Machine Vision and Applications (2008) 19:467–482
DOI 10.1007/s00138-007-0081-3

SPECIAL ISSUE PAPER

Automatic detection of dust devils and clouds on Mars

Andres Castano · Alex Fukunaga · Jeffrey Biesiadecki · Lynn Neakrase · Patrick Whelley · Ronald Greeley · Mark Lemmon · Rebecca Castano · Steve Chien

Received: 14 April 2006 / Revised: 5 January 2007 / Accepted: 27 February 2007 / Published online: 20 June 2007

Abstract The acquisition of science data in space applications is shifting from teleoperated data collection to an automated onboard analysis, resulting in improved data quality, as well as improved usage of limited resources such as onboard memory, CPU, and communications bandwidth. Science instruments onboard a modern deep-space spacecraft can acquire much more data that can be downloaded to Earth, given the limited communication bandwidth. Onboard data analysis offers a means of compressing the huge amounts of data collected and downloading only the most valuable subset of the collected data. In this paper, we describe algorithms for detecting dust devils and clouds onboard Mars rovers, and summarize the results. These algorithms achieve the accuracy required by planetary scientists, as well as the runtime, CPU, memory, and bandwidth constraints set by the engineering mission parameters. The detectors have been uploaded to the Mars Exploration Rovers, and currently are operational. These detectors are the first onboard science analysis processes on Mars.

Keywords Surveillance · Science · Rover · MER

A. Castano (✉) · A. Fukunaga · J. Biesiadecki · R. Castano · S. Chien
Jet Propulsion Laboratory, MS 125-209, 4800 Oak Grove Dr., Pasadena, CA 91109, USA
e-mail: Andres.Castano@jpl.nasa.gov

L. Neakrase · P. Whelley · R. Greeley
Department of Geological Sciences, Arizona State University, P.O. Box 871404, Tempe, AZ 85287, USA

M. Lemmon
Department of Atmospheric Sciences, Texas A&M University, 3150 TAMU, College Station, TX 77843, USA

1 Introduction

The introduction of onboard automated image processing in space applications is changing the way rover and lander missions are operated. For the Mars Exploration Rovers (MER) mission, process automation has already proved valuable in several critical engineering tasks. For example, during the Entry, Descent and Landing (EDL) stage of the spacecrafts carrying the MER rovers Spirit and Opportunity the automated image analysis program DIMES allowed the spacecrafts to land in pre-specified landing ellipses, directing the firing of rockets to reduce the uncertainties in horizontal capsule velocity caused by winds [4,5]. Similarly, the rovers can traverse dozens of meters at a time using GESTALT, an automated hazard avoidance program that uses stereo vision to steer the rover away from rocks and steep hills. It also can keep track of its position using onboard visual odometry, more accurately than wheel odometry which is affected by slippage [6]. In this paper, we present an application that extends such automation from the strictly engineering domain to the science domain. The goal is to automatically perform processing onboard that allows the rover to select or prioritize science data according to its science value. Several previous systems have demonstrated onboard science analysis for rovers in testbeds and field testing [3,13,17,24,27].

In this paper, we describe two algorithms called the Dust Devil and Cloud Symbolic Perception of Objects for Extraction and Recognition programs (SPOTTERs), developed from scratch, with no legacy code except for the use of the Sky SPOTTER, that we previously developed for the Onboard Autonomous Science Investigation System (OASIS) [2]. For simplicity, we will refer to the dust devil and Cloud SPOTTERs simply as extractors or detectors. The programs were tested, optimized, integrated into the MER flight software and are currently operational, acquiring science data on Mars.

Recent explorations on the Martian surface have revealed an environment far more dynamic than previously believed. In particular, the atmosphere of Mars is very dynamic and clouds, winds and dust devils are events useful to understand it. In the current environment (in geological terms), dust storms are the dominant mechanism for altering the surface of Mars, and are likely to have occurred for as long as Mars has had a dynamic atmosphere and a supply of dust size particles on the surface. Windblown sand and dust erodes rocks and landforms, and is deposited as sand dunes and mantles that can be hundreds of meters thick. Martian clouds, which provide a picture of the state of the atmosphere, are a major indicator of the seasonal characteristics of the wind. Understanding the physics and geology of windblown particles is essential for unraveling the complex history of the Martian surface. Observing active dust storms and dust devils from the surface and from orbit and tracking their evolution enables this understanding.

Martian clouds and dust devils are high science value events have been the subject of considerable study. Both dust devil and cloud detection campaigns (periodically scheduled imaging sessions) have been conducted, but in general, they are rare events. For example, only around 10–25% of the images collected during cloud campaigns have clouds in them. Prior campaigns have involved collecting images at fixed times for return to Earth, resulting in an inefficient use of downlink bandwidth as the majority of images do not contain the phenomena of interest (dust devils or clouds).

Dust devils and dust devil tracks have been observed in orbital imagery of Mars from the Mars Global Surveyor, Mars Odyssey, and Mars Express spacecrafts. Dust devils were also detected in Mars Pathfinder color imagery [12,23]. Even before the landing of Spirit in Gusev crater, there was speculation that active dust devils would be seen from the rover, based on orbiter images of the landing site that showed inferred tracks left by dust devils [14]. This speculation was confirmed, although many of MER dust devils are only visible in the images after applying contrast stretching to make the dust devil feature apparent. To establish the presence of a dust devil in a sequence, the scientists used tools that rated the likelihood of the sequence to contain dust devils, without identifying them by location; no high level automated ground measurement aid existed. Once the images were flagged, the images were inspected manually.

Our approach to automated dust devil detection is to use motion detection based on background subtraction. This approach is suitable given the conditions under which our algorithms will be operating, i.e., stationary-camera surveillance with dust devils as the only feature in the scene that can have significant motion on the order of seconds. Motion detection based on basic background subtraction techniques are extensions to frame differencing, i.e., the subtraction of two frames, one of which, ideally, is an image of the background of the scene. These methods include frame difference, averages, running averages and median-based background subtractors. Although these methods suffer from the presence of ghosts (residuals of objects in motion in one image detectable in other images of the set used for the analysis) and have problems dealing with shadows, they are very fast, and are therefore frequently used for real-time applications (e.g., traffic monitoring [7], congestion detection in train stations [21]). More sophisticated (and more computationally expensive) background subtraction techniques such as eigenbackgrounds, mean-shift and mixtures-of-gaussians, are able to reduce the effect of shadows and ghosts to various degrees (Cucchiara et al. provide a survey of papers in many of these techniques [10]).

The clouds that form on Mars are thin and wispy. Fluffy cumulus-like clouds do not form. Cloud formation on Mars is much more localized than on Earth, where clouds are found around the entire globe. On Mars, the majority of the clouds occur around the equatorial zone. Considerably more study will be required to understand the formation and dissipation of clouds in the Martian atmosphere. As with the dust devils, cloud detection in Martian data taken from the surface has previously been done manually, typically employing contrast stretching. The study of clouds in the atmosphere has led to regular cloud campaigns since the beginning of the MER mission. Our approach to automating the detection of clouds is to first identify the sky (equivalently, the skyline) and then determine if there are clouds in the region segmented as sky. In contrast to the dust devil detection, this algorithm operates on individual images. The time frame over which the clouds may change significantly is too long to require the rover to remain motionless on a regular basis, which would be necessary for effective application of image differencing or other motion-detection technique.

Although there is a large body of work in the remote sensing community on automatic detection of clouds using satellite imagery, detection of clouds from the ground has not been widely investigated. There has been extensive work, however, on horizon and sky detection, as it is commonly used for rover localization and mapping, e.g., [25,26]. We review two related works on detecting the horizon in the context of planetary rovers.

Cozman et al. used a skyline detector for Carnegie-Mellon University's Viper system, a teleoperation system aimed at providing a user with easy to understand information for facilitating effective rover driving [9]. This skyline detector operates by analyzing the columns in an image, descending down each column, marking pixels as belonging to the sky until a threshold on the Gaussian smoothed image is crossed. This algorithm was designed to be a tool for teleoperation, aiding the human operators by giving them an initial horizon delineation which could then be easily refined manually by the operator. The algorithm was not developed

as part of an automatic process, and no claim is made that the algorithm as initially proposed, would be suitable for an autonomous application. In particular, this system is geared toward speed over extremely high accuracy as it is part of an interactive system. As such, the skyline detector is prone to failures when the smoothed gradient is not sufficiently clean. Specifically, the authors note that it can fail in the presence of clouds. For our application, the sky detector must operate fully automatically and, clearly, must be robust to the presence of clouds.

