been proposed recently by several authors and is relatively simple to implement [20].

2 Texture Features

2.1 Feature extraction

Local texture features should condense information from a small neighborhood of a given pixel. A popular technique is based on the multiscale-multiorientation analysis by means of a Gabor filter bank. We used the implementation described by Manjunath and Ma in [16] with $N_\sigma = 3$ scales and $N_\tau = 4$ orientations between 0 and 180^o. These values represent a good compromise between computational complexity and classification performance. The filters are realized using an FFT implementation, and the magnitude of the complex Gabor output is used as local feature $f_{\sigma,\tau}(x)$ for each pixel x, scale σ and orientation τ. Note that this feature operator is non-linear.

2.2 Logarithm Processing

We implemented a simple pre-processing on the Gabor features to mitigate variations in image contrast (note that variations in average image brightness are filtered out by the zero-DC behavior of the Gabor filters). This operation consisted of taking the logarithm of the feature values, offset by one: $\log(f_{\sigma,\tau}(x) + 1)$. We observed that using logarithmic pre-processing consistently gave slightly improved classification results, and subsequently used it in all of our experiments.

2.3 Gaussian Smoothing

While the features $f_{\sigma,\tau}(x)$ can be used directly for classification, we also implemented the option of locally smoothing each component of the feature vector with a Gaussian kernel. In this case, rather than the distribution of the features $f_{\sigma,\tau}(x)$, we model the distribution of their variances[1]. A consequence of this procedure is that the image area represented by a local feature becomes larger. The standard deviation, σ_G of the Gaussian kernels was chosen to be proportional to the standard deviation of the envelope of the Gabor kernel ($\sigma_G = 4\sigma$).

2.4 Rotation Invariance

The principal orientation of a texture patch corresponding to a surface will change depending on the relative orientation between the surface and the camera. To deal with this variability, one may hope that the statistical model will account for the expected distribution of orientations, or use an operator which renders the texture features invariant to rotation. We implemented this option using a rotation-invariant operator based on the Fourier transform in the space of the orientations [7]. In particular, if DFT_τ represents the Discrete Fourier Transform computed along the orientation axis τ, the new orientation-invariant feature vector is $\hat{f}_{\sigma,\tau}(x) = \mathrm{DFT}_\tau^{-1}\left[|\mathrm{DFT}_\tau\left(f_{\sigma,\tau}(x)\right)|\right]$.

2.5 Histogram Features

Another feature vector that can be used for classification is formed by the normalized histograms of the components of $f_{\sigma,\tau}(x)$ within a square window centered on x. By normalized we mean that the histogram is divided by the number of points in

[1] More precisely, we model the variance of the square root of the features $f_{\sigma,\tau}(x)$.

the selected window. We experimented with window sizes, W, of 23 and 33 pixels, and with different numbers, N_b, of bins (8, 15 and 30). To determine the boundaries of the segments over which to compute the histograms, we looked at the min/max values of $f_{\sigma,\tau}(x)$ over the training sets. Histograms have a larger dimensionality than $f_{\sigma,\tau}(x)$ (equal to $N_b N_\sigma N_\tau$) and provide a rich statistical characterization of the local image behavior.

3 Statistical Texture Models

The heart of a classifier is in the way it represents the variability of the features within each class. We have selected two algorithms that use two different descriptions of the class statistics, one based on a parametric model and one based on histograms.

3.1 Models Based on Mixture of Gaussians

Maximum Likelihood (ML) classification relies on the knowledge of the class-conditional likelihood $p_j(f) = p(f|j)$, and assigns a feature f to the class j that maximizes $p_j(f)$. If the class priors $P_j = P(j)$ are uniform, ML and Bayesian classification coincide. Uniform priors is often a safe assumption, and is the only option when no further information is available. Alternatively, the prior for a given class may be chosen as the relative frequency this class appears in the training data set. However, note that, since Bayesian classification minimizes the expected Bayesian risk, classes with small prior probabilities are penalized. This can be a drawback in certain cases: for example, when it is important not to miss events with little probability of appearing. A solution, in this case, is to set different misclassification costs in the computation risk (which is equivalent to setting different prior probabilities). To compute the class-likelihoods $p_j(f)$, we used a mixture-of-Gaussians model:

$$p_j(f) = \sum_{k=1}^{N_j} P_{jk} G\left(f, \mu_{jk}, \sigma_{jk}\right)$$

where $G(f, \mu_{jk}, \sigma_{jk})$ is a Gaussian density with mean vector μ_{jk} and covariance matrix σ_{jk}. The parameters are estimated using the Expectation Maximization (EM) algorithm on a data set of labeled points. The selection of the model order N_j is obviously important. In our experiments, we selected such values (typically, equal to 3 or 4) according to preliminary testing; we are planning to use more theoretically founded methods such as cross-validation in the future. Initial values of μ_{jk} for the EM iterations are computed by an initial k-means clustering. This is a customary bootstrap technique that usually reduces the convergence time of EM. We experimented with both full and diagonal covariance matrices σ_{jk}. Note that a classification using the full covariance matrix may be computationally quite expensive, since for each pixel one must compute $(N_\sigma N_\tau)^2 \sum_{j=1}^{N_c} N_j$ multiplications/sums (as compared to $N_\sigma N_\tau \sum_{j=1}^{N_c} N_j$ in the case of diagonal covariance matrices.) In order to enforce spatial coherence, we implemented the option of post-processing the classification results (expressed in terms of posterior probabilities, assuming uniform priors P_j) using a "soft" version of Besag's Iterated Conditional Modes [2]. The relation of this technique to mean field theory is discussed in [21].

3.2 Models Based on Histograms

This classifier uses a different approach than ML. Rather than maximizing the conditional likelihood of the feature $f(x)$, one computes an estimate $\bar{p}(f(x))$ of the density of f in a neighborhood of x. Then $\bar{p}(f(x))$ is compared with the model conditional likelihood $p_j(f)$ using a suitable metric, and f is assigned to the class that minimizes such distance. In this case, both $\bar{p}(f(x))$ and $p_j(f)$ are expressed as marginal histograms (i.e., as the set of histograms of the component of f). Note that the model histogram representing $p_j(f)$ is simply the average of the local histograms $\bar{p}(f(x))$ for the pixels labeled as j in the training set. Thus, any class j is actually represented by the mean of the histogram features of the points labeled as j, where the concept of histogram feature is defined in Section 2.4. The classifier simply minimizes the distance between a given histogram feature and such means. It is implicitly assumed that all points within the selected window centered around the evaluation point belong to the same class. We have considered four different histogram distances [17]:

$$
\begin{aligned}
D_1(h_1,h_2) &= \textstyle\sum_i \sum_k |h_{1,k}(i) - h_{2,k}(i)| && (L1)\\
D_2(h_1,h_2) &= \textstyle\sum_i \sum_k (h_{1,k}(i) - h_{2,k}(i))^2 && (L2)\\
D_3(h_1,h_2) &= \frac{\sum_i \sum_k (h_{1,k}(i) - h_{2,k}(i))^2}{h_{w,k}(i)} && (\chi^2)\\
D_4(h_1,h_2) &= \textstyle\sum_i \sum_k h_{1,k}(i) \log \frac{h_{1,k}(i)}{h_{2,k}(i)} && (KL)
\end{aligned}
$$

where h_1,h_2 are histogram features ($h_{i,k}k(i)$ represents the value in the i-th bin of the histogram of the k-th component of f).
Computing the nearest class involves determining bin membership for each marginal histogram at every pixel. Since each class has different histogram bin boundaries, bin membership must be calculated for each class independently. The cost of computing all the histograms at each pixel over N_b with a window of size R is on the order of $W^2/log_2 N_b$. Thus, the classification cost is on the order of $N_c N_\sigma N_\tau (W^2 \log_2 N_b + N_b)$. To reduce computational time, classification was performed on the set of pixels obtained by subsampling both rows and columns by 5. This procedure is justified because the windows surrounding nearby pixels are overlapping (given that $W \geq 23$, resulting in a large number of points contributing to different local histograms. Of course, the larger the window size, the more similar the histograms for neighboring pixels will be. The histogram method thus intrinsically leads to smooth of the classification maps.

