Subkilometer Crater Discovery with Boosting and Transfer Learning

WEI DING, University of Massachusetts Boston
TOMASZ F. STEPINSKI, University of Cincinnati
YANG MU, University of Massachusetts Boston
LOURENCO BANDEIRA, Instituto Superior Tecnico
RICARDO RICARDO, University of Houston
YOUXI WU, ZHENYU LU, TIANYU CAO, and XINDONG WU, University of Vermont

Counting craters in remotely sensed images is the only tool that provides relative dating of remote planetary surfaces. Surveying craters requires counting a large amount of small subkilometer craters, which calls for highly efficient automatic crater detection. In this article, we present an integrated framework on autodetection of subkilometer craters with boosting and transfer learning. The framework contains three key components. First, we utilize mathematical morphology to efficiently identify *crater candidates*, the regions of an image that can potentially contain craters. Only those regions occupying relatively small portions of the original image are the subjects of further processing. Second, we extract and select image texture features, in combination with supervised boosting ensemble learning algorithms, to accurately classify crater candidates into craters and noncraters. Third, we integrate transfer learning into boosting, to enhance detection performance in the regions where surface morphology differs from what is characterized by the training set. Our framework is evaluated on a large test image of $37,500 \times 56,250\ m^2$ on Mars, which exhibits a heavily cratered Martian terrain characterized by nonuniform surface morphology. Empirical studies demonstrate that the proposed crater detection framework can achieve an F1 score above 0.85, a significant improvement over the other crater detection algorithms.

Categories and Subject Descriptors: I.5.2 [**Pattern Recognition**]: Design Methodology—*Classifier design and evaluation; feature evaluation and selection; pattern analysis*; I.5.4 [**Pattern Recognition**]: Applications

General Terms: Algorithms

Additional Key Words and Phrases: Classification, feature selection, transfer learning, spatial data mining, planetary and space science

A preliminary version of the article [Ding et al. 2010], without the new algorithms of TL-Max, TL-Min, and TL-MaxMin on sampling test instances and the corresponding new experimental results of those three algorithms, was published in the *Proceedings of the* 19th *ACM International Conference on Information and Knowledge Management (CIKM'10)*.

This work is partially supported by NASA grant NNX09AK86G.

Authors' addresses: W. Ding, Computer Science Department, University of Massachusetts Boston, 100 Morrissey Blvd., Boston, MA 02125-3393; email: ding@cs.umb.edu; T. F. Stepinski, University of Cincinnati, 2600 Clifton Ave., Cincinnati, OH 45221; email: stepintz@uc.edu; Y. Mu, Computer Science Department, University of Massachusetts Boston, 100 Morrissey Blvd., Boston, MA 02125-3393; email: yangmu@cs.umb.edu; L. Bandeira, Instituto Superior Tecnico, Av Rovisco Pais, 1, 1049-001 Lisboa, Portugal; email: lpcbandeira@ist.utl.pt; R. Ricardo, University of Houston, 4800 Calhoun Road, Houston, TX 77004; email: vilalta@cs.uh.edu; Y. Wu, Z. Lu, T. Cao, and X. Wu, Department of Computer Science, University of Vermont, Burlington, VT 05405; email: {youxiwu, zlu, tcao, xwu}@cems.uvm.edu.

DOI 10.1145/1989734.1989743 http://doi.acm.org/10.1145/1989734.1989743

ACM Reference Format:
Ding, W., Stepinski, T. F., Mu, Y., Bandeira, L., Ricardo, R., Wu, Y., Lu, Z., Cao, T., and Wu, X. 2011. Subkilometer crater discovery with boosting and transfer learning. ACM Trans. Intell. Syst. Technol. 2, 4, Article 39 (July 2011), 22 pages.
DOI = 10.1145/1989734.1989743 http://doi.acm.org/10.1145/1989734.1989743

1. INTRODUCTION

Impact craters, the structures formed by collisions of meteoroids with planetary surfaces, are among the most studied geomorphic features in the solar system because they yield information about the past and present geological processes and provide the only tool for measuring relative ages of observed geologic formations [Crater Analysis Techniques Working Group 1979; Tanaka 1986]. However, advances in surveying craters present in images gathered by planetary probes have not kept up with the advances in collection of images at ever-higher spatial resolutions. Today, as in the past, efficient crater detection in planetary images remains as a daunting task due to the following challenges [Kim et al. 2005].

(1) *Challenge 1: Lack of distinguishing features.* Craters, as a landform formation, lack strong common features distinguishing them from other landform formations. Their sizes differ by orders of magnitude. Their rims have often been eroded since their formation millions of years ago, resulting in shapes that depart significantly from circles. They frequently overlap, complicating the task of their separation from background.
(2) *Challenge 2: Heterogeneous morphology in images.* Planetary surfaces are not homogeneous where nonuniform surface morphology frequently exhibits. Furthermore, planetary images may be taken at different lighting conditions, at different resolutions, and their quality varies so that even morphologically identical craters may have different appearances in different images.
(3) *Challenge 3: Huge amount of subkilometer craters in high-resolution planetary images.* The size distribution of craters follows power-law [Tanaka 1986]; large craters that can be easily identified manually are rare and small subkilometer craters are abundant.

As a result, comprehensive catalogs of craters are restricted to only large craters using manual inspection of images, for example, 42, 283 Martian craters with diameters larger than 5 *km* [Barlow 1988], and 8, 497 named lunar craters with diameters larger than a few kilometers [Andersson and Whitaker 1982]. There are millions of smaller craters waiting to be identified in a deluge of high-resolution planetary images but no means for their efficient identification and comprehensive analysis. If left to manual surveys, the fraction of cataloged craters to the craters actually present in the available and forthcoming imagery data will continue to drop precipitously. Crater autodetection techniques are needed, especially to catalog small subkilometer craters that are most abundant. Surveying such craters is ill-suited for visual detection, due to their shear numbers, but well-suited for an automated technique. In summary, automating the process of small crater detection is the only practical solution to a comprehensive surveying of such craters.

This article partially addresses Challenges 1–3 by designing an innovative framework that uses feature extraction, feature selection, and supervised boosting ensemble learning. The three key components of proposed method are as follows.

—*Utilizing mathematical morphology on shape detection for efficient identification of regions indicative for craters.* Due to the shear number of small sub-km craters discussed in Challenge 3, a practical crater detection tool must use computational time wisely. We adapt the concept of *crater candidates* introduced by Urbach and Stepinski [2009]. Crater candidates are the regions of an image that can potentially contain craters. The benefits of identifying crater candidates at an early stage are two-fold: (i) Significant computational time is reduced at later stages of complicated calculations on feature extraction and classification, where crater candidates are used instead of pixel-based image blocks that are calculated from exhaustive search of the entire image. (ii) The number of false positive detections is reduced at the stage of classification, because a large portion of the image, including background, is removed from being classified.

—*Using a combination of image texture features and a family of supervised boosting ensemble learning algorithms to yield a highly accurate classifier.* Targeting at Challenge 1, we are the first research team that contruct image gradient texture features from crater candidates for rapid feature extraction. Those gradient texture features can efficiently capture the underlying image gradient structure without requiring prior domain knowledge. A set of base learners are built from those texture features and combined to build a strong classifier using boosting ensemble learning.

—*Applying transfer learning to feature selection and classifier induction, in order to minimize training for the application of a crater detection tool to a heterogeneous planetary surface.* As discussed in Challenge 2, an unseen test site may contain craters that are different from those in the training site. A set of transfer learning algorithms are newly designed to transfer knowledge from an old training site to a new unseen test site. We propose TL-Random, TL-Max, TL-Min, and TL-MaxMin algorithms to sample new test instances and add them into the existing training set, using random sampling, sampling of maximum, minimum, and combined maximum and minimum distributions, respectively.