A second important work on sky detection aimed at planetary robotics was that of Gullick et al. [17], and later further applied by Roush et al. [24]. This work was conducted at the NASA Ames Research Center, where the horizon detector was part of an image understanding framework for onboard autonomous science. This sky detector uses active contours, in which a sequence of particles, initially located at the top of the image, arrive at an equilibrium state between (a) a gravity-like force that pushes them downwards, (b) an upward buoyant force function of the edges of the image that pushes them upwards and, (c) tension between the neighbors of each particle. This sky detector uses three strong assumptions: (a) there is only one horizon pixel per column, (b) the camera roll is small and, (c) the horizon's slope has a predefined limit of 45°. In practice, these assumptions hold in a large number of cases; however, as the vantage point of the camera is lowered, nearby rocks become part of the skyline and the first assumption breaks. This is particularly important for our case, where the rover is at a low vantage point, frequently looking at a very ragged horizon, e.g., the horizon seen from the rover while inside a crater is almost exclusively made of nearby rocks, resulting in a skyline that may violate the first assumption in multiple places. The situation of having a rover inside a crater is not only possible but desirable, as the rover has access to geological layers exposed inside the crater, e.g., Spirit has spent many sols inside craters.

Another area that has a strong interest in rapid and reliable horizon detection is the field of Unmanned Air Vehicles [11]. In this domain, the horizon detector for UAVs can take advantage of the high altitude of the vehicle and approximate the horizon to be a straight line, turning the detection problem into a line-fitting problem using as constraints statistical features of the sky and the ground. In general, this work is not applicable to the problem of detecting the horizon for planetary robotics where a detailed skyline is required, since the straight-line assumption does not hold. For example, in the case of Spirit, the scenes are rocky and the horizon is usually fractured; in the case of Opportunity, the scenes are sandy with a horizon following the curves of dunes, but seldom flat.

The algorithms presented in this paper represent a practical application of robust, well-understood computer vision techniques in a highly constrained computing environment with real-time execution requirements and limited CPU and memory resources. Despite the variety of possible scenes and environmental conditions and the relative simplicity of the algorithms, the detectors achieve the accuracies desired by the scientists on the team without requiring case-by-case tuning.

Autonomous, onboard detection of clouds and dust devils enables a significant improvement to the previous operational scenario for planetary science, as well as enabling entirely new operational scenarios. Scientists have traditionally relied on a process where large amounts of data are collected onboard and sifted for valuable science data on the ground. Previously, in order to increase the amount of valuable science data products, it was necessary to increase the amount of data that was collected and downloaded to the ground. By shifting the analysis and filtering of the data from the ground to the rover, the autonomous detectors make possible to increase the amount of valuable science data products without increasing the volume of downloaded data. In addition, because of the necessity of downloading large amounts of data in order to perform detection on the ground, current operations rely on imaging campaigns that are targeted to times when events are likely to occur. Detecting rare or unexpected events (i.e., *discovery*) is impractical under the traditional operational regime, since it would require vast amounts of data to be transmitted, much of which would not contain the sought events. In contrast, with autonomous, onboard detection, it becomes possible to frequently collect and analyze data onboard, sending only the pertinent data back to the ground, thus enabling the discovery of rare and unexpected phenomena.

In this paper, we discuss the operational scenarios under which the algorithms are designed to be used and then describe the dust devil and cloud detection algorithms. We then provide results of performance testing for detection accuracy, CPU and memory usage, and rates of compression achieved.

2 Current and future scenarios

Martian dust devils and clouds are dynamic events that provide direct and indirect information about the Mars atmosphere [1,16,20]. Scientists wish to collect as much data as possible on these events to better understand them and the role they play in Martian weather. Currently, due to the restricted number of images that can be transmitted to Earth, campaigns to capture the events using the MER cameras are scheduled for times when the events are expected to occur. This allows the rovers to perform a documentation and verification task but not one of exploration or discovery. In the case of clouds, sequences of images of the Martian sky are taken at specific times and transmitted to Earth

with the expectation that some of them may have captured the phenomena. Presently, with a current success rate of imaging clouds between 10% and 25% of the time, the cloud campaigns are expensive in terms of bandwidth and rover resources. To improve the use of bandwidth, the scientists may download a thumbnail of the full resolution image, prior to deciding to download the actual image. Thumbnails are 8-bit 64×64-pixel (i.e., 4Kb) images that provide a rough summary of the actual image. However, for the case where the images are being searched for clouds, the thumbnails are not useful unless the image has a very evident cloud i.e., wispy and soft clouds most likely would be missed. In Fig. 1a we can see the thumbnail of an image that contains an evident cloud. In the case of dust devils, a campaign consists of acquiring a sequence of around 20 frames. The first, middle and last images of the set are sent to Earth, at high resolution, and based on those, a decision for the download or deletion of the full set is made. Thus, dust devils present in the sequence that fail to appear in one of the three downloaded frames are lost. Likewise, the full sequence is downloaded at a high bandwidth cost, despite the fact that the dust devil might have appeared in only a few of the frames. Again, the thumbnails are not useful for resolving the majority of dust devils which are medium-size, small or faint.

For the future scenario, the dust devil and the cloud detectors were designed to improve the quality of the science acquisition of these events, under the most general of conditions. The detectors were designed for MER, but the software and algorithms are applicable to any acquired surface imagery. Efforts are in progress to deploy the software to several future planned and proposed surface missions to Mars. Both detectors use gray-scale imagery (as opposed to color or multi-spectral) so they can use imagery from the panoramic, navigation or hazard cameras of the rovers. The cloud detector determines whether a single image contains clouds. If it does not, the image is deleted; otherwise, the program creates a binary thumbnail that shows the skyline and the detections. These binary thumbnails, in contrast to the standard thumbnails available without the detectors, highlight only the detected clouds, as shown in Fig. 1 b. The binary thumbnails have a resolution of 64×64 pixels and a depth of 1 bit for a size of 512 bytes so they can be downloaded without any appreciable cost. The scientists can either download the binary thumbnail to have additional information about the image or directly download the high-resolution image, which has already been deemed by the detector to have a high probability of containing a cloud.

For dust devil detection, we developed two operation modes. In the first mode, called *all-in-one*, we detect dust devils in a sequence of 4–8 images, all stored in memory at the same time. If a dust devil is present, a set of binary thumbnails that summarizes the changes in the images is prepared. The scientists can download the binary thumbnails

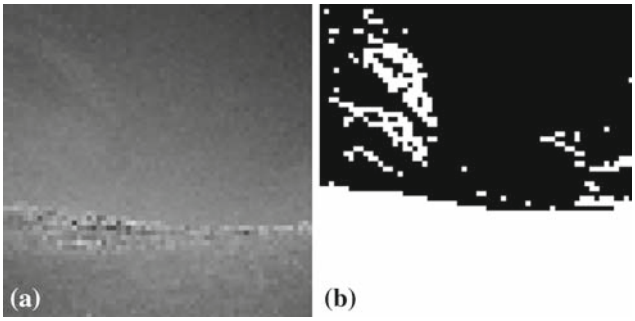


Fig. 1 **a** Standard and **b** binary thumbnails of an evident cloud

to have additional information about the sequence or directly download the images, which have already been deemed by the detector to have a high probability of containing a dust devil in at least one of the frames. In the second mode, called *feed*, we process large image sequences by having in memory one image at a time, i.e., change detection is carried out updating history records. In this case, individual frames are tagged as containing a dust devil or not and can be downloaded individually. As before, each full resolution image is accompanied by a binary thumbnail that can be downloaded to verify the contents of the full resolution image or to prioritize its download. Having discussed the operational scenarios under which the detectors will be run, we now describe the detection algorithms.

3 Image conditioning

The first step for both dust devil and cloud detection is to condition the images. The image conditioning operations performed are: border cropping, a single-step image filtering, minimization and bit-depth reduction, and noise estimation, as shown in Fig. 2. The first conditioning operation, border cropping, eliminates the effects introduced by the frame grabber and ensures that they do not affect the image statistics. While the size of the border to crop is a parameter, a typical value is five pixels per side. Border cropping is also used to restrict detections to a portion of the image. For example, dust devils detection is typically restricted to the top 256 rows of the full resolution 1024×1024 pixels image.

The second conditioning operation simultaneously low-pass filters the image, reduces its size and reduces its bit depth. Its net effect on the algorithm it to reduce both execution time and image noise. This is the most expensive step of both detectors, taking between 20 and 70% of the total execution time (depending on the reduction factor), and thus we will describe it in some detail.