4 Performance Evaluation

4.1 Data Sets

We have selected three image sets for our experiments. The first two sets have been collected at Fort Knox with cameras mounted on the DEMO III Experimental Unmanned Vehicle (XUV) [1]. One set contains 18 images collected by a camera in the visible spectrum, while for the other set (25 images) an infrared (FLIR) camera has been used. Texture analysis on infrared images may enable terrain cover classification at night. The images in the two sets contain a number of different situations typically encountered while driving on vegetated terrain. We selected the images so as to obtain a training/testing ensemble as diversified as possible. Sometimes (e.g., images 2 and 3 in the FLIR data set, Table 11) the images look very similar, but with different orientation. In other instances (e.g., image 10 and 11 in the FLIR data set, Table 12) the same scene is imaged from two different distances, resulting in similar appearance but at two different scales. The third data set consists of 13 images taken in the JPL "Mars Yard". These images contain a variety of different rocks and soil types, and are considered representative of scenes encountered by a rover exploring the surface of Mars.

4.2 Class Selection and Labeling

Selecting the set of classes is a very important and delicate operation. On the one hand, one's decision should be driven by the application, selecting as many classes as deemed appropriate for the task to be carried out. On the other hand, a large taxonomy may simply not make sense, if the classifier is unable to discriminate among such variety of classes within the selected feature space. Even the optimal Bayesian classifier will perform poorly if the class-conditional likelihoods overlap heavily. In practical terms, this corresponds to situations where a human would label two regions having exactly the same texture feature f with different class labels, based on contextual information.
For our experiments, we have identified four classes of interest in the first two data sets: soil (dirt), trees, bushes/grass, and sky. Discriminating soil from grass and bushes is important for autonomous navigation, and cannot be achieved by range analysis. Detecting tree lines has tactical value (although isolated trees can be detected robustly by range sensors). For the Mars Yard data set (Tables 15,16), six classes have been identified: sand, small pebbles, smooth surfaced rocks, glassy volcanic (layered) rocks, dimpled volcanic rocks, and billowy, medium-grained rocks. This class taxonomy reflects the desiderata of autonomous geology at Mars.
Critical to evaluating algorithms is the ground truth for comparison. All images in the data set have been hand-labeled; however, not all points in an image have been labeled. This is because there are situations not well represented by any of the chosen classes; further, as the filters have a spatial extent, the feature vectors near a region boundary repre-

sent textures from more than one class, and therefore should not be used for training. The second column in each table of results represents the hand-labeled regions where each class has a unique color throughout the data set. Points in the labeled regions are used for training as well as for testing, as described in the next section.

4.3 Performance Metrics

To provide a quantitative assessment of the algorithms' performances, we collected the classification results on the labeled regions in suitable confusion matrices. The entry, $CM_{j,k}$ of such a matrix represents the number of times a pixel labeled as j has been classified as k. We selected two scoring functions based on the confusion matrices. The first scoring function is the number of correctly classified points over the total number of points in the complete training set:

$$P_C = \frac{\sum_j CM_{j,j}}{\sum_{j,k} CM_{j,k}}$$

The second function provides an estimate of the probability of correct classification for each class:

$$P(C|j) = \frac{CM_{j,j}}{\sum_k CM_{j,k}}$$

Note that P_C actually represents the overall probability of correct classification, assuming that the class prior probabilities are given by the relative frequencies of points for each class in the data set. Indeed, in such a case, we can write

$$P(j) = \frac{\sum_k CM_{j,k}}{\sum_{j,k} CM_{j,k}}$$

and

$$P_C = \sum_j P(C|j)P(j)$$

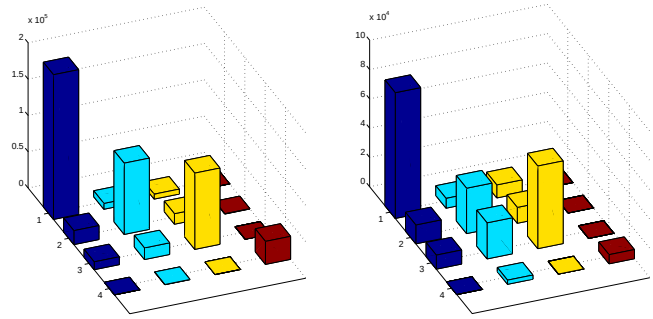
One problem with the measure P_C is that it is biased toward the results of the class with the highest number of points in the testing set. We therefore also used another global measure, $\bar{P}_C$, which corresponds to the average of $P(C|j)$. In other words, $\bar{P}_C$ is the probability of correct classification assuming that the classes are equiprobable.

5 Experimental Results

We tested our classification algorithms varying a number of parameters described in Section 2. In particular, we tried with and without smoothing the Gabor filter outputs (see Section 2.3, *Smooth* = 1–0), with and without rotation invariance operator (see Section 2.4, *RotInv* = 1–0), using diagonal or full covariance matrix for the Mixture Model algorithm (*FullCov*=1–0), and with and without spatial coherence enforcing (see Section 2.4, *SpatCoh*=1–0). We first trained the classifiers over the complete labeled data sets. This best-case scenario is important to understand the intrinsic limitations of our techniques. We also ran our experiments using only one half of the images for training, and the remaining half for testing.

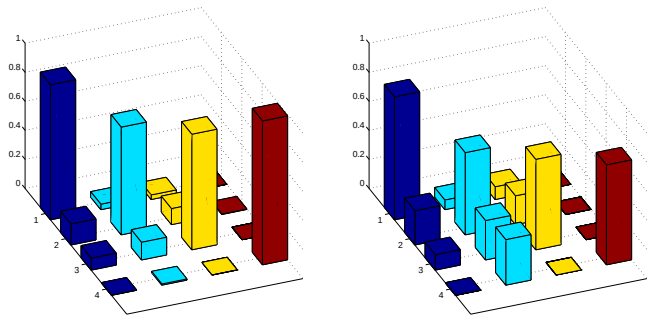
The performances varied considerably with different parameter selection. The numerical results corresponding to the optimal parameter selection are shown in Tables 1–3. For the Mixture Model classifier, the parameter vector that gave the best performances (in terms of P_C) was consistently equal to *Smooth*=1, *RotInv*=0, *FullCov*=1, *SpatCoh*=1. Tables 4 and 5 show the classification results after invididually varying each of the parameters with respect to the optimal combination for the Visible Spectrum Dataset. Note that in particular, enforcing rotation invariance consistently deteriorated the results. Using diagonal covariance matrices rather than full (which, as discussed above, would reduce the computational burden) does not seem to degrade the results dramatically, at least when only half of the images are used for training. Smoothing the Gabor features and enforcing spatial coherence appear to be both very important operations to ensure good classification results.

For what concerns the Histogram method, there are two more variables to be considered (N_b and W), as well as the choice of the histogram distance (see Section 3.2). The Chi-square distance consistently gave better classification results than the other three, and we decided to only show results with such a choice. Furthermore, given that the Histogram method has inherent low spatial resolution (because of the window size W), we decided not to use spatial coherence enforcement post-processing (*SpatCoh*=0). It turned out that there was no pa-



(Full) (Half)

Figure 1: Confusion matrices for the results of Table 1 using the Mixture Model algorithm.

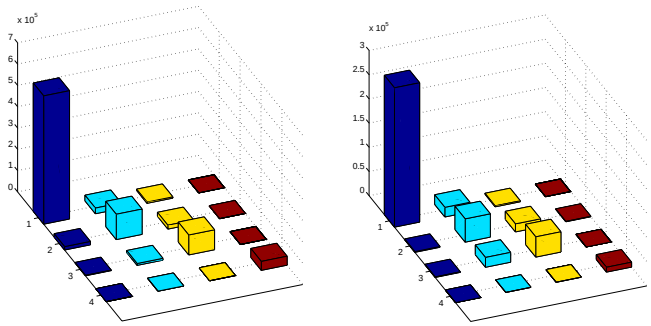


(Full) (Half)

Figure 2: Confusion matrices with rows normalized to 1 for the results of Table 1 using the Mixture Model algorithm.