The entire framework is evaluated on a large, high-resolution image of Martian surface ($37{,}500 \times 56{,}250\ m^2$), featuring high density of small sub-km craters and spatial variability of crater morphology. The proposed boosting ensemble learning algorithms with transfer learning achieve an F1 score above 0.85 on crater detection, a significant improvement over the other crater detection algorithms. The transfer learning algorithms have proved powerful on regions where surface morphology differs as characterized by the training set. The experimental results demonstrate robustness and good accuracy that validate our approach and make it feasible to construct a robust and reliable crater autodetection framework that can be widely adopted for planetary research.

The rest of the article is organized as follows. Section 2 discusses the proposed crater detection framework: Sections 2.1 and 2.2 explain how to construct crater candidates and image texture features from those candidates. Section 2.3 provides a brief review on unsupervised versus supervised crater detection methods. Section 3 introduces our ensemble boosting algorithms used for crater detection with and without using transfer learning. Section 4 presents our empirical study on finding craters in a large, high-resolution planetary image. Section 5 summarizes our work and discusses future directions.

2. A FRAMEWORK FOR AUTOMATIC CRATER DETECTION

The flow chart indicating components of our method is shown in Figure 1. A key insight behind our method is that a crater can be recognized as a pair of crescent-like highlight and shadow regions in an image (see Figure 2). Pairwise crescent-like shapes are identified from images using a shape detection method based on mathematical morphology [Urbach et al. 2007], and those that can be matched are used to construct crater candidates [Urbach and Stepinski 2009], the locations where craters likely reside. The input objects of supervised learning are derived from image blocks containing

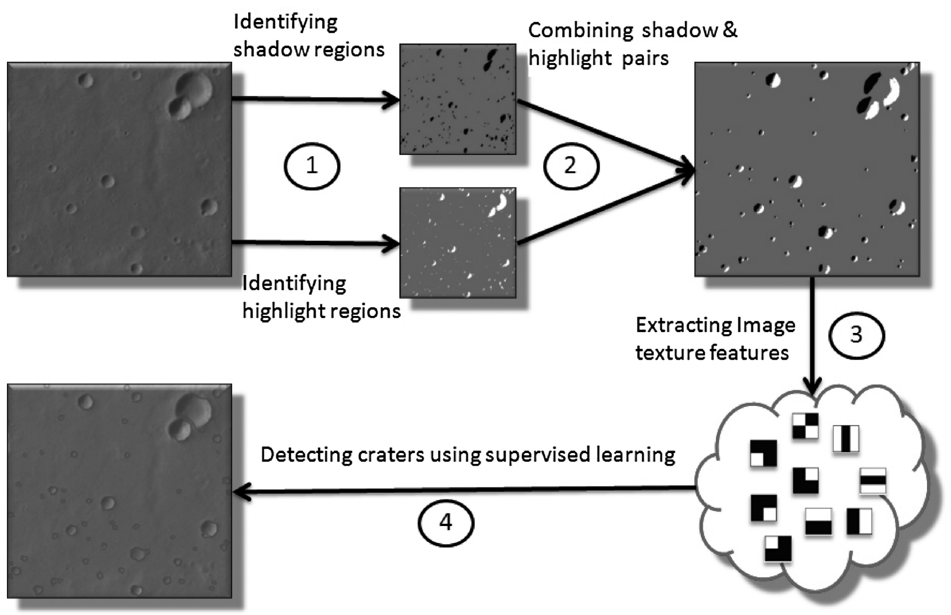

Identifying shadow regions
Combining shadow & highlight pairs
1
2
Identifying highlight regions
Extracting Image texture features
3
Detecting craters using supervised learning
4

Fig. 1. Diagram illustrating the crater detection framework. (1) Crescent-like shadow and highlight regions are identified using shape filters. (2) Shadow and highlight regions that can be matched are used to construct crater candidates. (3) Image texture features are extracted from crater candidates using square kernels. (4) Craters are identified using supervised learning algorithms.

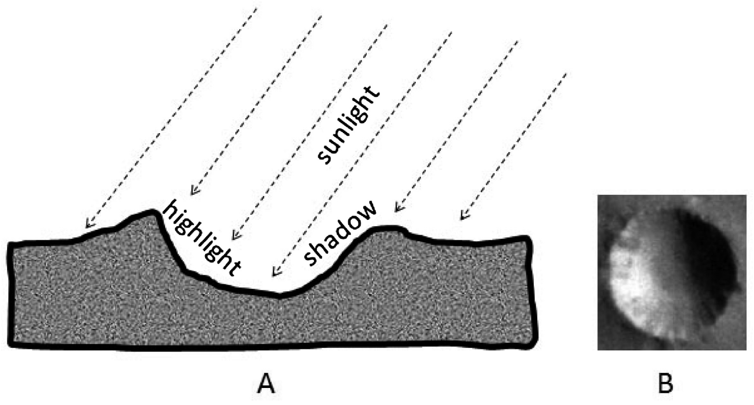


Fig. 2. (A) Diagram explaining why an image of a crater consists of crescent-like highlight and shadow regions. (B) An image of an actual 1 *km* crater showing the highlight and shadow regions.

crater candidates and the classification is performed on feature vectors based on image texture features.

2.1. Finding Regions That Are Indicative For Craters

In order to reduce the load on the classification module, we first identify crater candidates: parts of an image that contain crescent-like pairs of shadows and highlights. Identification of crater candidates is achieved using an image processing method based on mathematical morphology proposed by Urbach et al. on object detection in Urbach and Stepinski [2009] and Urbach et al. [2007]. Figure 3 shows a flow diagram of the

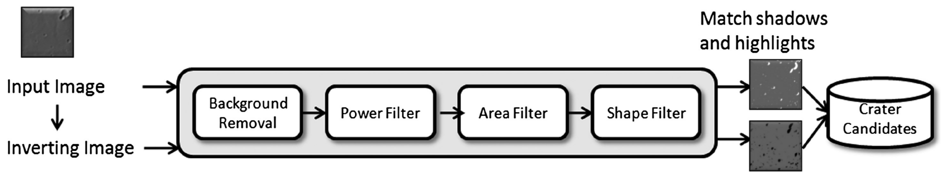


Fig. 3. Diagram illustrating individual steps in constructing crater candidates.

method used for identification of crater candidates. The highlight and shadow shapes are processed in parallel using inverted image to process the shadow shapes. The goal is to eliminate all the shapes that are not indicative of craters while keeping the highlight and shadow shapes. The step of Background Removal deletes shapes, such as mountains, that are too large to be part of the craters; the Power Filter removes shapes that lack sufficient contrast; the Area Filter removes shapes that are too small for reliable crater detection; the Shape Filter uses shape attributes that are invariant to translation, rotation, and scaling to preserve or remove regions of an image exclusively on the basis of their shapes. Utilization of the Shape Filter, that requires only a single parsing of an image, improves performance by a factor of 5 to 9 in comparison with other shape detection methods [Urbach et al. 2007]. In the final step, highlight and shadow regions are matched so that each pair corresponds to a single crater candidate. This method does not have high enough accuracy to constitute a stand-alone crater detection technique, but is ideal for identification of crater candidates. More calculations must be performed to discriminate craters from noncraters in those crater candidates. Compared to the shape features used in Urbach and Stepinski [2009] that results less satisifying results on crater detection (experimental results will be given in Section 4.6), in this article, we construct image texture features from the crater candidates to be used by the classification algorithms.

2.2. Image Texture Feature Construction

We use image texture features reminiscent of Haar basis functions which were first proposed in Papageorgiou et al. [1998] for detection of objects and later popularized by Viola and Jones [2004] in the context of face detection. These features can be thought of as image masks consisting of black and white sectors. Different from vertical and horizontal rectangle features used in face detection [Viola and Jones 2004], we specially design nine square mask-features shown in Figure 4. A symmetric square mask is used because a crater to be identified is in a symmetric shape. A mask in different scales is scanned through the region of a crater candidate. Each position of the mask produces a single feature value. The value of a feature is the difference between the sum of gray scale values in pixels located within the white sectors and the black sectors. The number of features is equal to the number of masks used multiplied by the number of positions overlaid by those masks. All features can be calculated very efficiently in one image scan, using an integral image data structure [Viola and Jones 2004].