MER images have a native resolution of 1024×1024 pixels, stored in 2 bytes per pixel to accommodate a depth of 12 bits per pixel [22]. The onboard processor, a 20 MHz

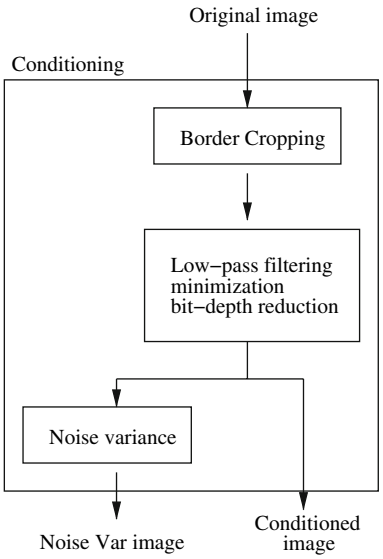


Fig. 2 Conditioning process of cloud and dust devil campaign images

RAD6000, is not fast enough to permit an analysis of the full resolution image in a reasonable amount of time (under 30 s). Our solution is to analyze the image at lower resolutions. This approach is used by other automated processes onboard, including hazard avoidance.

Pixels of the minimized image are obtained as follows. The bit-depth resolution of non-overlapping windows of the border-cropped images is reduced from 12 to 8 using a linear rescaling, i.e., the min–max range of the original image is remapped onto the 0–255 range of the second image. The results of low-pass filtering these windows are used as the new pixels. Although the type of low-pass filter can be set to either a mean filter or a median filter (or even removed completely, reducing the operation to one of subsampling), in practice we use the median filter almost exclusively. Although more computationally expensive than a mean filter, the median filter is necessary to reduce the salt-and-pepper type image noise that can cause false identifications of features, in particular for the image differencing approach used for dust devil detection. The width, k, of the $k \times k$ median filter window is a selectable parameter, where $3 \leq k \leq 7$.

The median filter was implemented using a counting sort, taking advantage of the fact that the image pixels have been reduced to a length of 1-byte. The algorithm employs a binning array, T, of length 256. For each pixel in the filter window, A, the count in the bin corresponding to the pixel value is incremented. The median is the index for which the cumulative bin sum is equal to or larger than half the number of pixels in the filter window. On exit, the binning array is reinitialized to zero, to prepare it for the next call to the routine. The pseudo-code of the algorithm is:

```
UCHAR- MEDIAN (A(1 : len), T(0 : 255))
1   sum ← 0, min ← 255, max ← 0
2   for i ← 1 to len
3     T[A[i]] ← T[A[i]] + 1
4     if (A[i] < min) min ← A[i]
5     else if (A[i] > max) max ← A[i]
6   for i ← min to max
7     sum ← sum + T[i]
8     if (sum ≥ len/2) median ← i; break
9     T[i] ← 0
10  for j ← i to max
11    T[j] ← 0
12  return median;
```

This implementation of the median filter runs in time linear in the sum of the lengths of the input array A and the distance between the minimum and maximum values of input array D. In our case, where the window size varies from 3 to 7, the length of the input array A varies between 9 and 49. The distance D will depend on the image, both on its noise and its texture. The liner median filter has low overhead in terms of both memory and computations, but its benefits may not be apparent for very small windows. In our experiments, this implementation was comparable in speed to a median filter using an insertion sort for windows of size 5×5. For smaller size windows, the median filter based on the insertion sort was slightly faster than the linear median, but for larger windows, the linear median filter was substantially faster.

This algorithm is appropriate for our application because the size of the binning array is small (256 elements). Our experiments showed that it outperformed median filter implementations with sorting routines based on comparisons. However, because the performance depends on the length of the binning array, the performance degrades significantly if the elements to be filtered require more bits to represent. A second reason to use this type of filter is that, in our application, consecutive calls to the filter do not share values and thus, we cannot make use of partially sorted arrays. The output of each filtered window of the original image yields a single pixel in the minimized image and since the windows do not overlap, there is no history or shared information that can be exploited. The third reason is that the filter must support input arrays of various sizes, i.e., the size of A is defined by the minimization factor k. Hence, we did not use efficient median filter implementations that exploit specific sizes of the filter window.

The second goal of image minimization is to increase the signal-to-noise ratio (SNR) of the minimized image as compared to that of the original image. Taking a mean filter as a lower bound for the median filter, we have a reduction in the standard deviation of the noise of the order of $1/k$. Furthermore, although the median filter is more expensive than

1	-2	1
-2	4	-2
1	-2	1

Fig. 3 Noise estimation operator based on difference of Laplacians

the mean filter, it eliminates (for all practical purposes) the effects of the dead pixels of the sensor on the minimized image.

The last step of the image conditioning shared by the dust devil and cloud detectors is the estimation of the variance of the noise of the minimized image. This will be used as a confidence value to bias the detection thresholds. In all cases, it is estimated using a difference of Laplacians [18] convolving the minimized image with the kernel shown in Fig. 3. The selection of this method to estimate the noise variance was also motivated by its speed.

4 The dust devil SPOTTER

All dust devil and clouds campaigns are carried out while the rover and camera are not moving. In this static scene case, the two common methods for detecting dust devils are the comparison of two or more spectral bands of the scene (particularly using red and blue ratios) and the detection of change in a sequence. We decided to base the dust devil detector on change detection in gray-scale sequences so that it could be applied to panoramic, navigation and hazard camera imagery; the spectral method could have been used only with imagery from the panoramic cameras (the only ones equipped with a filter wheel). In theory, detecting change in the scene is not equal to detecting dust devils as Martian clouds also have motion. In practice, if image noise can be accounted for, the vast majority of hits of a change detector in a static Martian scene are caused by dust devils.

We based our change detector on background subtraction because of its speed. The computational requirements eliminate the use of optical flow or other more sophisticated techniques; furthermore, given that the scene is static, they are unnecessary. Within the confines of background subtraction, the requirements favor the basic methods that use frame difference, averages, running averages and medians. We selected the fastest possible methods that allowed us to obtain the hit-to-miss ratios required by the scientists in the team. Since the events tend to be faint and difficult to detect, the hit-to-miss ratios are strongly correlated with the handling of the image noise. For the following analysis, we will approximate the noise of a pixel by an additive noise ϵ drawn from a random Gaussian zero-mean distribution with a standard deviation σ, and regard the multiplicative noise component of the pixel as negligible. Subindices indicate the frame of the sequence associated with the variable.

4.1 Frame difference

Detecting changes between two images is largely reduced to subtracting them and thresholding the result. Consider the effects of frame differencing on a single pixel (x, y). Let us assume that there was no motion in the scene in image $n-1$, i.e.,

$$I_{n-1}(x, y) = \mu + \epsilon_{n-1} \tag{1}$$

where μ, the mean of the pixel, is the expected value of the pixel under noiseless conditions. Let us further assume that there is a change in the scene in image n (e.g., a dust devil passing by) that changes the value of the pixel by δ, i.e.,

$$I_n(x, y) = \mu + \delta + \epsilon_n \tag{2}$$

Then,

$$I_n(x, y) - I_{n-1}(x, y) = \delta + (\epsilon_n - \epsilon_{n-1}) \tag{3}$$

Since the noise components ϵ_i are independent samples, the variance of their difference is the sum of their variances, i.e., the standard deviation of the difference is $\sqrt{2}\sigma$. Defining the SNR of the pixel as the ratio of its mean to its standard deviation we have

$$\mathrm{SNR}_{fd} = \frac{\delta}{\sqrt{2}\sigma} \tag{4}$$

If we know a priori that the moving objects have a high contrast (e.g., a bright dust devil moving over a dark Martian scene), then the difference of frames produces a difference in intensity that can be thresholded with confidence very rapidly. However, the method fails for the detection of faint dust devils, where the magnitude of the changes is closer to the noise of the image, as thresholding frequently will incorrectly classify image noise as change (false positive), and/or classify actual change as noise (false negative). Since most of the dust devils that have been found on Mars are faint and can only be observed by a person when the sequence is animated (usually requiring equalization or other remapping) the threshold cannot, in general, be selected reliably using the frame difference method.