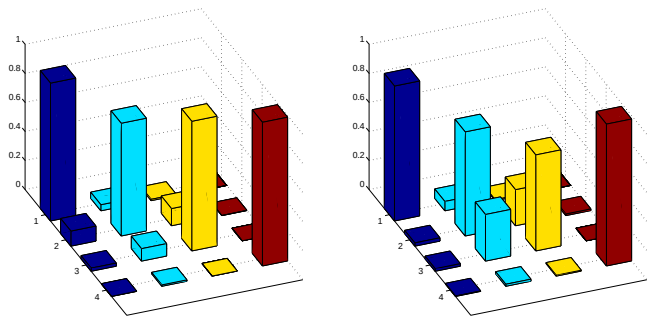
rameter vector that was optimal for all three data sets. Indeed, for the Visible Spectrum and the Rock Dataset, the optimal parameters where $Smooth$=1, N_b=15 and W=33 (as in the Mixture Model case, the rotation invariance operator always degraded the results, and we will assume $RotInv$=0 in the following.) For the IR Dataset, however, the best results were attained with $Smooth$=0, W=33, and N_b=30 when all images were used for training, or N_b=15 when training was performed on only one half of the images. In the case of the Visible Spectrum Dataset, if no smoothing is performed on the Gabor features ($Smooth$=0), the performances lower to P_C=0.7532, $\bar{P}_C$=0.7651 when training on full dataset, and P_C=0.6381, $\bar{P}_C$=0.5727 when training on half dataset. The matrices in Table 5 show the results for different choices of the parameters N_b and W in the Visible Spectrum Dataset (assuming $Smooth$=1).

In Tables 8-16 we show in the third and the fourth columns the classification results obtained using the optimal parameters for the mixture-model and histogram-based classifiers respectively. These results have been obtained with the classifiers trained over each whole data set.



(Full) (Half)

Figure 3: Confusion matrices for the results of Table 2 using the Mixture Model algorithm.



(Full) (Half)

Figure 4: Confusion matrices with rows normalized to 1 for the results of Table 2 using the Mixture Model algorithm.

6 Conclusions

We have assessed the performance of two representative texture-based classification methods on three sets of real-world images. The two classification methods considered (Mixture Model and Histogram) yielded fairly consistent results in most

Visible Spectrum Dataset				
	Full		Half	
Prob. Correct	Mixture	Hist	Mixture	Hist
P_C	0.8507	0.7561	0.7014	0.6686
$\bar{P}_C$	0.8649	0.7841	0.6805	0.6258
$P(C\|1)$ (soil)	0.9305	0.8159	0.8463	0.8707
$P(C\|2)$ (trees)	0.7418	0.7083	0.5660	0.4936
$P(C\|3)$ (bushes,grass)	0.7961	0.6639	0.6223	0.5558
$P(C\|4)$ (sky)	0.9909	0.9488	0.6874	0.5830

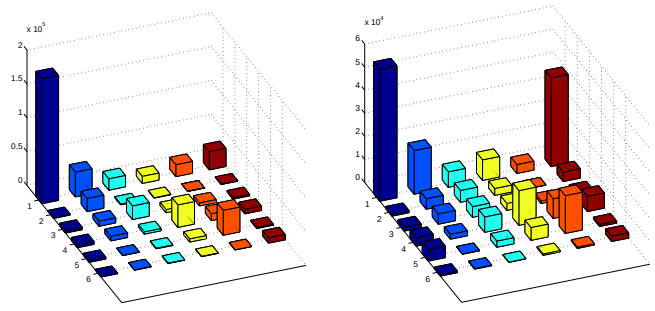
Table 1: Classification performance on visible spectrum image data set. The performance measures are described in the text. Results for two different runs are shown: 1) train on entire data set, test on entire data set, 2) train on half data set, test on other half.

IR Dataset				
	Full		Half	
Prob. Correct	Mixture	Hist	Mixture	Hist
P_C	0.9164	0.8564	0.8612	0.7868
$\bar{P}_C$	0.9019	0.7955	0.8211	0.7207
$P(C\|1)$ (soil)	0.9492	0.9260	0.9294	0.8911
$P(C\|2)$ (trees)	0.7761	0.6776	0.7169	0.4749
$P(C\|3)$ (bushes,grass)	0.8930	0.6816	0.6612	0.5886
$P(C\|4)$ (sky)	0.9893	0.8971	0.9769	0.9281

Table 2: Classification performance on infrared data set (see caption of Table 1).

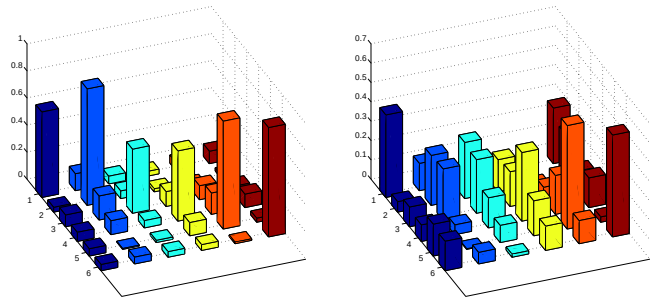
Rock Dataset				
	Full		Half	
Prob. Correct	Mixture	Hist	Mixture	Hist
P_C	0.6321	0.4910	0.4071	0.4028
$\bar{P}_C$	0.6812	0.5167	0.3963	0.4227
$P(C\|1)$ (sand)	0.6307	0.5047	0.4267	0.3545
$P(C\|2)$ (pebbles)	0.8603	0.6960	0.2581	0.5274
$P(C\|3)$	0.4734	0.3539	0.2769	0.2912
$P(C\|4)$	0.5189	0.3556	0.3547	0.3584
$P(C\|5)$	0.7963	0.6105	0.5350	0.6626
$P(C\|6)$	0.8075	0.5794	0.5267	0.3420

Table 3: Classification performance on Mars Yard rocks data set (see caption of Table 1).



(Full) (Half)

Figure 5: Confusion matrices for the results of Table 3 using the Mixture Model algorithm.



(Full) (Half)

Figure 6: Confusion matrices with rows normalized to 1 for the results of Table 3 using the Mixture Model algorithm.

Visible Spectrum Dataset		
	Mixture	
P_C	Full	Half
RotInv=1	0.6975	0.5370
Smooth=0	0.7835	0.6869
FullCov=0	0.8007	0.7056
SpatCoh=0	0.7868	0.6522

Table 4: Probability of correct classification (P_C) using the Mixture Model algorithm as a function of different parameters (see text.)

Visible Spectrum Dataset		
	Mixture	
$\bar{P}_C$	Full	Half
RotInv=1	0.7498	0.5908
Smooth=0	0.7999	0.6518
FullCov=0	0.8187	0.6877
SpatCoh=0	0.8136	0.6480

Table 5: Probability of correct classification ($\bar{P}_C$) using the Mixture Model algorithm as a function of different parameters (see text.)

Visible Spectrum Dataset			
P_C	N_b=8	N_b=15	N_b=30
W=23	0.7333	0.7426	0.7354
W=33	0.7478	0.7561	0.7423

(Full)

Visible Spectrum Dataset			
P_C	N_b=8	N_b=15	N_b=30
W=23	0.6347	0.6428	0.6406
W=33	0.6561	0.6686	0.6578

(Half)

Table 6: Probability of correct classification (P_C) using the Histogram algorithm with different values of the paraneters N_b and W with respect to the case of Table 1.

Visible Spectrum Dataset			
$\bar{P}_C$	N_b=8	N_b=15	N_b=30
W=23	0.7670	0.7794	0.7412
W=33	0.7652	0.7841	0.6997

(Full)

Visible Spectrum Dataset			
$\bar{P}_C$	N_b=8	N_b=15	N_b=30
W=23	0.6027	0.6399	0.6113
W=33	0.5754	0.6258	0.5403

(Half)

Table 7: Probability of correct classification ($\bar{P}_C$) using the Histogram algorithm with different values of the paraneters N_b and W with respect to the case of Table 1.

cases, although the Mixture Model algorithm was almost always slightly superior. The performances were rather satisfactory for two of the dataset considered, with a simple set of terrain cover classes. The results on the geological data set, however, where a more ambitious ensemble of surface classes was targeted, were less impressive. One reason might be that hand labeling was based on more criteria than mere visual texture. Nevertheless, even in the case of the geological database, the correct classification rate was much higher than the bound of random classification, which suggests that the information in its present form may be useful when used in conjunction with additional features other than texture, such as color and shape.