To represent a crater candidate in terms of Haar-like features, we first extract square image blocks around each crater candidate. In our experiments, we use the size twice that of the candidate in order to include regions surrounding crater rims. The underlying texture information of each crater candidate is encoded in the set of nine mask-features in different scales, having various granularities and positioned at finely sampled locations. Thus an image containing a crater candidate and its immediate surroundings is described by thousands of texture features. Those features are not independent from each other and those over-complete features compensate the limited texture information a single square mask-feature can capture. Underlying gradient

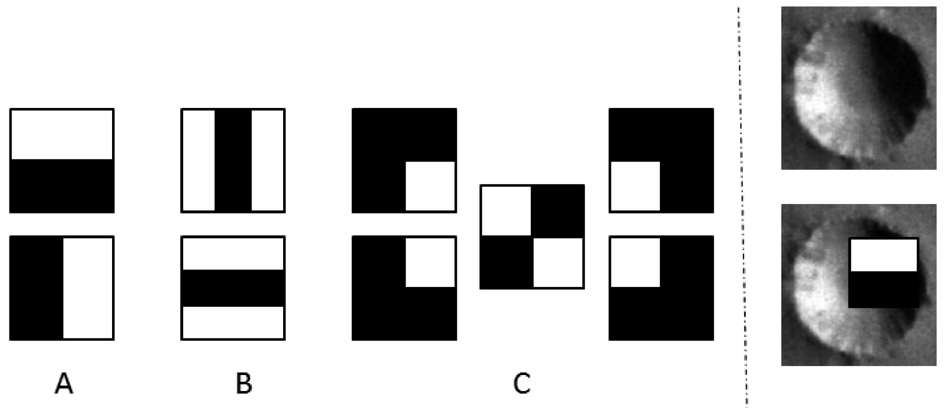


Fig. 4. 9 types of square masks: (A) 2 two-rectangle masks to capture horizontal and vertical gradient texture, (B) 2 three-rectangle masks to capture horizontal and vertical edge gradient texture, (C) 5 four-rectangle masks to capture diagonal gradient texture. Far-right: An example of a two-rectangle mask overlay on a crater. A crater is a depression in the surface and appears in the image as a pair of shadow and highlight semicircular shapes. The illumination is from north-east.

texture information is encoded by those features without the requirement of prior domain knowledge. If a single simple feature can be viewed as a weak learner, that is, only using this feature to classify crater candidates by constructing a single-node decision tree, it is a natural choice to build a strong ensemble classifier out of thousands of weak learners, using the boosting approach.

2.3. Unsupervised vs. Supervised Crater Detection

Salamuniccar and Loncaric [2007] provided an extensive review of all previous research on crater detection algorithms. Existing efforts on detecting craters in planetary images can be divided into two general categories: unsupervised approaches and supervised approaches.

The unsupervised methods identify crater rims in an image as circular or elliptical features [Leroy et al. 2001; Honda et al. 2002; Cheng et al. 2002; Bandeira et al. 2007; Kim et al. 2005]. In particular, the original image is preprocessed [Leroy et al. 2001; Bandeira et al. 2007; Kim et al. 2005] to enhance the edges of rims, and the actual detection is achieved by means of the Hough transform [Hough V 1962] or genetic algorithms [Honda et al. 2002]. Unsupervised methods have the advantage of being fully autonomous but the performance is usually at least one magnitude less accurate than supervised methods.

The supervised methods [Burl et al. 2001; Vinogradova et al. 2002; Wetzler et al. 2005] take advantage of domain knowledge in the form of labeled training sets that guide classification algorithms. In Burl et al. [2001] and Vinogradova et al. [2002], a continuously scalable template model technique was used to achieve detection. In Wetzler et al. [2005], a number of algorithms were tested and the Support Vector Machine algorithm was shown to achieve the best rate of crater detection. More recent methods [Kim et al. 2005; Martins et al. 2009] incorporated techniques originally developed [Viola and Jones 2004] for the purpose of face detection. These methods concentrated on the classification component of crater detection and did not incorporate identification of crater candidates or transfer learning, as what has been extensively studied in this article.

Notice that previous research on crater detection algorithms—supervised and unsupervised methods—focused predominantly on partially addressing Challenge 2,

Table I. Summary of the Three Learning Algorithms

Algorithm	Sampling in Test Set	Difference
Boost	No	Iterative Weight Updating
Naive	No	Greedy Weight Updating
TL	TL-MinMax TL-Min TL-Max TL-Random	Iterative Weight Updating in Same & Different Distributions

in which morphologically identical craters exhibiting different appearances in different images [Leroy et al. 2001; Honda et al. 2002; Cheng et al. 2002; Bandeira et al. 2007; Kim et al. 2005; Vinogradova et al. 2002; Wetzler et al. 2005; Burl et al. 2001; Martins et al. 2009]. In addition, the bulk of previous work relies on inefficient exhaustive search of the entire image using pixel-based approaches. This may work for finding a small number of large craters in low-resolution images, but not for finding a very large number of small craters in high resolution-images. Billions of pixels in a high-resolution planetary image inevitably become a bottleneck of scalability of those crater detection methods.

The problem of finding crater candidates has only recently been raised in Urbach and Stepinski [2009], but the relatively low crater detection rates using a decision tree J48 are reported due to the use of less discriminative geometric shape features. Urbach and Stepinski's method uses a small set of features (16 features used in their experiments) to describe the shapes of the shadow and high regions of crater candidates. However, other noncrater landforms in similar shapes makes using shape features an unideal choice on crater detection. It is well known that the classification performance is primarily controlled by the quality of features. In this article, we use a large set of texture features (1089 features used in our experiments) in combination of boosting ensemble learning algorithms to achieve better accuracy on crater detection. Detailed comparison will be presented in Section 4.6.

To the best of our knowledge, the problem of transfer learning in the context of autodetection of craters has not been previously addressed. This omission renders most existing approaches impractical for planetary research as the benefit of automation decreases significantly if new training sets need to be established for every new image or even for various segments of the same image. In the next section, we will design several supervised algorithms, some of which integrate transfer learning.

3. BOOSTING WITH AND WITHOUT TRANSFER LEARNING

To classify crater candidates into craters and noncraters on the basis of texture features, we have designed and implemented three supervised learning algorithms. These algorithms simultaneously select subset features necessarily for accurate classification and train the final ensemble classifier based on the supplied training set. The first is the Boost algorithm, a variant of the AdaBoost algorithm inspired by the methodology of face detection [Viola and Jones 2004]. The second is the Naive algorithm, a drastic simplification of the Boost algorithm using a greedy approach instead of the boosting method. The third is the TL algoritm, a transfer learning algorithm using four different sampling methods. Table I gives a brief summary of the three algorithms.

3.1. Boosting without Transfer Learning

A crater candidate at this stage is represented as a feature vector $\widehat{x} = \langle f_1, \ldots, f_N \rangle$. Each feature f_i, $i = 1 \ldots N$, is produced by a square mask-feature in a particular position overlaying the cater candidate.