4.2 Difference of average and frames

The detection of faint dust devils requires a better SNR than that provided by the frame difference method. A better method is that of acquiring a low-noise background image through temporal averaging and subtracting it from the image

that we want to analyze. For example, assume that the average of a pixel over $n-1$ images of a static background is

$$A_{n-1}(x,y) = \frac{1}{n-1}\sum_{j=1}^{n-1} I_j(x,y) \tag{5}$$

which has a mean μ and a standard deviation $\sigma/\sqrt{n-1}$. Thus, if pixel $I_n(x,y)$ has a change δ, we have

$$A_1(x,y) = I_n(x,y) - A_{n-1}(x,y) \tag{6}$$

$$= (\mu + \delta + \epsilon_n) - \frac{1}{n-1}\sum_{j=1}^{n-1}(\mu + \epsilon_j) \tag{7}$$

which has a mean δ. Since the noise components of A_{n-1} and I_n are independent, then the standard deviation of their difference is the sum of their variances, i.e.,

$$\sigma_{A1} = ((\sigma/\sqrt{n-1})^2 + \sigma^2)^{\frac{1}{2}} \tag{8}$$

$$= \frac{\sigma\sqrt{n}}{\sqrt{n-1}} \tag{9}$$

so the SNR ratio of the difference is

$$\mathrm{SNR}_{A1} = \frac{\sqrt{n-1}}{\sqrt{n}}\frac{\delta}{\sigma} \tag{10}$$

The SNR of the difference of the average and frames reduces to that of the frame difference in Eq. (4) for $n = 2$ but for n large it tends to

$$\mathrm{SNR}_{A1}|_{n\to\infty} = \frac{\delta}{\sigma} \tag{11}$$

with a gain of $\sqrt{2}$ over the frame difference method, i.e., a maximum gain of a 41.42% as n increases.

The added benefit of increasing n, the number of frames in the average, is inversely proportional to n. The relative gain of differencing a frame from an average over that of differencing only two frames is

$$\frac{\mathrm{SNR}_{A1}}{\mathrm{SNR}_{\mathrm{d}}} = \sqrt{2}\sqrt{\frac{n-1}{n}} \tag{12}$$

A second interesting figure is the value of n needed to achieve a sizeable percentage of the maximum possible gain in SNR of $\sqrt{2}$, i.e.,

$$\mathrm{gain} = \frac{\mathrm{SNR}_{A1} - \mathrm{SNR}_{\mathrm{d}}}{\mathrm{SNR}_{A1}|_{n\to\infty} - \mathrm{SNR}_{\mathrm{d}}}100\% \tag{13}$$

$$= \frac{\sqrt{2(n-1)} - \sqrt{n}}{\sqrt{2n} - \sqrt{n}}100\% \tag{14}$$

As expected, both Eqs. (12) and (13) are an exclusive function of n, independent of the standard deviation of the distribution σ. These figures are tabulated for various values of n in Table 1.

Table 1 Relative increase in gain of difference of average and frames with respect to frame difference and percentage of maximum improvement of difference of average and frames over frame difference for a given number of images n use in the average

n	$\mathrm{SNR}_{A1}/\mathrm{SNR}_{\mathrm{d}}$	Gain %
2	1.000	0.0
3	1.155	37.5
4	1.225	54.2
5	1.264	63.8
6	1.291	70.3
7	1.309	74.5
8	1.323	78.4
9	1.333	80.6
10	1.342	82.5
20	1.378	91.6
50	1.400	96.6
100	1.407	98.2

The rover has the constraint of having only ten image buffers; sequences larger that ten images require the compression and storage of an image (or its deletion) before a new one can be captured. Thus, for the all-in-one mode, ten images is the absolute maximum sequence length that we can have (larger sequences require a flushing of buffers, making the images unavailable for onboard processing). Still, according to Table 1 this is a perfectly reasonable value as using 4–8 images in the average already provides from 50 to 80% of the maximum possible benefit over frame difference to be gained using averages.

The difference of averages and frames has an asymmetry in that it assumes that the average image A_{n-1} is composed of images of a static scene. This is a simple assumption in surveillance tasks where the camera can survey the scene for a long period of time and acquire a good estimate of the background with no foreground objects. However, in our application, a sequence of 4–8 frames is all that we have available and we do not know which frames do not contain changes. Thus, we restore the symmetry of a sequence of n frames by finding the changes in image I_i using the difference between A_n, the average of the n images of the sequence, and $A_{n,i}$, the average of the $n-1$ images of the sequence that excludes I_i, i.e.,

$$A_n - A_{n,i} = \frac{1}{n}\sum_{j=1}^{n} I_j - \frac{1}{n-1}\sum_{j=1,\ j\neq i}^{n} I_j \tag{15}$$

In practice, $A_{n,i}$ is calculated as an update of A_n:

$$A_{n,i} = \frac{n\,A_n - I_i}{n-1} \tag{16}$$

Since A_n and $A_{n,i}$ share terms, they cannot be considered independent variables and thus, it is not easy to derive a closed form expression for the SNR of their difference. However, we have verified experimentally that it has the comparable limits and gains to those of the standard differencing of averages and frames. The feed mode is treated in the same manner, replacing the averages with running averages, usually of length $n = 6$. The use of background subtraction based on average and running averages were sufficient to satisfy the hit-to-miss requirements, so no further methods were analyzed. In Fig. 4 we show the difference of averages for a frame of a 7-image sequence, stretched between 0 and 255 for display purposes. The main dust devil of this image, on the right, appears prominently, while ghosts from other frames are apparent on the top left of the image.

Once we have performed the background subtraction, we proceed with the thresholding. Since noise is a function of parameters over which we have no control, such as time of day, orientation of camera with respect to the sun, etc., a fixed threshold is bound to perform correctly only under a narrow set of conditions. In Fig. 5, we present the histogram equalization of the difference of averages shown in Fig. 4, which shows that the local noise (in a small neighborhood of a 3×3 pixel size) of the image has fairly homogeneous statistics except at the location of the dust devils, ghosts and the horizon line. We take advantage of this by estimating the variance of the local noise of the difference of averages image and then biasing the detection threshold as follows:

$$\text{thres}_{dd} = k_1 \left(1 + \frac{1}{k_2 {\sigma_{ln}}^2}\right) \tag{17}$$

where k_1 is the unbiased threshold under noiseless conditions and k_2 defines the influence of the local noise variance on the final threshold. We use a value of $k_1 = 14$, found experimentally; this figure gives an idea of the narrow set of gray-scale values that is used to discriminate between a dust devil and background noise. Higher noise variances, that occur at locations where the estimate of the noise variance breaks (at edges and textured foreground objects) indicate the presence of a dust devil so we lower the threshold to reduce false negatives; lower noise variances indicate areas of the image where there were no changes so we increase the threshold to avoid false positives. In essence, we are using the variance of the noise of the difference of averages as a second indicator of the presence of foreground objects. Since noise is out of our control, this indicator is less reliable than the difference of averages and thus, simply biases the main threshold. Finally, since the image of a dust devil is bounded within a small area of the image, we use a blob filter to make sure that there are more than a given number of detections within a local region. The threshold criteria for the blob filter is to preserve pixels belonging to blobs with a radius of 2 pixels that contain at least 2 pixels.

Fig. 4 Difference of averages for a frame of the 7-image sequence

Fig. 5 Histogram equalization of the difference of averages image shown in Fig. 4

4.3 Representative example

The detections of the last five frames of a 7-image sequence where the all-in-one mode was used is shown in Fig. 6. These images show a single very evident dust devil moving from left to right. In the last frame a second dust devil, near the center of the image, has also been captured. In addition to the dust devils, the figure illustrates the effect of dust devil *ghosts*. In the all-in-one mode, all the images are analyzed simultaneously, using their average. Hence, the effects of a high-contrast dust devil in any of the images is reflected in all the other images, creating large image changes at locations where there actually is a dust devil in a different frame. Thus, for example, the presence of the left-most dust devil of the first frame of Fig. 6 creates ghost images in the other frames. On the other hand, the image of the dust devil in the fourth frame of Fig. 6 is soft enough that it does not create ghosts in any of the other frames. For this application, where computational resources are low and where the primary objective is detection (as opposed to tracking), it is not worthwhile to incur in the additional cost of removing these ghosts. In fact, the presence of ghosts is as good an indication of motion as the dust devils themselves. In addition, the effect of ghosts is limited to a few frames around the location of the dust devil, either because we are using the all-in-one mode, restricted to 4–8 images, or because the running average of the feed mode restricts the influence of a dust devil to the number of frames used in the averaging (usually 6).