Acknowledgements

The research described in this paper was funded by the NASA Remote Exploration and Experimentation Project and by the DARPA MARS program and was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

References

[1] Bellutta, P., R. Manduchi, L. Matthies, K. Owens, A. Rankin, "Terrain Perception for DEMO III", *IEEE Intelligent Vehicles Symposium 2000*, Dearborn, MI, October 2000, 326-332

[2] Besag, J.,"On the statistical analysis of dirty pictures," *J. Royal Statistical Society B*, 48(3), pp. 259-302, 1986.

[3] Chang, K. I., K. W. Bowyer, and M. Sivaguranath, "Evaluation of Texture Segmentation Algorithms," in *Int. Conf. Comp. Vis. Pat. Recognition*, pp294-299, 1999.

[4] De Bonet, J. S. and P. Viola, "Texture recognition using a non-parametric multi-scale statistical model," *Proc. IEEE Conf. Comp. Vision Pat. Rec.*, 1998.

[5] Gimel'farb, G. L. and A. K. Jain, "On retrieving textured images from an image database," *Pattern Recognition*, 29(9), pp. 1461-1483, 1996.

[6] Gottlieb, C. C.and H. E. Kreyszig, "Texture descriptors based on co-occurrence matrices," *Comp. Vis., Graphics, Image Proc.*, 51, 1990.

[7] Greenspan, H., S. Belongie, P. Perona, R. Goodman, S. Rakshit, C.H. Anderson, "Overcomplete Steerable Pyramid Filters and Rotation Invariance", *Proc. IEEE CVPR*, 1994, 222-228.

[8] Haralick, R. M., K. Shanmugam, and I. Dinstein, "Textural features for image classification," *IEEE Trans. Sys. Man. Cyb.* 3(6), pp. 610-621, 1973.

[9] Hepplewhite, L., T. J. Stonham, and R. J. Glover, "Automated visual inspection of magentic disk media," *Proc. Third ICECS*, Vol. 2, pp. 732-735, 1996.

[10] Jennane, R., W. J. Ohley, S. Majumdar, and G. Lemineur, "Fractal analysis of bone X-ray tomographic microscopy projections," *IEEE Trans. Medical Imaging*, 20(5), pp 443-449, May 2001.

[11] Kaplan, L.M., et al. "Fast texture database retrieval using extended fractal features," *Storage and Retrieval for Image and Video Databases, VI*, Vol. 3312, pp. 162-173, SPIE Press, 1998.

[12] Laine, A. and J. Fan, "Texture classification by wavelet packet signature," *IEEE Trans. Pat. Anal. Mach. Intel.*, 15(11), pp. 1186-1191, Nov. 1993.

[13] Li, S. C. and V. Castelli, "Deriving texture feature sets for content-based retrieval of satellite image database", *Proc. Int. Conf. Image Processing*, 1997.

[14] Manduchi, R., "Bayesian fusion of color and texture segmentations," in *Int. Conf. Comp. Vision*, 1999.

[15] Manduchi, R., S. Dolinar, F. Pollara, A. Matache, "Onboard Science Processing and Buffer Management for Intelligent Deep Space Communications", *Proc. Of IEEE Aerospace 2000*, 2000.

[16] Manjunath, B. S. and W. Y. Ma, "Texture features for browsing and retrieval of image data," *IEEE Trans. Pat. Anal. Mach. Int.*, 18(8), pp. 837-842, 1996.

[17] Puzicha, J., J.M. Buhmann, Y. Rubner, and C. Tomasi, "Empirical evaluation of dissimilarity measures for color and texture," in *Int. Conf. Comp. Vision*, pp. 1165-1172, 1999.

[18] Randen, T. and J. H. Husoy, "Filtering for texture classification: a comparative study," *IEEE Trans. Pat. Anal. Mach. Intell.*, 21(4), pp. 291-310, 1999.

[19] Roberts, D. A., M.O. Smith, J.B. Adams, "Green Vegetation, Nonphotosynthetic Vegetation, and Soils in AVIRIS Data", *Remote Sens. Environ.*, 44:255-269, 1993.

[20] Rubner, Y. and C. Tomasi, "Texture metrics," in *Int. Conf. Sys. Man. Cyb.*, pp. 4601-4607, 1998.

[21] Zhang, J., J.W. Modestino, "The Mean-Field Theory in EM Procedures for Markov Random Fields", *Proc. IEEE ISIT*, 1991.

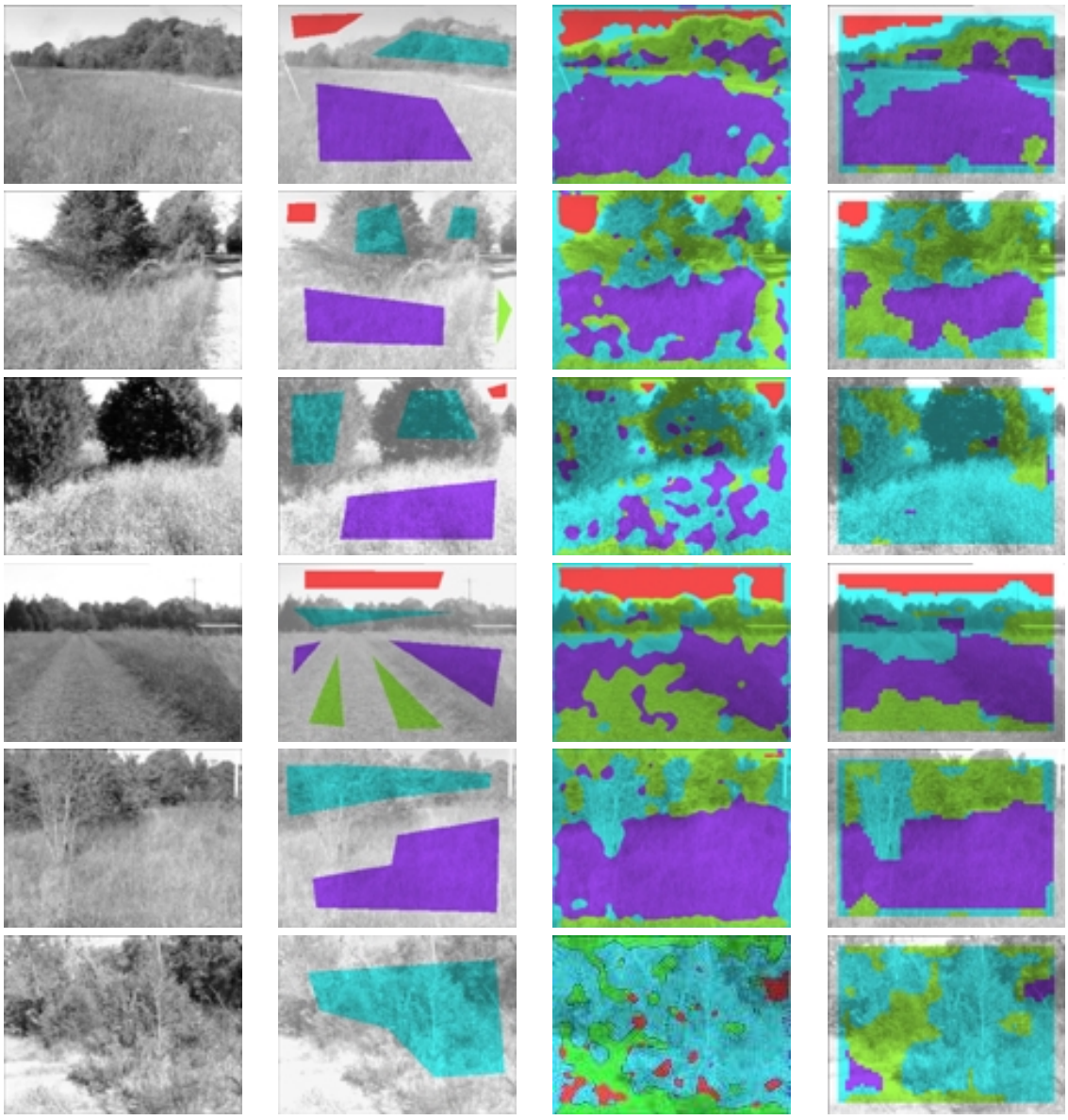


Table 8: Visible spectrum data set. (a) original gray scale image (b) hand labeled regions (c) results using mixture of Gaussians model (d) results using histogram model. The results shown were generated after training on the whole data set. Classes are sky (red), trees (cyan/blue), dirt/soil (green), bushes/grass (purple).