ALGORITHM 1: Boost: A boosting algorithm for feature selection and classification

```
Require: .
   (1) Given crater candidates $(\widehat{x_1}, y_1), \ldots, (\widehat{x_n}, y_n)$ where $y_i = 0, 1, i = 1, \ldots, n$ for non-crater
   and crater examples respectively.
   (2) Initialize weights $w_i = \frac{1}{2m}$ if $y_i = 0$, $w_i = \frac{1}{2l}$ if $y_i = 1$, where m and l are the number of
   non-crater and crater examples respectively.
1: for $t = 1 \ldots T$ do
2:     Normalize the weight, $w_{t,i} = \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}, i = 1, \ldots, n$
       so that $w_t$ is a probability distribution.
3:     Select the best weak classifier with respect to the weighted error
       $\epsilon_t = argmin_{f,p,\theta} \sum_i w_i |h(\widehat{x_i}, f, p, \theta) - y_i|$,
       For each feature, $f$, train a classifier $h$, which is restricted to using a single feature.
4:     Define $h_t(\widehat{x}) = h(\widehat{x}, f_t, p_t, \theta_t)$, where $\widehat{x}, f_t, p_t, \theta_t$ are the minimizers of $\epsilon_t$.
5:     Update the weights:
       $w_{t+1,i} = w_{t,i}\beta_t^{1-e_i}, i = 1, \ldots, n$
       where $e_i = 0$ if a crater candidate $x_i$ is classified correctly, $e_i = 1$ otherwise, and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.
6: end for
7: The final strong classifier is:
   $h(\widehat{x}) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(\widehat{x}) \geq \mu \sum_{t=1}^{T} \alpha_t \\ 0 & otherwise \end{cases}$
   where $\alpha_t = ln\frac{1}{\beta_t}$ and $\mu$ is a user-defined threshold.
```

The Boost algorithm (see Algorithm 1) generates a sequence of weak classifiers $h_t(f)$ and combines them through a weighted boosting approach to build a strong ensemble classifier $H(\widehat{x})$:

$$H(\widehat{x}) = \sum_{t=1}^{T} \alpha_t h_t(f), \tag{1}$$

where T is the number of iterations, $t = 1, \ldots, T$; f, $f \in \{f_1, \ldots, f_N\}$, is the single feature selected at each boosting iteration to construct a weak classifier $h_t(f)$, and α_t is the learned weight of hypothesis $h_t(f)$ when adding the newly selected weak classifier into the ensemble. The Boost algorithm (Algorithm 1) iteratively selects one feature at a time and stops when reaching T iterations; note that $T \ll N$. Different from the traditional AdaBoost algorithm that usually uses the entire feature set, Boost at each iteration selects only one best feature at one time. Thus feature selection is integrated into the boosting iteration. Three core steps are required to complete one boosting iteration.

(1) *Weak Classifier Learning*. The construction of a weak classifier $h_t(f)$ on a single feature f at iteration t is straightforward. Given n crater candidates, $(\widehat{x_1}, y_1), \ldots, (\widehat{x_n}, y_n)$ where class label $y_i = 0, 1$ $(i = 1, \ldots, n)$ is for noncrater and crater examples respectively, a weak classifier $h_t(f)$, consists of a feature f, a threshold θ, and a polarity p indicating the direction of the inequality.

$$h_t(f) = \begin{cases} 1 & if\ f(\widehat{x}) < p\theta \\ 0 & otherwise \end{cases} \tag{2}$$

A weak leaner $h_t(f)$ can be essentially viewed as a decision stump, a single-node decision tree. Exhaustive search is conducted in order to find best values of p and θ. For a feature f, $f \in \{f_1, \ldots, f_N\}$, $2 \times n$ single-node decision tress are built, where 2 represents positive and negative signs of p, and the value of every candidate $\widehat{x_i}$, $i = 1 \ldots n$, on feature f is used as possible values for θ.

(2) *Feature Selection*. Calculate the weighted error sum of each weak classifier and select the best learner (a.k.a. the best feature) that produces the minimum error.
(3) *Weight Updating*. Update weights using the same method proposed in AdaBoost [Freund and Schapire 1995]: increase the weights of incorrectly classified examples and decrease the weights of correctly classified examples. The incorrectly classified examples will have more chances of being chosen in the next iteration when calculating the weighted error sum in step 2. Hence, the next selected feature concentrates more on the mistakes made by the earlier features. The key advantage of the Boost algorithm is that the weights encode the classification results of the previous features and this information is used to select the next best feature.

ALGORITHM 2: Naive: A naive greedy algorithm for feature selection and classification

```
Require: .
   (1) Given crater candidates $(\widehat{x_1}, y_1), \ldots, (\widehat{x_n}, y_n)$ where $y_i = 0, 1, i = 1, \ldots, n$ for non-crater
   and crater examples respectively.
   (2) Initialize weights $w_i = \frac{1}{2m}$ if $y_i = 0$, $w_i = \frac{1}{2l}$ if $y_i = 1$, where m and l are the number of
   non-crater and crater examples respectively.
1: Normalize the weight, $w_{t,i} = \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}, i = 1, \ldots, n$
   so that $w_t$ is a probability distribution.
2: Select the best t $(t = 1, \ldots, T)$ weak classifiers with respect to the weighted error
   $\epsilon_t = \sum_i w_i |h(\widehat{x_i}, f, p, \theta) - y_i|$,
   For each feature, $f$, train a classifier $h$, which is restricted to using a single feature.
3: Define $h_t(\widehat{x}) = h(\widehat{x}, f_t, p_t, \theta_t)$ where $\widehat{x}, f_t, p_t, \theta_t$ are the minimizers of $\epsilon_t$, and $t = 1, \ldots, T$.
4: $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.
5: The final strong classifier is:
   $h(\widehat{x}) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(\widehat{x}) \geq \mu \sum_{t=1}^{T} \alpha_t \\ 0 & otherwise \end{cases}$
   where $\alpha_t = ln \frac{1}{\beta_t}$ and $\mu$ is a user-defined threshold.
```

As depicted in Algorithm 1, steps 2–4 are used for Weak Classifier Learning and Feature Selection, and step 5 is for Weight Update. The number of craters is usually less than the number of noncraters. The initial weight of each training instances is designed to cope with imbalance data by using different group average weights in the positive and negative classes, respectively. The weights of positive examples are not necessarily the same as those of negative examples, whereas every positive example (a true crater) in the training set has the same weight and every negative example (a noncrater) shares the same weight.

In order to reduce the computational cost of the Boost algorithm, we design a simplified greedy version of the algorithm and call it the Naive algorithm (see Algorithm 2). The Naive classifier uses the same Weak Classifier Learning step and selects the top T best features using the weighted error sum in the step of Feature Selection as a criterion without any further iterations on the step of Weight Updating.

Time Complexity Analysis. The time complexity of the Boost algorithm is $O(TNn)$, where n is the number of training examples, N is the number of total features, and T is the number of boosting iterations. In particular, each feature produces n weak classifiers, based on each feature value for every training example according to the threshold θ; N features produce Nn classifiers; it takes $O(Nn)$ time to find the weak classifier that produces the minimum error; and it takes $O(TNn)$ time to select the top T features after T boosting iterations.

The time complexity of the Naive algorithm is $O(Nn)$ as no boosting iterations are performed. Interestingly, the Naive classifier performs decently well in some circumstances during our real-world case study (see Section 4).

3.2. Boosting with Transfer Learning

Boost and Naive assume that both training and testing instances are drawn independently and identically from the same underlying distribution. What if training and test instances are from different distributions? We have designed a transfer learning based algorithm, inspired by the TrAdaBoost algorithm [Dai et al. 2007], which is capable of transferring knowledge from the old training data to the new test data. We refer to it as the TL algorithm. In principle, transfer learning algorithms are often used when the training set and test set are not in the same feature space or have the same distribution [Pan and Yang 2010]. The TL algorithm (Algorithm 3) has the same three steps as the Boost algorithm, but the Weight Updating step is different as it attempts to transfer knowledge from the original training set to the new test data. As the Boost algorithm is not expected to perform well if the test data has a different distribution from the training data, because the critical set of features that best serves to distinguish craters in the training set may not be the same as that in the test set.