The detection of dust devils in a sequence is encoded in binary thumbnails. In Fig. 7 we show the seven binary thumbnails that correspond to the image sequence that includes the images in Fig. 6. The sequence shows the main dust devil traversing the scene; the second dust devil, which appears in the last frame is somewhat shadowed by the ghost of the

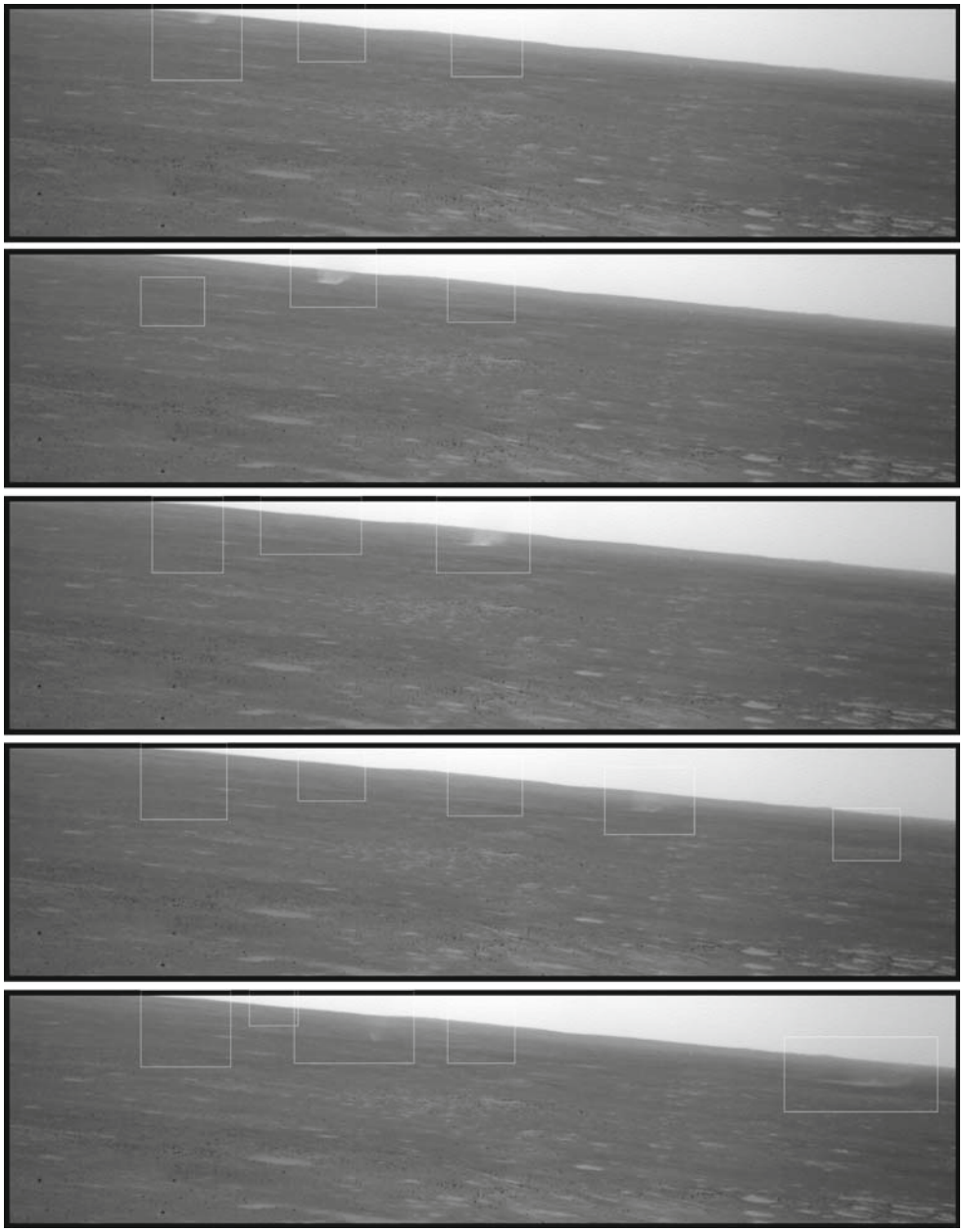


Fig. 6 Detection of dust devils. Images 3–7 of a 7-image sequence containing two dust devils

image of the first dust devil in the fourth frame. In spite of the presence of ghosts, the binary thumbnails show that there is structured motion in the scene and therefore dust devils are present. This sequence provides the scientists with information that can be used to verify that the detector captured a dust devil event and to prioritize its downloading. Given the small size of the binary thumbnails, they can be also used as a cost-effective method to tune and/or debug the detector once it is under Mars environmental conditions.

4.4 Compression

The images sent to Earth from MER are compressed using the ICER wavelet compressor [19]. ICER uses two compression criteria: image quality (proportional to bits per pixel) and compressed image size; the compression stops when either criteria is met. Most images downloaded from MER have a quality that lies between 1 and 2 bits per pixel which, for the 7-image dust devil sequence of 1024×256 pixel images that

Fig. 7 Binary thumbnails of 7-image sequence with two dust devils. The thumbnails have been resized to match the proportions of the images in Fig. 6

we are using as an example, yields a total size of between 1.75Mb and 3.5Mb, depending on the number of bits per pixel.

For dust devils images, we can further increase the compression ratio using the fact that the rover acquires images when it is motionless and that they are acquired within the span of a few minutes, where the statistical characteristics of noise and scene illumination can be considered constant. In this case, we can download the first image of the sequence and the detected changes of the subsequent images; the sequence is recovered by pasting these changes onto the first image. We reduce the information of these subsequent images masking out the areas without changes thus allowing ICER to encode the masked-out region using a small number of bits. Figure 8 shows the masked version of the last image of the sequence shown in Fig. 6.

Transmitting full resolution masked images, instead of transmitting a series of small cropped images, allows us to treat all the images of the dust devil campaign as standard MER images. Additional compression ratios due to masking depend on the size of the masked area. As an example, the size of the compressed masked image in Fig. 8 is about 75% smaller than that of the unmasked version of the same image, while preserving the same image quality in the areas of interest, for a total compression ratio that lies between 0.5 and 0.25 bits per pixel. Taking this rate as representative of the example sequence, we could download the full 7-image example sequence using only 1.25Mb to 625Kb (one unmasked and six masked images). For the feed mode, which might operate for periods of time long enough to cause the environment conditions to change, it is possible to force the download of additional full resolution images at predefined intervals of time or frames.

5 The cloud SPOTTER

Cloud detection poses difficulties similar to those of the dust devil detection. While clouds that can be easily detected by a person can also be detected easily by an algorithm, faint clouds with values approaching the noise levels of the image, are difficult to detect automatically. Unfortunately, the vast majority of clouds on Mars fall into the latter class. The set of images from MER available for the design of the cloud detector consisted, not of long sequences like in the dust devil case, but of single images or sequences of up to 3 images each. This prevents a motion detection approach to cloud detection. Furthermore, motion detection would have had only partial success as the motion of the clouds is a function of the wind, i.e., even evident clouds might not exhibit detectable motion under low wind conditions. Thus, the cloud detector is based on the analysis of single gray-scale images.

5.1 The sky SPOTTER

The first step to analyze the sky is to segment it from the ground. In related work, this type of image segmentation has been used by Cozman et al. as a precursor for mountain detection, to use as a guide for rover pose estimation [8]. The sky detector consists of three steps. First, we place seeds in a window that slides along the top of the image whenever the homogeneity of the window is high. Then we grow the seeds into a region S (not necessarily connected) using the edges of the image as the stop condition. The edges of the image are estimated using a Sobel edge detector. Finally, each pixel located above a pixel in S whose value exceeds a threshold is also labeled as part of S; this adds into S the sky regions with high noise and clouds that the growing algorithm might have missed. For example, the image shown in Fig. 9a yields the sky mask in Fig. 9b. During this process, we determine the skyline (the pixels at the interface of the sky and the ground) and the horizon (the image row below which there are no

Fig. 8 Masked product that corresponds to the last image of the sequence shown in Figs. 6 and 7

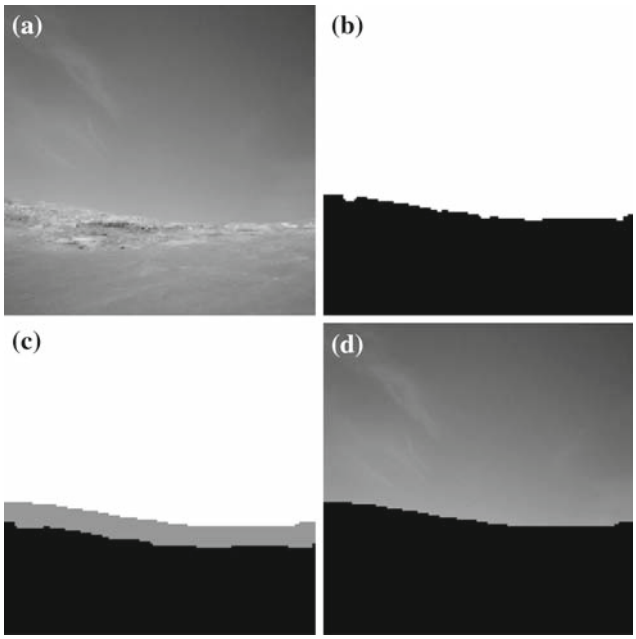


Fig. 9 Example of sky detection on an image with evident clouds: **a** minimized image, **b** sky mask, **c** skyline buffer and **d** segmented sky