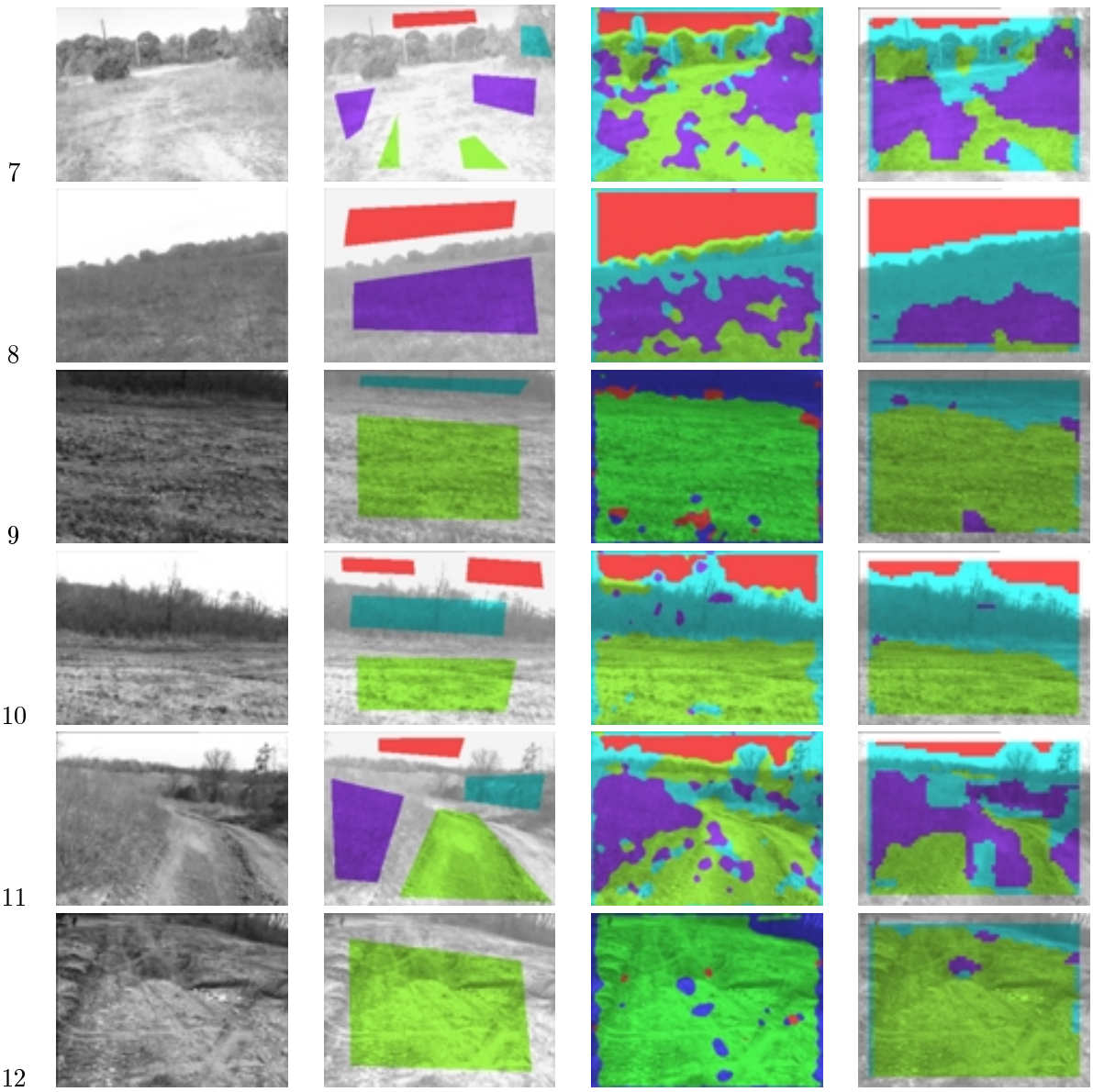


Table 9: Visible spectrum data set, cont. (a) original gray scale image (b) hand labeled regions (c) results using mixture of Gaussians model (d) results using histogram model. The results shown were generated after training on the whole data set. Classes are sky (red), trees (cyan/blue), dirt/soil (green), bushes/grass (purple).

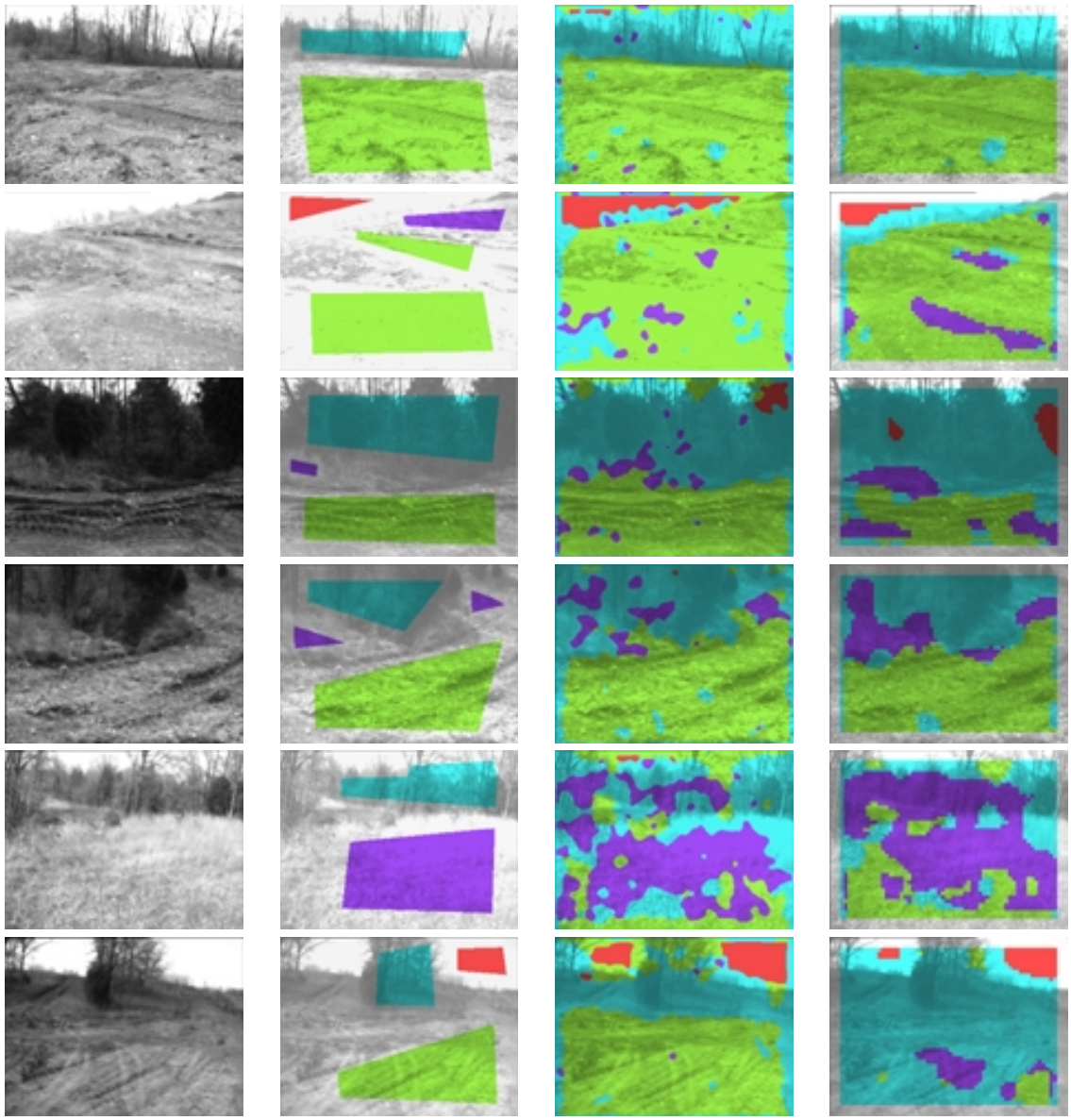


Table 10: Visible spectrum data set, cont. (a) original gray scale image (b) hand labeled regions (c) results using mixture of Gaussians model (d) results using histogram model. The results shown were generated after training on the whole data set. Classes are sky (red), trees (cyan/blue), dirt/soil (green), bushes/grass (purple).