ALGORITHM 3: TL: A boosting algorithm using transfer learning for feature selection and classification

```
Require: .
    (1) Given a training set that includes crater candidates
    $(\widehat{x_1}, y_1), \ldots, (\widehat{x_{n_d}}, y_{n_d}), (\widehat{x_{n_d+1}}, y_{n_d+1}), \ldots, (\widehat{x_{n_d+n_s}}, y_{n_d+n_s})$, where
    $y_i = 0, 1, i = 1, \ldots, n_d, n_d + 1, \ldots, n_d + n_s$ for non-crater and crater examples respectively.
    This training set has $n_d$ diff-distribution examples $(1, \ldots, n_d)$ and $n_s$ same-distribution
    examples $(n_d + 1, \ldots, n_d + n_s)$, and $n = n_d + n_s$.
    (2) Initialize weights $w_i = \frac{1}{2m}$ if $y_i = 0$, $w_i = \frac{1}{2l}$ if $y_i = 1$, where m and l are the number of
    non-crater and crater examples respectively.
1: for $t = 1 \ldots T$ do
2:     Normalize the weight, $w_{t,i} = \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}, i = 1, \ldots, n$
       so that $w_t$ is a probability distribution.
3:     Select the best weak classifier with respect to the weighted error
       $\epsilon_t = argmin_{f,p,\theta} \sum_i w_i |h(\widehat{x_i}, f, p, \theta) - y_i|, i = n_d + 1, \ldots, n_d + n_s$
       For each feature, $f$, train a classifier $h$ in same-distribution data, which is restricted to
       using a single feature.
4:     Define $h_t(\widehat{x}) = h(\widehat{x}, f_t, p_t, \theta_t)$ where $\widehat{x}, f_t, p_t, \theta_t$ are the minimizers of $\epsilon_t$.
5:     Update the weights:
       $w_{t+1,i} = w_{t,i}\beta_t^{-e_i}$, if $n_d + 1 \leq i \leq n_d + n_s$ (increase the weights for the same-distribution)
       $w_{t+1,i} = w_{t,i}\beta^{e_i}$, if $1 \leq i \leq n_d$ (decrease the weights for the diff-distribution)
       where $e_i = 0$ if example $x_i$ is classified correctly, $e_i = 1$ otherwise, and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$,
       $\beta = \frac{1}{1+\sqrt{2ln\frac{n}{T}}}$
6: end for
7: The final strong classifier is:
    $h(\widehat{x}) = \begin{cases} 1 & \sum_{t=\lceil \frac{T}{2} \rceil}^{T} \alpha_t h_t(\widehat{x}) \geq \mu \sum_{t=\lceil \frac{T}{2} \rceil}^{T} \alpha_t \\ 0 & otherwise \end{cases}$ where $\alpha_t = ln\frac{1}{\beta_t}$ and $\mu$ is a user-defined threshold.
```

We denote the previous original training data as the diff-distribution training data; and here we are uncertain about the similarity and usefulness of this data for the new task. We denote the additional small portion of labeled test data, which is a representative of the new set of crater candidates, as the same-distribution training data.

During the training process, we apply the Boost algorithm to the same-distribution training data to build a model; the weights of misclassified examples are increased during the next iteration while the weights of correctly classified examples are decreased. The key component is that we transfer knowledge from the old training data to the new test data by modifying the weights of misclassified examples from the diff-distribution training data. Those misclassified examples are considered as the ones that are dissimilar to the same-distribution examples and should be deemphasized. Accordingly, we decrease (not increase) the weights of those examples in order to weaken their impact. The weight-changing mechanism selects good examples (similar to the labeled test data) from the old training data to compensate the insufficient training examples in the same-distribution data. The change of weight factor $\beta = \frac{1}{1+\sqrt{2ln\frac{n}{T}}}$ for misclassified examples from diff-distribution and the threshold voting $\sum_{t=\lceil\frac{T}{2}\rceil}^{T} \alpha_t h_t(\widehat{x}) \geq \mu \sum_{t=\lceil\frac{T}{2}\rceil}^{T} \alpha_t$ in the final strong classifier are to assure that the average training loss on the diff-distribution converges to zero [Dai et al. 2007; Freund and Schapire 1995].

ALGORITHM 4: Sampling methods to construct the same-distribution set from a test set

```
Require: .
    (1) Given a training set as diff-distribution set which includes examples $(\widehat{x_1}, y_1), \ldots, (\widehat{x_{n_d}}, y_{n_d})$,
    where $y_i = 0, 1, i = 1, \ldots, n_d$ for non-crater and crater examples respectively and a test set
    which include examples $(\widehat{x_1}, y_1), \ldots, (\widehat{x_m}, y_m)$, where $y_i, i = 1, \ldots, m$ are unknown.
    (2) The number $n_s$ indicates how many examples in a test set will be regarded as
    same-distribution set
    (3) Input parameter $K$ for the # of nearest neighbors when sampling a test instance.
 1: Quantize the input space range into bins and re-represent the training samples and test
    samples in a probability mode.
 2: Caculate the Kullback-Leibler divergence between the test set and the training set. A
    distance matrix is $\mathbf{D} \in \mathbb{R}^{m \times n_d}$. Each row in matrix $\mathbf{D}$ corresponds to the distances between
    one testing example to $n_d$ training examples.
 3: Construct the Min-distribution divergence vector $D_{min}$:
    for each sample $\widehat{x_i}$, $i = 1, \ldots, m$, in the test set, $K$ nearest neighbors in training samples can
    be found according to the distance matrix $\mathbf{D}$, then $D_{min}(i) = \frac{1}{K}\sum_{j=1}^{K} \mathbf{D}(i, j)$.
 4: Construct the Max-distribution divergence vector $D_{max}$:
    for each sample $\widehat{x_i}$, $i = 1, \ldots, m$, in the test set, $K$ training samples with farthest distances
    can be found according to the distance matrix $\mathbf{D}$, then $D_{max}(i) = \frac{1}{K}\sum_{j=1}^{K} \mathbf{D}(i, j)$.
 5: Construct the filtered same-distribution set:
    TL-Min: The same-distribution set $S_{min}$ under the min filter is composed of $n_s$ examples in
    the test set which have the smallest $D_{min}$.
    TL-Max: The same-distribution set $S_{max}$ under the max filter is composed of $n_s$ examples in
    the test set which have the largest $D_{max}$.
    TL-MinMax: The same-distribution set $S_{minmax}$ under the min-max filter is composed of
    $\lfloor\frac{n_s}{2}\rfloor$ examples which have the smallest $D_{min}$ and $\lceil\frac{n_s}{2}\rceil$ examples which have the largest $D_{max}$
    in the test set.
```

There are two major differences between the TL algorithm and the existing algorithm TrAdaBoost [Dai et al. 2007].

(1) *Feature Selection*. We use an embedded approach in feature selection (steps 3–4 in Figure 3). In our method, we select the best feature in each iteration while constructing a strong classifier sequentially. The key contribution of the algorithm is that some features contribute more in the new test data and should be transferred and emphasized, while some features provide less or no contributions at all and thus should be deemphasized. The subset features that best discriminate craters

and noncraters in the old training set are not necessarily the same subset features in a new unseen test set.

(2) *Sampling method from the test set*. TrAdaBoost uses random sampling to choose new test instances and adds them into the training set. In addition to the random sampling, which we denoted as TL-Random in this article, we introduce three new methods TL-Max, TL-Min, and TL-MaxMin (see Algorithm 4) to construct the same-distribution set in order to transfer knowledge more efficiently.

For the TL algorithm, we extracted some samples from a test set to compose the same-distribution set. When a training set and a test set are in different distributions, the quantity of the diff-distribution set, a.k.a. the original training set, is inadequate to train a transferable classifier. After combining the diff-distribution set with the same-distribution set, the quality of newly selected samples may have a great influence on classifier training. Apparently, randomly selecting samples to construct the same-distribution set cannot guarantee the quality of the same-distribution. We take into consideration of the distribution divergence when constructing the same-distribution set. Normally, the samples that distribute significantly differently with the training samples should have more contribution for classifier induction. However, the samples which are greatly different from the main trend distribution could be outliers thus lead to wrong training results. Furthermore, the test samples which have very similar distribution with the training samples may also be useful, as those samples may not be in the same class with those in the training set. For example, the sample in the test set is a crater but the samples in the original training set which share a similar distribution may not be craters. Therefore, the testing samples in large and small distribution differences to the training samples have their own benefits and deficiencies.