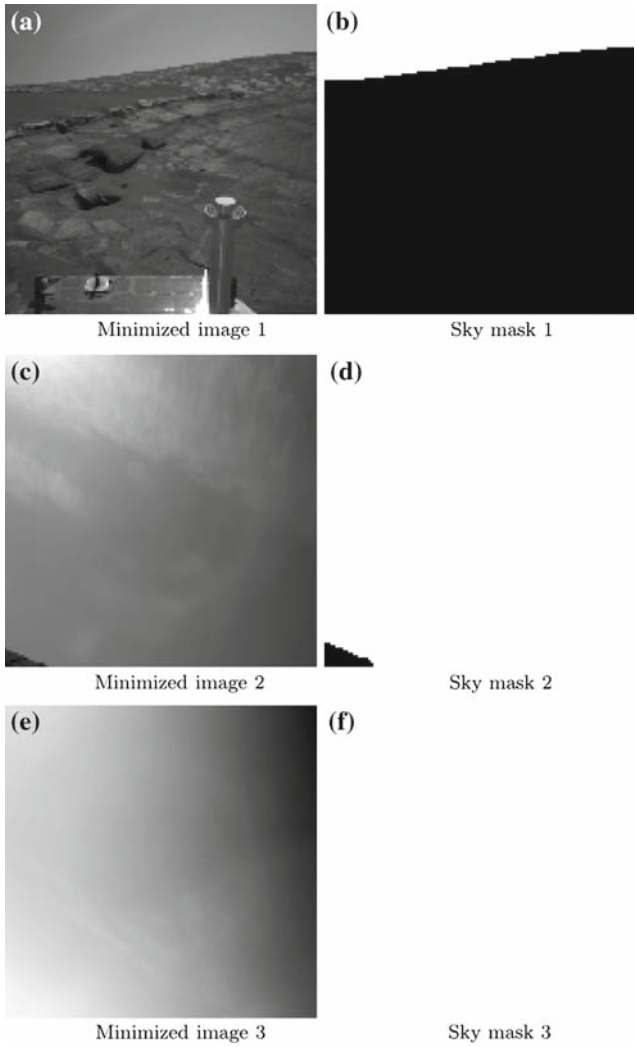


Fig. 10 Examples of sky detection: **a–b** dark sky, **c–d** cloudy sky, **e–f** no ground

sky pixels). We create a buffer around the skyline, which can be used to eliminate undesired intensity gradients at the sky-ground interface. The buffer area of the skyline of Fig. 9a is shown in Fig. 9c. The final result of masking the original image with the sky mask (which includes the buffer) is shown in Fig. 9d.

The sky detection process is automatic, fast and robust to a number of conditions. Internally, it adapts to different image reduction factors; externally, it adapts to different environmental conditions like noise, gradients of illumination and dusk and dawn. Likewise, it is not affected by image artifacts caused by lens inter-reflections, or by image artifacts caused by cosmic rays, which manifest themselves on the image as spurious shot noise. It can discern sky from ground even in cases where the interface between them is fairly soft, but it does fail when this interface is not clearly defined, as in the case where the horizon is saturated because the camera is aimed toward the sun. Figure 10 shows the sky masks, already buffered about the skyline, for three images of Martian sky that exemplify the cases of dark sky, cloudy sky and no ground visible. The performance of the sky detector is close to 100% correct, which decouples the subsequent performance of the cloud detector from that of the sky detector.

5.2 Cloud detection

Once the sky has been segmented, the cloud detector searches for variations in the sky region using a Sobel edge detector; strong edges indicate large gradients on the sky that are

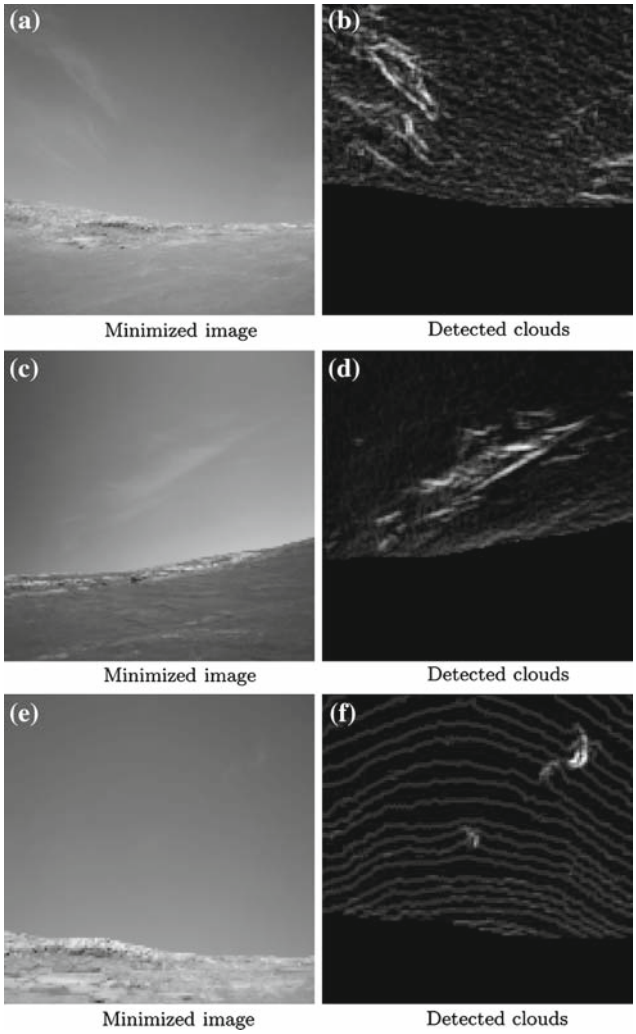


Fig. 11 Examples of cloud detection on images with **a**–**d** evident clouds, and **e**–**f** wispy clouds

caused by the presence of clouds. Examples of images with evident and wispy clouds and the corresponding detected clouds are shown in Fig. 11.

The threshold of the value of an edge that corresponds to a cloud is weighted by the noise of the image. In this case, the threshold is defined as

$$\text{thres}_{\text{cloud}} = k_3 \left(1 - \frac{1}{k_4 \sigma_{\text{sky}}}\right) \tag{18}$$

where k_3 is the threshold for the noiseless case and k_4 controls the amount of bias due to noise. In our case, we used a value of $k_3 = 10.9$ found experimentally, which indicates that for the training set the cloud detector has better discrimination characteristics than the dust devil detector, i.e., it is easier to detect a faint cloud against a homogeneous textureless sky than it is to detect a faint dust devil moving against a heterogeneous textured ground.

Although both Eq. (17) and Eq. (18) bias the detection thresholds using noise and have basically the same form, they function in a different way. First, in the dust devil case, the noise estimator breaks in the presence of edges created by the motion on the scene, which allow us to use it as a secondary cue about the presence of objects. Hence, we are using the noise as an estimator of local structure so the variance of the noise used to bias the threshold is local. For the clouds, the noise estimator does not break as there are no strong edges in the sky area and thus, the noise itself does not give any indication of the structure of the image and it tends to be homogeneous over the sky region. Hence, the noise used to bias the cloud threshold is the average noise of the whole sky area. Second, in the dust devil case, the noise is estimated over the background subtracted image and thus the biasing of the threshold is inversely proportional to the noise: high noise variance is a secondary indication of image edges so we lower the threshold to avoid false negatives. For the cloud detector case, the noise is estimated over the image itself, so high noise simply indicates unreliable data and thus, the bias should be proportional to the noise: high noise variance indicates unreliable data so we raise the threshold to avoid false positives.

The cloud detector, in contrast to the sky detector, can fail in the presence of many effects because it assumes that any large changes in the sky correspond to the presence of clouds. For example, vignetting may cause gradients at the corners of the image, which the detector can mistake for clouds. Vignetting effects are produced by a low exposure due to a low camera gain or an incorrect shutter speed. They also appear naturally under low light conditions (e.g., at dusk and dawn). A second cause of failure is shot noise caused by cosmic rays. This causes a local change of intensities in pixels that is often large enough that it is not filtered by the low-pass filter, particularly in the case where the reduction factor is low, e.g., 3 or 4. However, it is not cost-effective to correct it as this effect is rare (observed in around 1% of the images of our test set) and would be computationally expensive to compensate. Finally, lens inter-reflections can cause image effects that can be mistaken for clouds. These effects increase in number and intensity as the angle between the viewing direction of the camera and the sun narrows. One such effect (a bright halo centered on the image) can be seen in Fig. 10c and e.