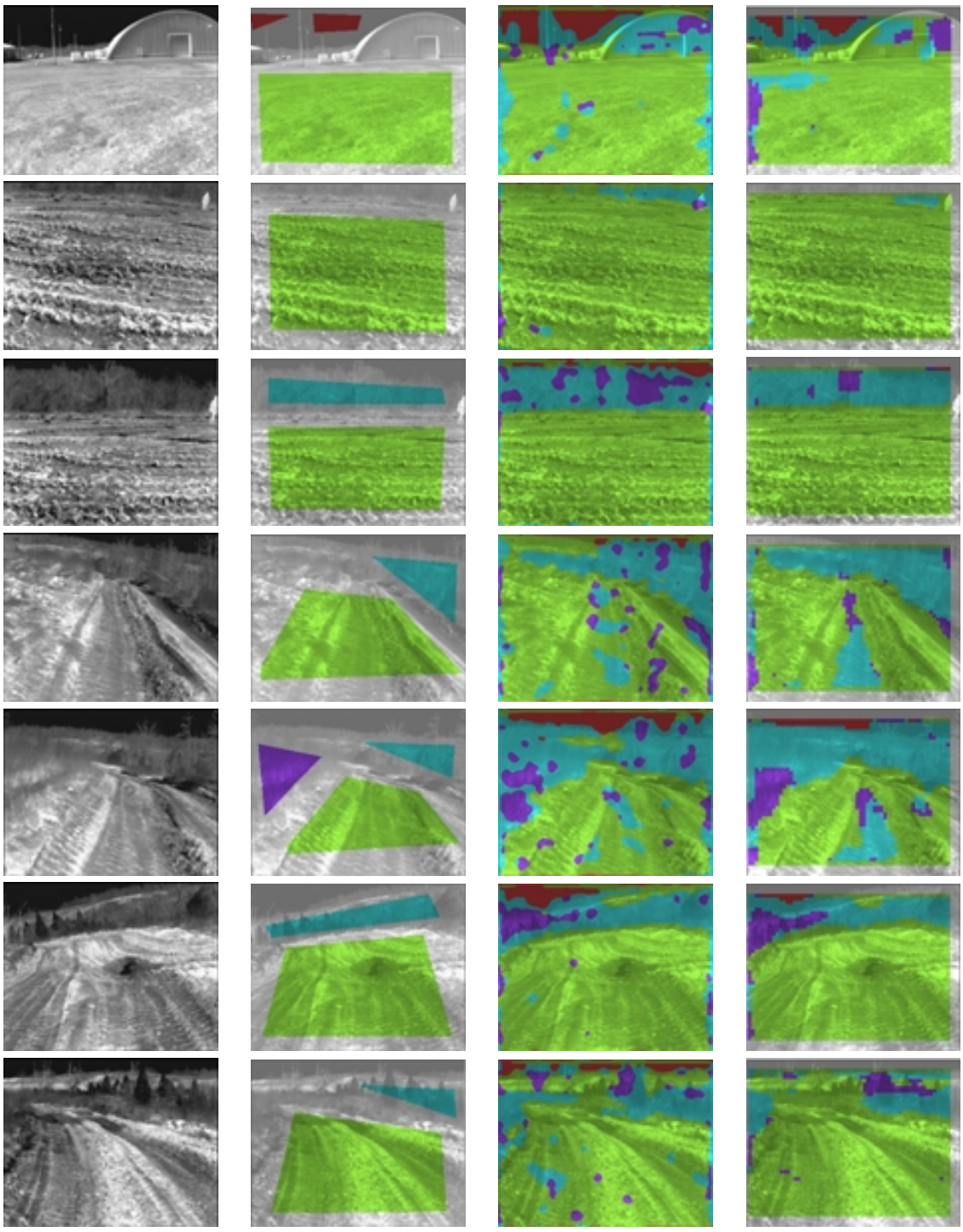


Table 11: Infrared data set. (a) original gray scale image (b) hand labeled regions (c) results using mixture of Gaussians model (d) results using histogram model. The results shown were generated after training on the whole data set. Classes are sky (red), trees (cyan/blue), dirt/soil (green), bushes/grass (purple).

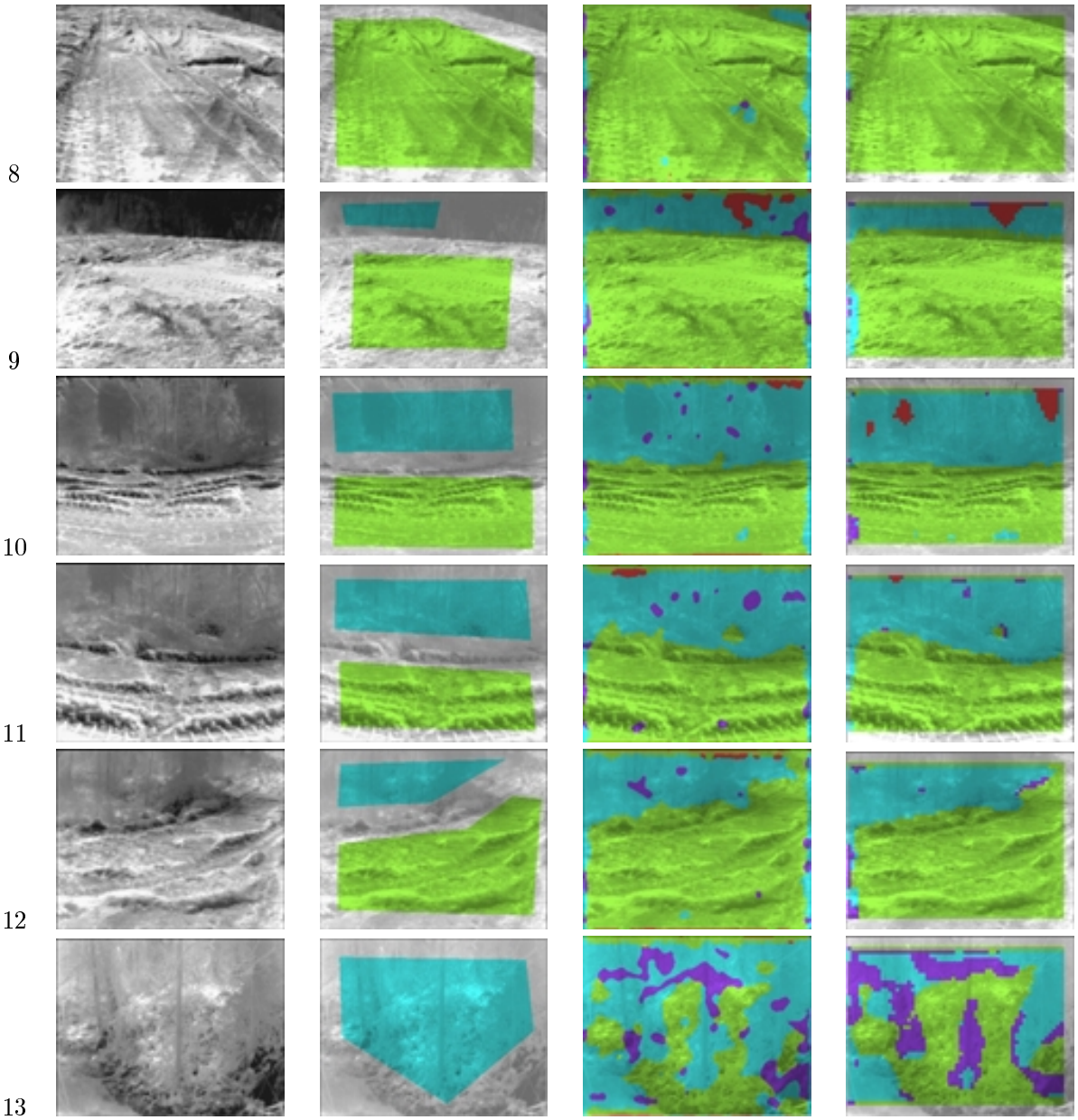


Table 12: Infrared data set, cont. (a) original gray scale image (b) hand labeled regions (c) results using mixture of Gaussians model (d) results using histogram model. The results shown were generated after training on the whole data set. Classes are sky (red), trees (cyan/blue), dirt/soil (green), bushes/grass (purple).