In order to make the training process geared to the new knowledge gained in the same-distribution, we propose to use three new methods, TL-Min, TL-Max, and TL-MinMax, to build a same-distribution set, considering the closest distribution, farthest distribution, and combined cases, respectively. The detailed construction method is in Algorithm 4. To calculate the divergence of the samples, we firstly quantize all the training samples and testing samples with a certain bin number and rerepresent all the samples by a probability distribution (step 1). The quantization range is determined by the minimum and maximum value of input samples. Kullback-Leibler(KL) divergence [Kullback and Leibler 1951][1] is applied for the probability distribution divergence calculation (step 2). We use Min(Max)-distribution divergence vectors to find the test instances closest (farthest) to the instances in the diff-distribution set (steps 3–4). A TL-Min filter is constructed to select same-distribution samples with the minimum distribution difference, a TL-Max filter for the maximum distribution difference, and a TL-MinMax fileter for the combination of these two filters to form a same-distribution set. After the same-distribution set is constructed, feature selection and classifier induction are conducted using the TL algorithm described in Algorithm 3.

Time Complexity Analysis. The time complexity of the TL algorithm is the combination of the same-distribution construction and the boosting calculation. Same as the Boost algorithm, the boosting step takes $O(TNn)$, where n is the number of training examples, N is the number of total features, and T is the number of boosting iterations. It takes $O(mn)$ on the construction of the same-distribution, where m is the number of testing

[1]In principle, given two discrete random variables P and Q, KL-divergence calculates information gain achieved if P can be used instead of Q. It is also called the relative entropy, for using Q instead of P. It is essentially the difference between two probability distributions P and Q.

	# of candidates	Precision			Recall			F1		
		Boost	Naïve	TL	Boost	Naïve	TL	Boost	Naïve	TL
West Region	6708	0.897	0.889	**0.911**	0.791	**0.821**	0.779	0.841	**0.853**	0.840
Central Region	2935	0.805	0.772	**0.902**	0.783	**0.816**	0.769	0.794	0.794	**0.830**
East Region	3432	0.915	0.867	**0.954**	0.809	**0.855**	0.843	0.892	0.861	**0.895**
All Regions	13075	0.881	0.856	**0.919**	0.807	**0.827**	0.791	0.842	0.841	**0.851**

Fig. 5. Performance results of the Boost, Naive, and TL algorithms; parameter values: Boost–150 features selected and $\mu = 0.525$; Naive–150 features selected and $\mu = 0.675$; TL–150 features selected, $\mu = 0.500$, TL-Random select 253 new test instances into the same-distribution set.

examples, because each test instance needs compare to each training instance. Thus the TL algorithm has a time complexity of $O(TNn + mn)$.

4. EXPERIMENTAL RESULTS

4.1. Test Image

We have selected a portion of the High-Resolution Stereo Camera (HRSC) nadir panchromatic image h0905 [HRSC Data Browser 2009], taken by the Mars Express spacecraft, to serve as the test set. As illustrated in Figure 15, the selected image has a resolution of 12.5 meters/pixel and a size of 3,000 by 4,500 pixels ($37{,}500 \times 56{,}250\ m^2$). A domain expert manually marked $\sim$3,500 craters in this image to be used as the ground truth to which the results of autodetection are compared. The image represents a significant challenge to automatic crater detection algorithms because it covers a terrain that has spatially variable morphology and because its contrast is rather poor (which is most noticeable when the image is inspected at a small spatial scale). We divide the image into three sections denoted as the west region, the central region, and the east region (see Figure 15). The central region is characterized by surface morphology that is distinct from the rest of the image. The west and east regions have similar morphology but the west region is much more heavily cratered than the east region.

4.2. Training Set Construction

In the first stage of our method, we identify 13,075 crater candidates in the image using the pipeline depicted in Figure 3. The dataset is imbalanced as the majority objects are noncrater candidates. 1,089 image texture features are constructed using the 9 square-mask features described in Figure 4. The training set for the Boost and Naive algorithms consists of 204 true craters and 292 noncrater examples selected randomly from amongst crater candidates located in the northern half of the east region. Thus, the training set uses only 3.75% of the total dataset. Note that we have purposely restricted the locations of examples in a training set to a specific sector of the image in order to mimic actual planetary research; it is likely that in current studies such craters are identified in a specific region and are in need of identification by a supervised learning algorithm in the rest of the image. For the TL algorithm results shown in Figure 5, we have constructed an additional training set (same-distribution set), using random sampling(TL-Random), consisting of 253 crater candidates (102 true craters and 153 noncraters) selected from random locations throughout the entire image. The ratio between the false and true examples in the same-distribution data is proportional to that in the diff-distribution data ($\frac{153}{102} >= \frac{292}{204}$). The original training set consisting of 496 examples from the northeastern section of the image serves as the diff-distribution set.

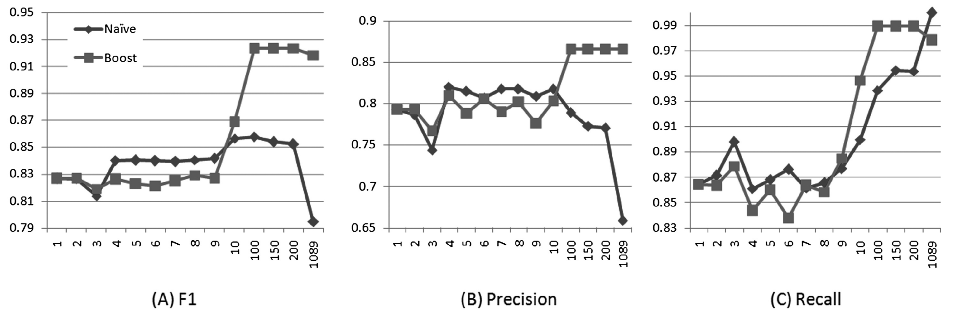


Fig. 6. Boost versus Naive. x-axis: number of features selected; y-axis: performance scores.

4.3. Comparative Performance of Boost, Naive, and TL

The table in Figure 5 summarizes the performance results of crater detection by the three algorithms: Boost, Naive, and TL. The ground truth of the entire image serves as an external criterion to evaluate the performance of the three algorithms on the unseen test set. Of the three algorithms, the number of features used to construct a strong classifier and the values of the threshold μ are selected to maximize the performance of each classifier.

The candidate data has an imbalanced class distribution and the successful detection of true craters is more significant than the detection of noncraters. Hence we use recall ($r = \frac{TP}{TP+FN}$) and precision ($p = \frac{TP}{TP+FP}$) and F1 as the evaluation metrics, where TP stands for the number of true positive detections (detected craters that are actual craters), FP stands for the number of false positive detections (detected craters that are actually not), and FN stands for the number of false negative "detections" (nondetection of real craters). F1 measures the harmonic mean between precision and recall $\frac{2}{\frac{1}{r}+\frac{1}{p}}$. The values of precision, recall, and F1 are listed, and the best performance of each measure is highlighted in bold. A precision score of 1.0 means that every object classified as a crater is indeed a crater but says nothing about the number of craters that are not recognized by classifiers as such. A recall score of 1.0 means that every true crater is classified as such but says nothing about how many other landforms were incorrectly classified as craters. An F1 score of 1.0 means that all the existing craters are correctly identified and all the objects classified as craters are true craters.

Of the three algorithms compared, the TL classifier using random sampling (TL-Random) yields the best precision in all regions and the Naive classifier yields the worst precision in all regions. On the other hand, the Naive classifier has the highest recall in all regions whereas the TL classifier has the lowest value of recall, except in the east region, where the Boost classier has the lowest value of recall. Overall, the TL classifier has the highest value of F1 in all regions except the west region where the Naive classifier has the highest value of F1.