The cloud detection step is followed by the creation of a binary thumbnail that can be downloaded to verify detection or prioritize the download of the image. For example, the detected clouds in Fig. 11b are summarized in the binary thumbnail shown in earlier Fig. 1b; the same clouds are more difficult to observe in the standard thumbnail shown in Fig. 1a. Finally, if the image is selected for download, the scientist can download the complete image or the area of the image above the horizon line, increasing the compression ratio of

the image. In the later case, the area of the image below the horizon line is masked out as described for dust devils (see Sect. 4.4). The additional gain in compression depends on the proportion of sky vs. ground in the image.

6 Results

The engineering constraints are determined by the rover resources and the time allocated to the task. Presently, the resources consumed by a search campaign are image acquisition time (e.g., about 20 s per image), the onboard memory occupied by the images while the scientists decide on the science value of the set, and the bandwidth used by the down-link. Tests on the Surface System Testbed (the SSTB—a rover functionally identical to the MERs, with a 20 MHz RAD6000 CPU) show that, excluding image acquisition, the running time of the cloud detector is under 20 s per image and that of the dust devil detector is approximately 15 s per image in all-in-one mode; in feed mode, after the overhead needed to set the running average, the analysis of each new image takes 10 s. For both the clouds and the dust devils, the full size images are stored onboard; however, unlike the present campaigns, it is possible to gray-out the portions that do not contain science data (see Sect. 4.4). Since the images are stored on board after compression, there is a decrease in both storage space and bandwidth usage, while preserving the original size of the image and the resolution of area of the image that contains the science event.

The science constraints were set by the scientists who are currently carrying out the dust devil and cloud search campaigns. Their specifications were a hit ratio above 80%, a false positive ratio of 20% (low, to reduce the downloading of images that do not have science events) and a false negative ratio of 10% (very low, to avoid missing potentially valuable images). These ratios had to be achieved for a test set of their selection, which contained Mars images acquired during the campaigns, under a variety of conditions in terms of science content, noise, relative direction of the sun, time of day, resolution and level of compression.

More than 500 active dust devils have been documented by the MER Spirit, mostly using the Navcam frames [15]. Although the features are difficult to detect, image processing techniques were used to enhance the dust devils and to derive detail. In many cases, sequences of frames were obtained, which enabled animations of active dust devils to be constructed, from which velocities and changes in dust content were derived. The result of this image classification effort was used as the ground truth for the development of the dust devil detector. The dust devil detection algorithm was tested on 385 images composed of 25 image sequences, all acquired from the left navigation camera of the Spirit rover. Each sequence was 6–20 images long. The set of sequences was biased toward faint dust devils, i.e., barely perceptible to the naked eye, even when animated; many of the dust devils could not be seen without processing the sets. Given these sequences, we analyzed all contiguous 4, 6, and 8 image subsets using the all-in-one mode. A true positive is scored solely on whether or not the image contains dust devils, e.g., we are not requiring the location to be correct. The results are presented in Table 2.

Table 2 Results from the dust devil detection algorithm

No.	Tuples	+	−	Correct	False −	False +
4	279	120	159	237 (84.9%)	10	32
6	228	121	107	190 (83.3%)	14	24
8	180	116	64	155 (86.1%)	10	15

Table 3 Results from the cloud detection algorithm

Type	No	Correct	False −	False +
Evident cloud	29	29 (100%)	0	0
Wispy cloud	13	13 (100%)	0	0
Uncertain	5	3 (60%)	3	0
No cloud	163	152 (93.2%)	0	11

The ground truth used for the classification of clouds was obtained by an atmospheric science student supervised by one of the authors, who reviewed all the candidate images. Using manual contrast adjustment and filtering he created a catalog of all the images that contained clouds. The cloud detection algorithm was tested on 210 images, composed of 47 images with clouds and 163 images with sky but without clouds. Of the images with clouds, 29 contained evident clouds, 13 contained soft, hard-to-see wispy clouds, and 5 were ‘borderline’ cases where scientists could not decide with certainty whether a cloud was present. The results are presented in Table 3.

7 Conclusion

We have described two detectors aimed to increase the science return of the Mars Exploration Rover campaigns that search for dust devils and clouds. The design of these extractors was constrained by the limitations of the rovers (memory, speed and bandwidth) and by the accuracy goals

provided by the MER participating scientists in the team. In addition, we used the intrinsic characteristics of the tasks (e.g., static rover and camera) and the Martian environment to generate assumptions that helped in the determination of the algorithms to use, e.g., everything that moves is a dust devil, any anomaly in the sky is a cloud. We illustrated both algorithms using representative examples and presented the results of their performance with respect to runtimes, bandwidth usage, and accuracies achieved, all of which satisfy the quantitative requirements given by the users (scientists) and mission operations management. These programs have been tested on Mars and are currently part of the operation commands available to the scientists.

In this paper, we have focused on relatively simple cloud and dust devil detection algorithms that operate successfully given the severe CPU and memory constraints aboard the Mars Rovers. Investigation of more sophisticated techniques (e.g., machine learning-based detectors) to determine the possible algorithms for cloud and dust devil detection on future missions, with possibly less restrictive resource constraints, is a promising direction for future research.

Acknowledgements We thank Aaron Kiely for providing the ICER code used to estimate the compression ratios of masked detector products and Adnan Ansar for his help in characterizing the results of the difference of averages of the dust devil detector. The research in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. This work was supported by NASA as follows: algorithm prototyping and testing by the Software Intelligent Systems and Modeling (SISM) program under the OASIS task, algorithm refinement and flight code integration by the Mars Technology program (MTP) under the MER On-board Science task and, further algorithm refinement and MER testing and operation by the New Millennium Program (NMP) under the MER Infusion Studies task.