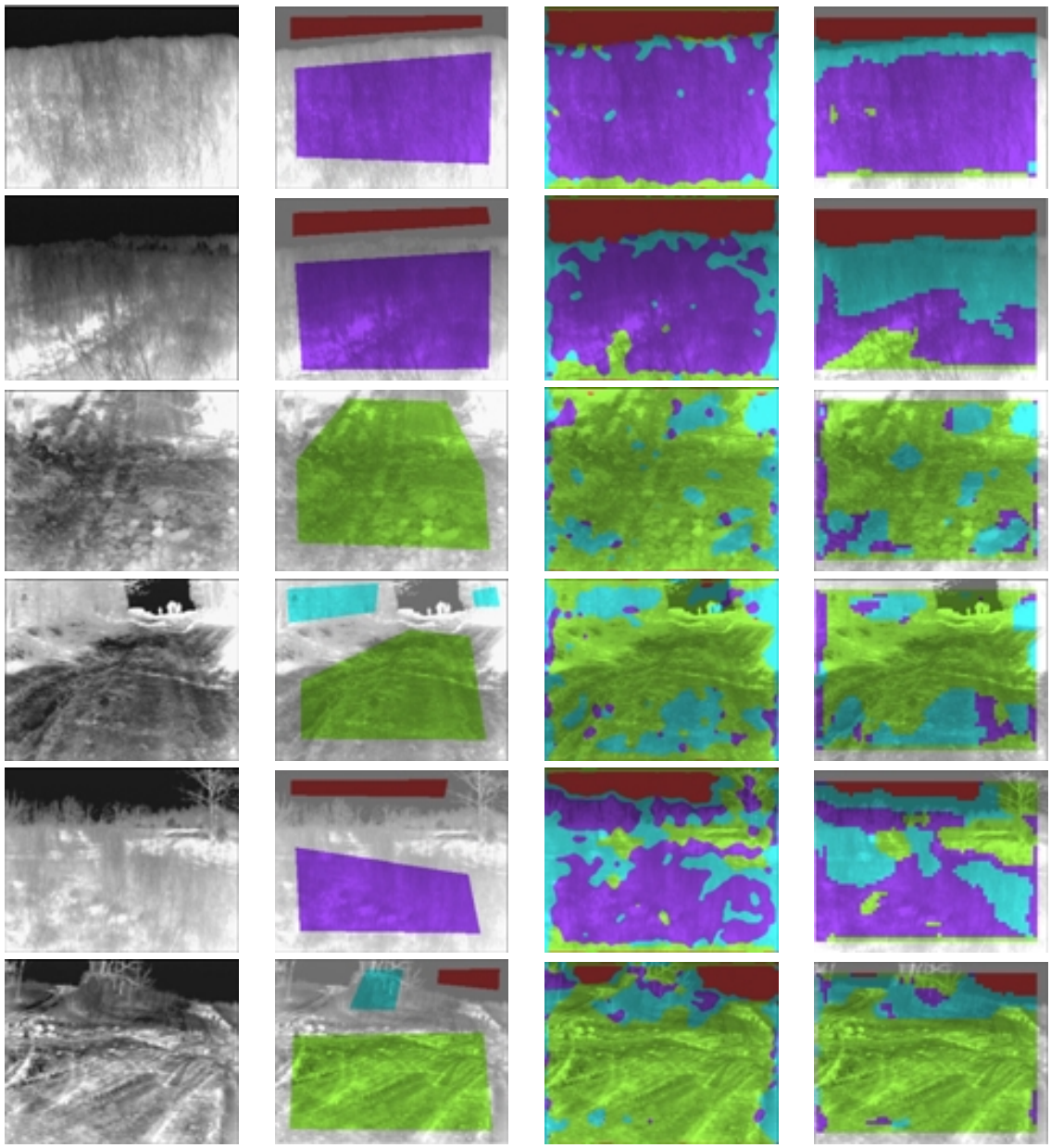


Table 13: Infrared data set, cont. (a) original gray scale image (b) hand labeled regions (c) results using mixture of Gaussians model (d) results using histogram model. The results shown were generated after training on the whole data set. Classes are sky (red), trees (cyan/blue), dirt/soil (green), bushes/grass (purple).

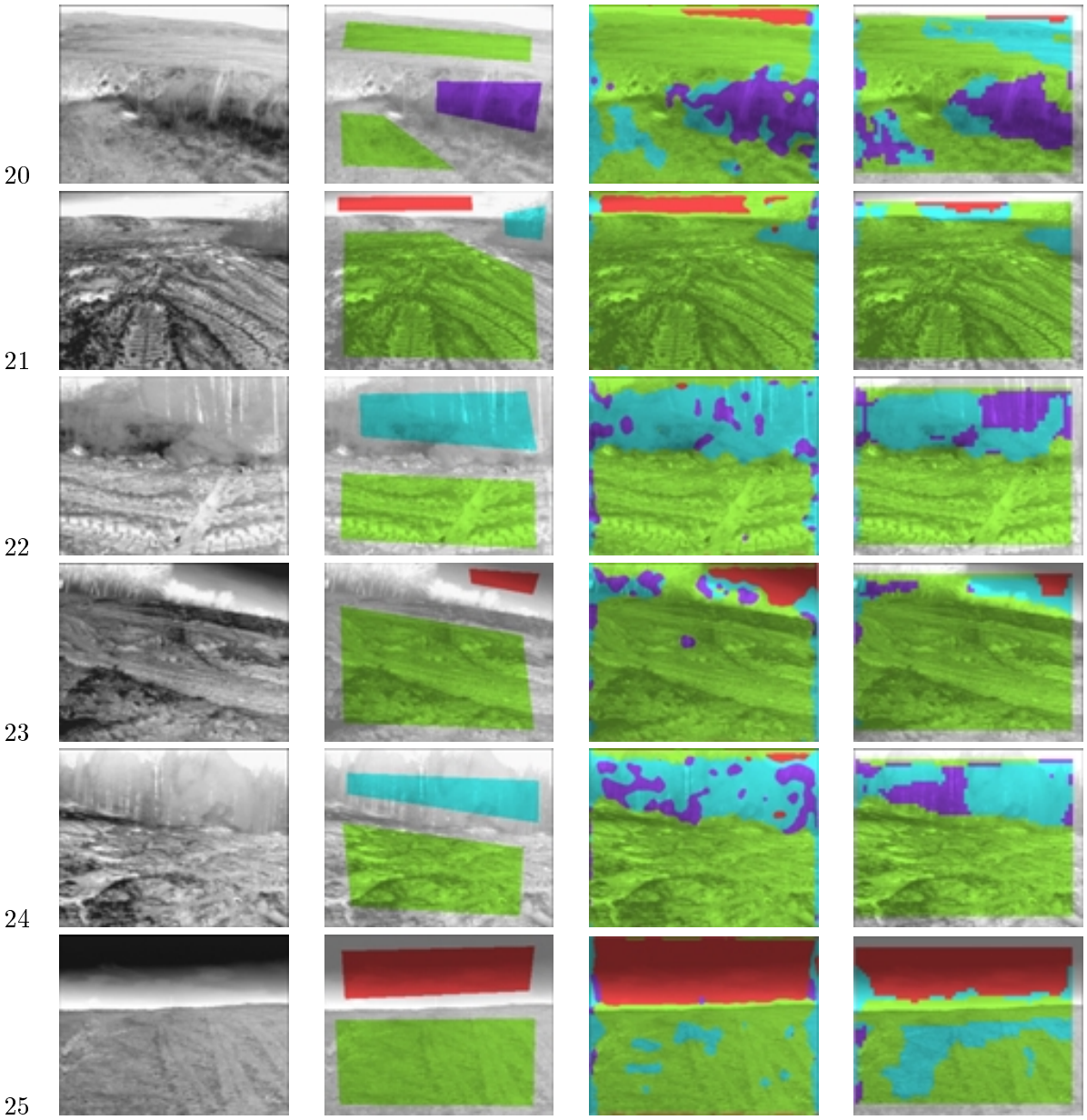


Table 14: Infrared data set, cont. (a) original gray scale image (b) hand labeled regions (c) results using mixture of Gaussians model (d) results using histogram model. The results shown were generated after training on the whole data set. Classes are sky (red), trees (cyan/blue), dirt/soil (green), bushes/grass (purple).