The Naive classifier performs surprisingly well considering its simple nature and low computational cost. We take an in-depth look into the performance of the Boost and Naive classifiers on the northeastern section of the image containing 1406 crater candidates of which 496 constitute a training set for both algorithms. Figure 6 shows the precision, recall, and F1 for these classifiers as a function of the number of features selected to construct a strong classifier. The Boost classifier clearly outperforms the Naive classifier on F1 and precision measures if more than 100 features are selected. However, the recall measures of the two classifiers remain comparable regardless of the number of selected features. Thus, the Boost classifier is superior to the Naive classifier on crater candidates that closely resemble those in the training set, but that

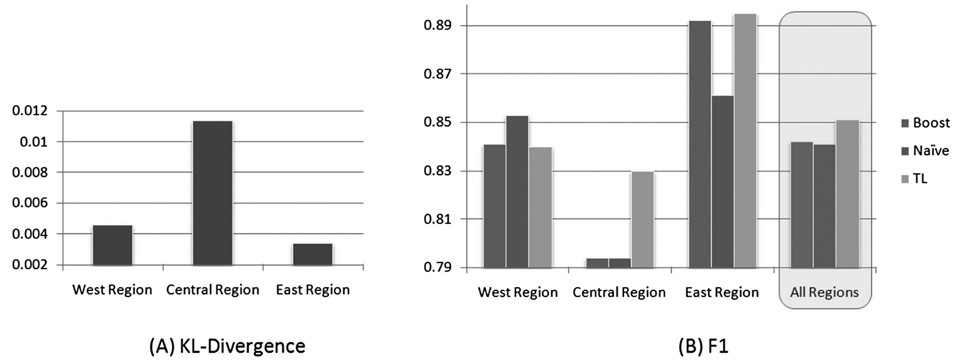


Fig. 7. (A) KL-divergence measures between the set of feature vectors in the original training set and the sets of feature vectors in the west, central, and east regions. (B) Graphical illustration of F1 scores of the three algorithms. (Best viewed in color.)

disadvantage decreases and/or disappears when classifying crater candidates that are less similar to those in the training set. We link the relatively small advantage (or lack of advantage) of the Boost classifier over the naive classifier to the peculiarity of image texture features in the context of crater detection. Top features (weak classifiers) are actually quite strong performers by themselves capable of achieving an F1 score as high as 0.81. These features limit the advantage of the boosting algorithm that works best with an ensemble of weak classifiers.

4.4. Distribution Divergence Filters with Transfer Learning

In order to better understand the results of the three proposed algorithms Boost, Naive, and TL, it is useful to assess dissimilarity between the set of feature vectors in the original training set and those in the west, central, and east regions. Figure 7(a) shows such dissimilarity as measured by the KL-divergence; Figure 7(b) plots the F1 scores graphically of the three regions. Clearly, the central region is most dissimilar to the training set, whereas the east region is the most similar (since the training set was selected from the northeastern portion of the image). This is why the TL classifier performs best (relatively to the other classifiers) in the central region. It is expected that the TL classifier would have the least advantage in the east region, as it is the region best characterized by the training set, but the results show that the TL classifier has the smallest gain (if any) in the west region. This can be explained by the fact that the west region has a similar character to the east region, but is much more heavily cratered, so in fact, relatively fewer additional training samples come from these regions resulting in no sufficient information gain to be exploited by the TL classifier.

Randomly selecting samples from the test set cannot always guarantee the quality of the selected samples. Thus, we apply the distribution divergence analysis filters to select the cotraining samples. We test the TL-Min, TL-Max, TL-MinMax filters, and TL-Random on the north half the west region (denoted as Region 1) and the north half of the central region (denoted as Region 2). Region 1 is selected as a site that closely resembles the training set, and the region is also featured with high-density subkilometer craters. Region 2 is used as a site that has a heterogeneous surface with different morphology from the training set.

The distributions of these two test sets Regions 1 and 2, and the training set are reported in Figures 8 and 9. In each figure, all the samples from the test set and training set are quantized into 1 to 50 bins. The bin sizes of different figures may

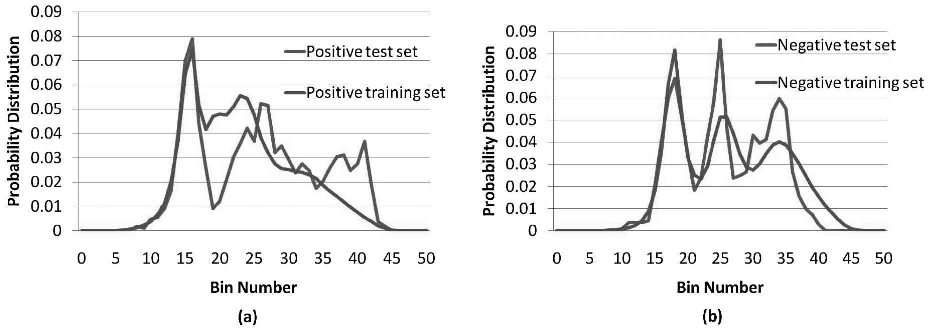


Fig. 8. Distribution comparison between Region 1 and the training set. (a) Distribution of positive samples. (b) Distribution of negative samples.

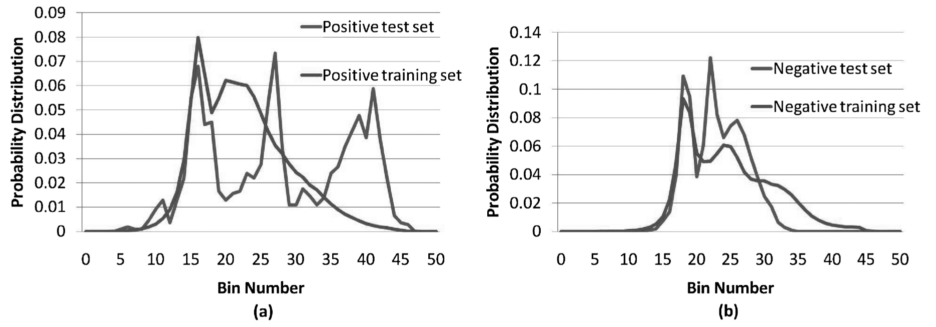
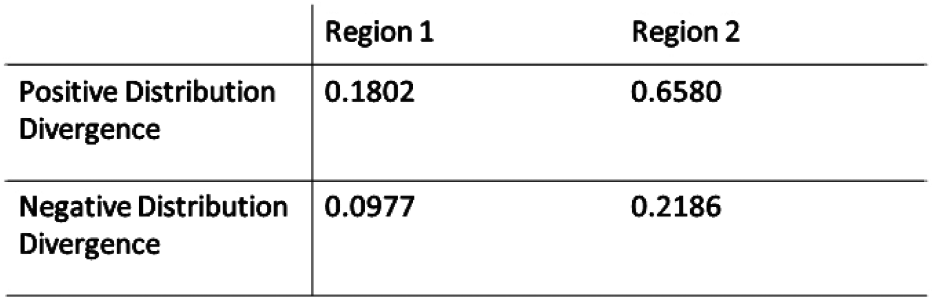


Fig. 9. Distribution comparison between Region 2 and the training set.(a) Distribution of positive samples. (b) Distribution of negative samples.

be different due to different distributions of the two test sets, thus the training set curves may vary in those two figures. In Figures 8(a) and 9(a), we can find that the blue curve is very similar to the red curve, where the blue/red curve denotes positive test/training examples (craters). However, In Figures 8(b) and 9(b), the blue curve always has big differences with the red curve, where in those figures the blue/red curve denotes negative test/training examples (noncraters). This illustrates that the craters (positive samples) are always similar and the noncraters (negative samples) are different with each other in their own ways. Furthermore, the test samples in Figure 8 distribute significantly differently from the training set than those test samples in Figure 9, which means the model trained from the training set may be more suitable in Region 1 than Region 2 because the significantly different surface morphology in Region 2. Figure 10 shows the KL- divergence and probability distributions, between positive and negative examples in the training set and Regions 1 and 2, respectively. The divergence between Region 2 and the training set is almost 3 times larger than the divergence between Region 1 and the training set.

The experimental results of the four algorithms, TL-Random, TL-Min, TL-Max, and TL-MinMax, are reported in Figure 11 for Region 1 and Figure 12 for Region 2. Figure 11 indicates that when the samples are not sufficient, TL-MinMax slightly outperforms the TL-Random and achieves its peak F1 score 0.8532 with 90 same-distribution samples. Because Region 1 is similar to the training set, the TL-Min has less contribution to improve classification performance. And TL-Max achieves better results when there are sufficient samples to select. In Figure 12, TL-MinMax and TL-Random are comparable and TL-MinMax is slightly better than TL-Random most



	Region 1	Region 2
Positive Distribution Divergence	**0.1802**	**0.6580**
Negative Distribution Divergence	**0.0977**	**0.2186**

Fig. 10. Distribution comparison between Region 1, Region 2, and the training set. The smaller the KL-distribution divergence, the similar the two sets.