References

1. Balme, M.R., Whelley, P., Greeley, R.: Mars: dust devil track survey in Argyre Planitia and Hellas Basin. J. Geophys. Res. **108** (2003). 10.1029/2003JE002096
2. Castano, R., Anderson, R.C., Judd, M., Estlin, T., Gaines, D., Castano, A., Bornstein, B., Wagstaff, K., Stough, T.: Recent advances: onboard autonomous science investigation system. In: EOS Trans. AGU, vol. 85(17), Fall Meet. Suppl., Abstract P43A-0913 (2004)
3. Castano, R., Estlin, T., Gaines, D., Castano, A., Bornstein, B., Stough, T., Chouinard, C., Judd, M.: Opportunistic rover science: finding and reacting to rocks, clouds and dust devils. In: Proc. IEEE Aerospace Conference. Big Sky, Montana (2006)
4. Cheng, Y., Goguen, J., Johnson, A., Leger, C., Matthies, L., SanMartini, M., Willson, R.: The Mars Exploration Rover descent image motion estimation system. IEEE Intell. Syst. **19**(3), 13–21 (2004)
5. Cheng, Y., Johnson, A., Matthies, L.: MER-DIMES: a planetary landing application of computer vision. In: Proc. IEEE Conf. Computer Vision and Pattern Recognition, San Diego, CA, vol. 1, pp. 806–813 (2005)
6. Cheng, Y., Maimone, M., Matthies, L.: Visual odometry on the Mars exploration rovers. In: Proc. IEEE Int'l Conf. Systems, Man and Cybernetics, Big Island, HW, vol. 1, pp. 903–910 (2005)
7. Cheung, S.C., Kamath, C.: Robust techniques for background substraction in urban traffic video. In: SPIE Electronic Imaging, San Jose, CA, vol. 5308, pp. 881–892 (2004)
8. Cozman, F., Krotkov, E.: Automatic mountain detection and pose estimation for teleoperation of lunar rovers. In: Proc. IEEE Int'l Conf. Robotics and Automation, Albuquerque, NM, pp. 2452–2457 (1997)
9. Cozman, F., Krotkov, E., Guestrin, C.E.: Outdoor visual position estimation for planetary rovers. Autonomous Robots **9**(2), 135–150 (2000)
10. Cucchiara, R., Grana, C., Piccardi, M., Prati, A.: Detecting moving objects, ghosts and shadows in video streams. IEEE Trans. Pattern Anal. Mach. Intell. **26**(10), 1337–1342 (2003)
11. Ettinger, S.M., Nechyba, M.C., Ifju, P.G., Waszak, M.: Vision-guided flight stability and control for micro air vehicles. Adv. Robotics **17**(7), 617–640 (2003)
12. Ferri, F., Smith, P.H., Lemmon, M., Renno, N.O.: Dust devils as observed by mars pathfinder. J. Geophys. Res. **108**(E12) (2003)
13. Gilmore, M., Castano, R., Mann, T., Anderson, R.C., Mjolsness, E., Manduchi, R., Saunders, R.S.: Strategies for autonomous rovers at mars. J. Geophys. Res. **105**(E12), 29,223–29,237 (2000)
14. Greeley, R., Kuzmin, R.O., Rafkin, S.C.R., Michaels, T., Haberle, R.M.: Wind-related features in Gusev crater, Mars. J. Geophys. Res. **108** (2003). doi:10.1029/2002JE002006
15. Greeley, R., Whelley, P., Arvidson, R., Cabrol, N., Foley, D., Franklin, B., Geissler, P., Golombek, M., Kuzmin, R., Landis, G., Lemmon, M., Neakrase, L., Squyres, S., Thompson, S.: Active dust devils in Gusev crater, Mars: Observations from the Mars Exploration Rover, Spirit. J. Geophys. Res. (in press)
16. Greeley, R., Whelley, P., Neakrase, L.: Martian dust devils: Directions of movement inferred from they tracks. Geophys. Res. Letters **31** (2004)
17. Gulick, V., Morris, R., Ruzon, M., Roush, T.: Autonomous image analysis during the 1999 Marsokhod rover field test. J. Geophys. Res. Planets **106**(E4), 7745–7764 (2000)
18. Immerkær, J.: Fast noise variance estimation. Comput. Vision Image Understanding **64**(2), 300–302 (1996)
19. Kiely, A., Klimesh, M.: The ICER progressive wavelet image compressor. IPN Prog. Rep. **42**(155), 1–46 (2003)
20. Lemmon, M., Wolff, M., Smith, M., Clancy, R., Banfield, D., Landis, G., Ghosh, A., Smith, P., Spanovich, N., Whitney, B., Greeley, R., Thompson, S., Bell, J., Squyres, S.: Atmospheric imaging results from the Mars Exploration Rovers: spirit and opportunity. Science **306**(5702), 1753–1756 (2004)
21. Lo, B., Velastin, S.A.: Automatic congestion detection system for underground platforms. In: Proc. Int'l Symp. Intell. Multimedia, Video and Speech Processing, Hong Kong pp. 158–161 (2001)
22. Maki, J., III, J.B., Herkenhoff, K., Sqyres, S., Kiely, A., Klimseh, M., Schwochert, W., Litwin, T., Willson, R., Johnson, A., Maimone, M., Baumgartner, E., Collins, A., Wadsworth, M., Elliot, S., Dingizian, A., Brown, D., Hagerott, E., Scherr, L., Deen, R., Alexander, D., Lorre, J.: Mars exploration rover engineering cameras. J. Geophys. Res. **108**(E12) (2003)
23. Metzger, S.M., Carr, J.R., Johnson, J.R., Parker, T.J., Lemmon, M.T.: Dust devil vortices seen by the mars pathfinder camera. Geophys. Res. Lett. **26**(18), 2781–2784 (1999)
24. Roush, T., Shipman, M., Morris, R., Gazis, P., Pedersen, L.: Essential autonomous science inference on rovers (EASIR). In: IEEE Aerospace Conference, Big Sky, MT, vol. 2, pp. 790–800 (2004)
25. Stein, F., Medioni, G.: Map-based localization using the panoramic horizon. IEEE Trans. Robotics Automat. **11**(6), 892–896 (1995)

26. Talluri, R., Aggarwal, J.K.: Image/map correspondence for mobile robot self-location using computer graphics. IEEE Trans. Pattern Anal. Mach. Intell. **15**(6), 597–601 (1993)
27. Thompson, D., Niekum, S., Smith, T., Wettergreen, D.: Automatic detection and classification of geological features of interest. In: Proc. IEEE Aerospace Conference, Big Sky, Montana (2005)

Author biographies

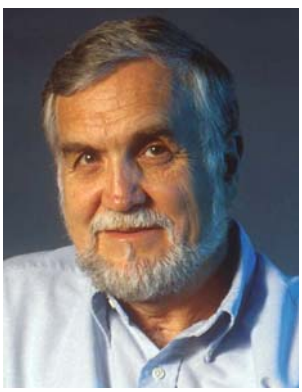
Andres Castano (BS '87, MEE '89, MS '92, Ph.D. '98) is a senior member of technical staff of the Machine Vision Group at NASA's jet propulsion laboratory (JPL). His work at JPL has focused on detection and recognition. He has participated in various target detection projects for science acquisition and surveillance for both NASA and military sponsors. Before joining JPL, he worked at the Information Sciences Institute at the University of Southern California (USC/ISI) where he was the lead roboticist for the CONRO project on reconfigurable robots.

Alex Fukunaga is currently an assistant professor in the Global Edge Institute at the Tokyo Institute of Technology. He was previously a researcher in the Artificial Intelligence Group at the NASA/Caltech Jet Propulsion Laboratory. He received a Ph.D. in Computer Science from the University of California, Los Angeles, and an A.B. in Computer Science from Harvard University. His research interests are in combinatorial optimization, heuristic search, automated planning and scheduling, and autonomous agents.

Jeffrey Biesiadecki has been a software engineer at NASA's Jet Propulsion Laboratory since 1993, after completing his Master's degree in Computer Science at the University of Illinois, Urbana-Champaign. He designed and implemented the core motor control and non-autonomous mobility flight software for the Mars Exploration Rovers, and is also one of the rover drivers for the Mars Exploration Rover "Opportunity", responsible for command sequences that tell the rover where to drive and how to operate its robotic arm on the surface of Mars.

Lynn Neakrase is a Ph.D. candidate at Arizona State University whose research interests include Martian cryogeology, and aeolian processes with a focus on atmosphere-surface interactions in laboratory simulations and analytical modeling. He is currently investigating dust devil mechanics including sediment flux and the effects of surface roughness on dust devils in the laboratory. He is a collaborator on the Athena Science Team for the Mars Exploration Rovers working on dust devil cataloging and surface roughness characterization.

Patrick Whelley received his B.S. in Geological Sciences from Arizona State University, and has remained there to complete his Masters work. He has been involved with Dr. Ronald Greeley and the Planetary Geology Group since 1999, via the education outreach program, Earthwatch. His current research consists of an extensive survey of MOC images investigating dust devils and examining the relationship between the frequency of dust devil tracks to Martian geomorphology and environmental constraints. He is a collaborator on the Athena Science Team that currently operates Spirit and Opportunity on the surface of Mars.



Ron Greeley is Regents' Professor of Geology, and has been involved in lunar and planetary exploration since the mid-1960s. His current interests are in deciphering the complex evolution of planetary surfaces through the study of landforms and the processes responsible for their formation. His focus includes the interaction of the atmosphere with the surface geology through the movement of windblown sand and dust on Earth, Mars, and Venus, using a combination of laboratory experiments, fieldwork on terrestrial analogs, and analysis of spacecraft data. He is currently leading the study of dust devils seen from the Mars Explorations Rover, Spirit.

Mark Lemmon graduated from The University of Washington with a B.S. in Physics in 1989 and from the University of Arizona with a Ph.D. in Planetary Sciences in 1994. He was a postdoc on the Galileo Probe and Mars Pathfinder missions. He is a science team member for the Mars Exploration, Rovers and Mars Science Laboratory camera team, and is the imaging team lead for the Phoenix Mars Lander.

Rebecca Castano is the supervisor of the Machine Learning and Instrument Autonomy (MLIA) group at JPL. She received her Ph.D. in Electrical Engineering from the University of Illinois with her dissertation in the area of computer vision. She is the technology lead for science data processing for the Autonomous Sciencecraft Experiment on the New Millennium Program's ST6 project and the team lead of the Onboard Autonomous Science Investigation System (OASIS) project. Her research interests include machine learning, onboard science, computer vision, and pattern recognition.

Steve Chien is a principal scientist at the Jet Propulsion Laboratory, California Institute of Technology where he leads efforts in autonomous systems. He has contributed to or led efforts deploying autonomy flight software to the Deep Space One, Three Corner Sat, Earth Observing One, and Mars Exploration Rovers mission. He has been awarded NASA medals for this work in 1997, 2000, and 2006. He is a three-time honoree in the NASA Software of the Year competition, most recently in 2005.