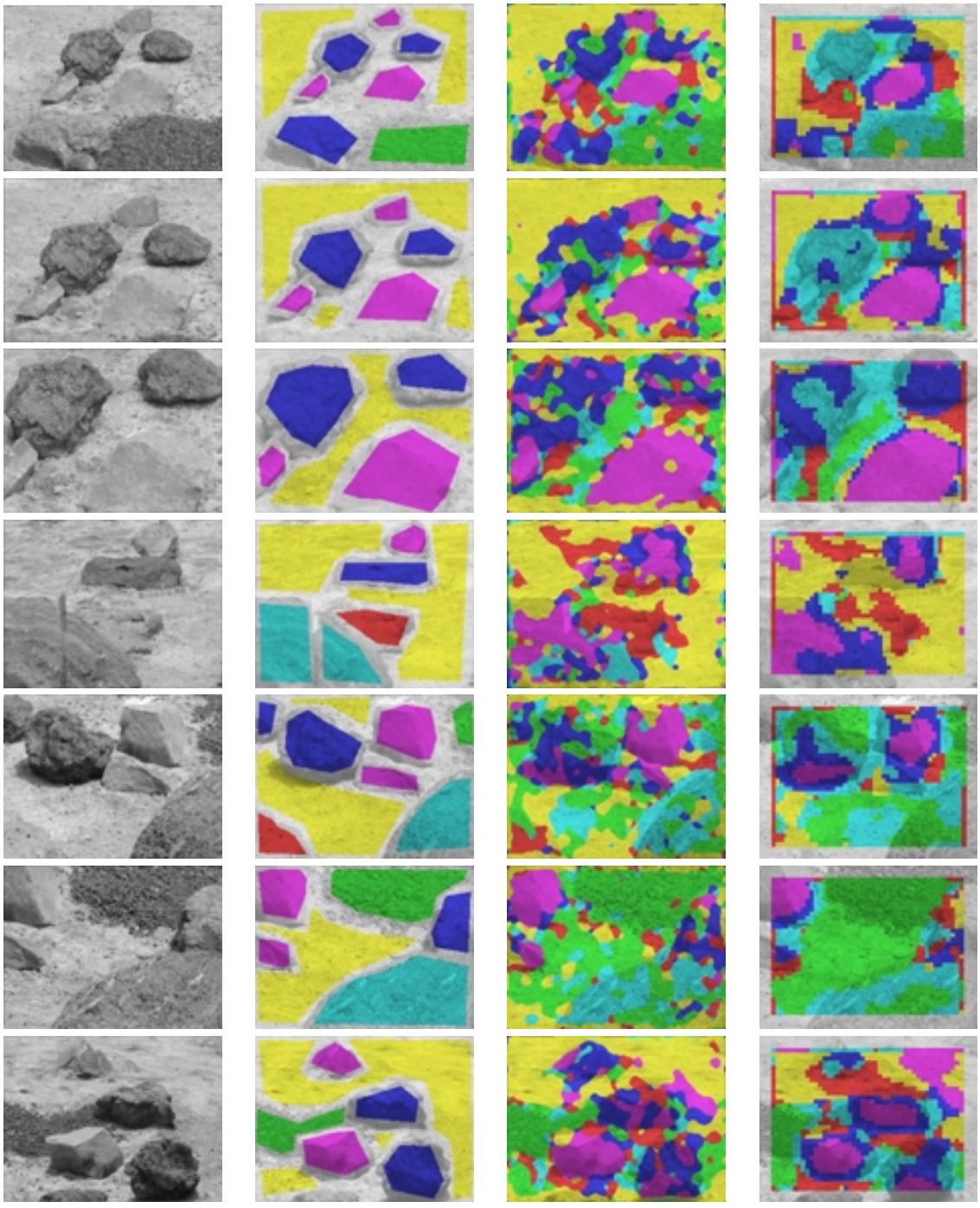


Table 15: Mars yard rocks data set. (a) original image (b) hand labeled regions (c) results using mixture of Gaussians model (d) results using histogram model. The results shown were generated after training on the whole data set. Classes are sand (yellow), small pebbles (green), smooth surfaced rocks (magenta), glassy volcanic rocks (cyan), dimpled volcanic rocks (red), medium-grained rocks (blue).

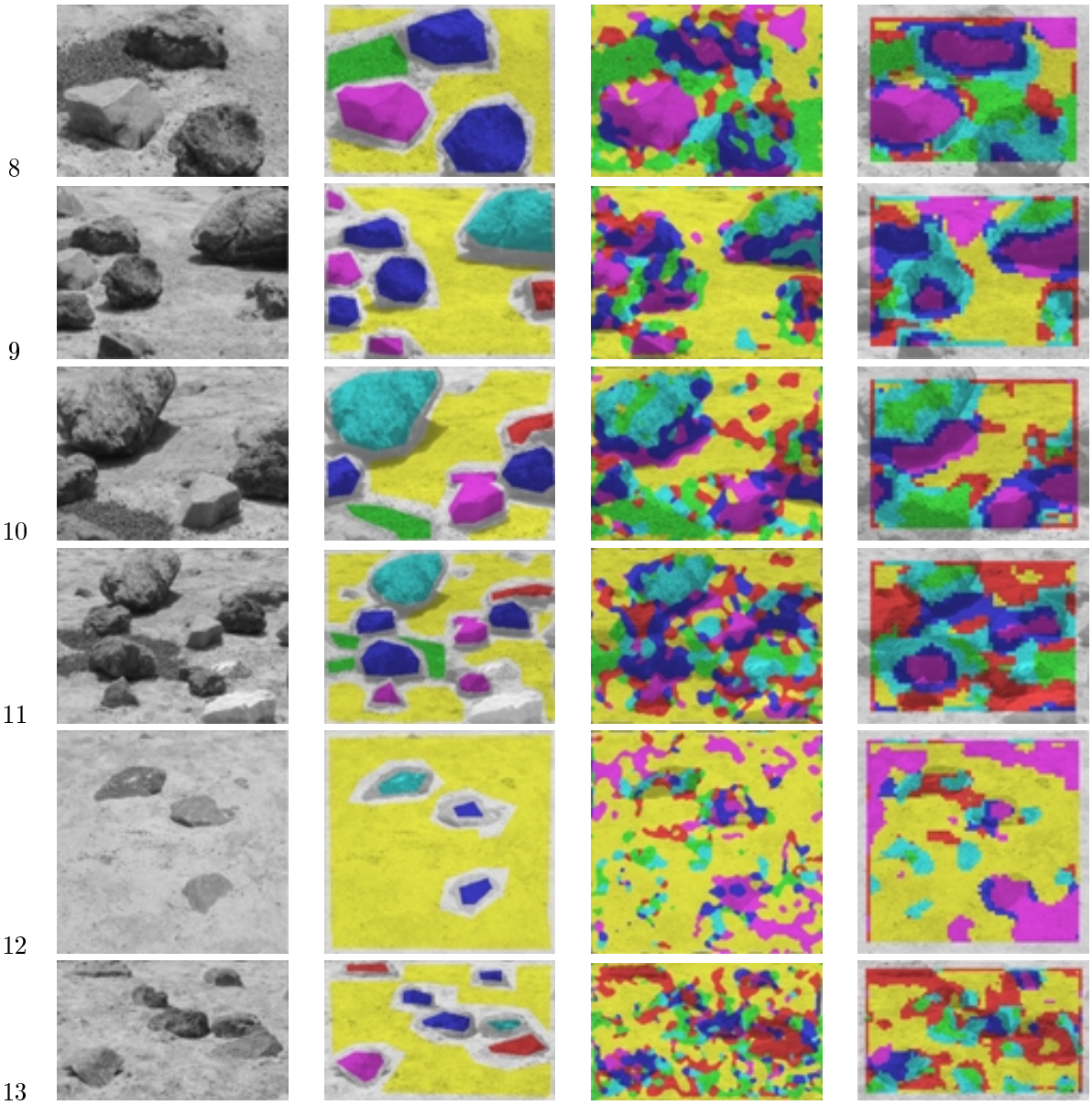


Table 16: Mars yard rocks data set, cont. (a) original image (b) hand labeled regions (c) results using mixture of Gaussians model (d) results using histogram model. The results shown were generated after training on the whole data set. Classes are sand (yellow), small pebbles (green), smooth surfaced rocks (magenta), glassy volcanic rocks (cyan), dimpled volcanic rocks (red), medium-grained rocks (blue).