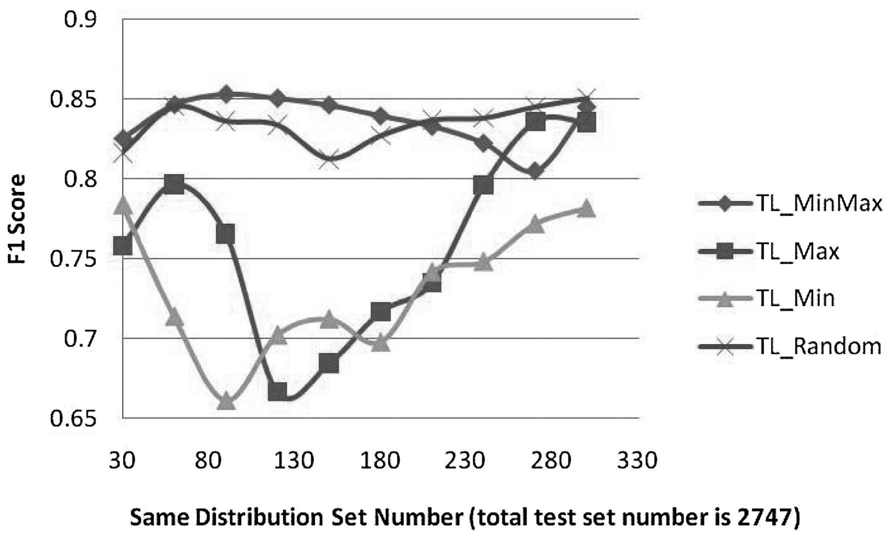


Fig. 11. F1 score versus the size of same distribution samples in Region 1. A comparison of TL-MinMax, TL-Min, TL-Max, and TL-Random algorithms.

of the time. Because of the difference between the training set and the test set, TL-Max can select the samples which have significant difference to help reconstruct the model and capture the main trend of the sample distribution. But the performance of TL-Max is limited if there is no sufficient test samples to select.

4.5. Feature Selection by Naive, Boost, and TL

It is instructive to compare top features (weak classifiers) selected by each of the three classification algorithms (Naive, Boost, and TL). Figure 13 shows six top features selected by each algorithm. The top two features selected by the three algorithms concentrate on the transition between the shadow and the highlight which best define the characteristics of a crater, but there are significant differences between other selected top features. Features selected by the Naive algorithm are relatively strong by themselves. Most of them utilize the transition between the shadow and the highlight to distinguish craters from no craters, while the next best feature selected by the Boost algorithm always attempts to correct mistakes done by the previous feature. Figure 14 illustrates how the second best feature selected by the Boost algorithm corrects the mistakes by the first best feature, and we can observe that this feature performs well

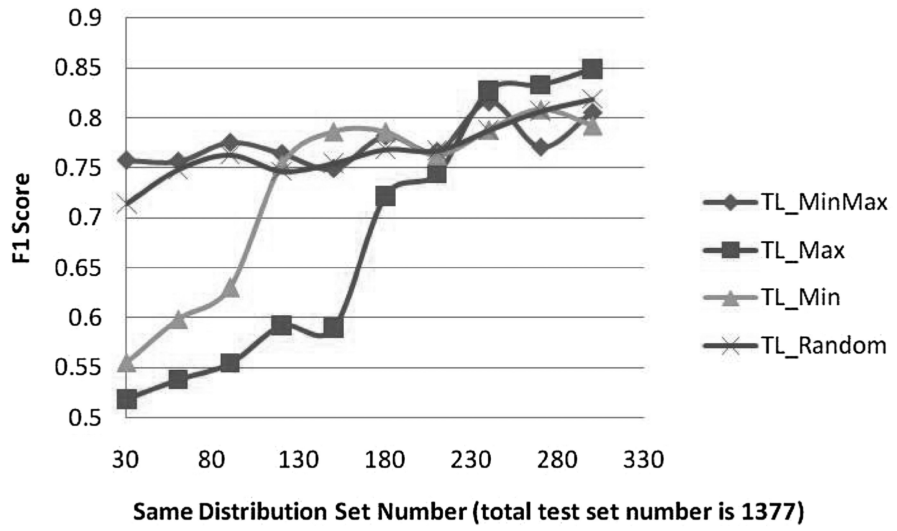


Fig. 12. F1 score versus the size of same distribution samples in Region 2. A comparison of TL-MinMax, TL-Min, TL-Max, and TL-Random algorithms.

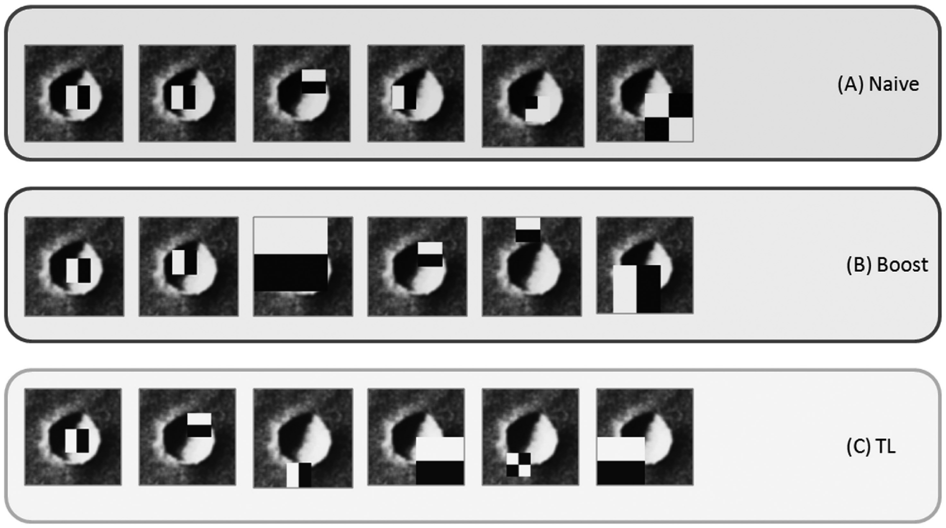


Fig. 13. Top 6 features selected by Naive, Boost, and TL with random sampling (TL-Random), respectively.

on candidates with shifted shadow regions. Not all top features selected by the TL algorithm utilize the transition between shadows and highlights, but rather crater rims. This indicates the new test data has different characteristics on crater edges.

Figure 15 displays the results of the TL algorithm, using top 150 features and the threshold $\mu = 0.500$. Notice that the large craters $\geq$5000-meter in diameter are intentionally not detected as we set the parameters of our algorithm to target small subkilometer craters (large craters on Mars have already been identified manually [Barlow 1988]).

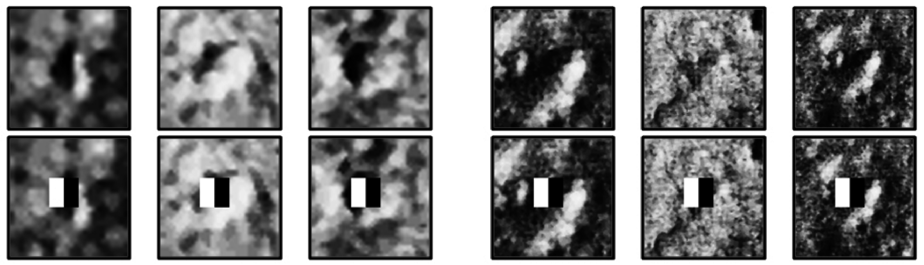

(a) 1st row: True craters misclassified by the 1st best feature; (B) 2nd row: True craters overlaid by the 2nd best feature.
(b) 1st row: Noncrater candidates misclassified by the 1st best feature; (B) 2nd row: Noncrater candidates overlaid by the 2nd best feature.

Fig. 14. The second best feature selected by the Boost algorithm successfully classified 6 misclassified examples using the first best feature.

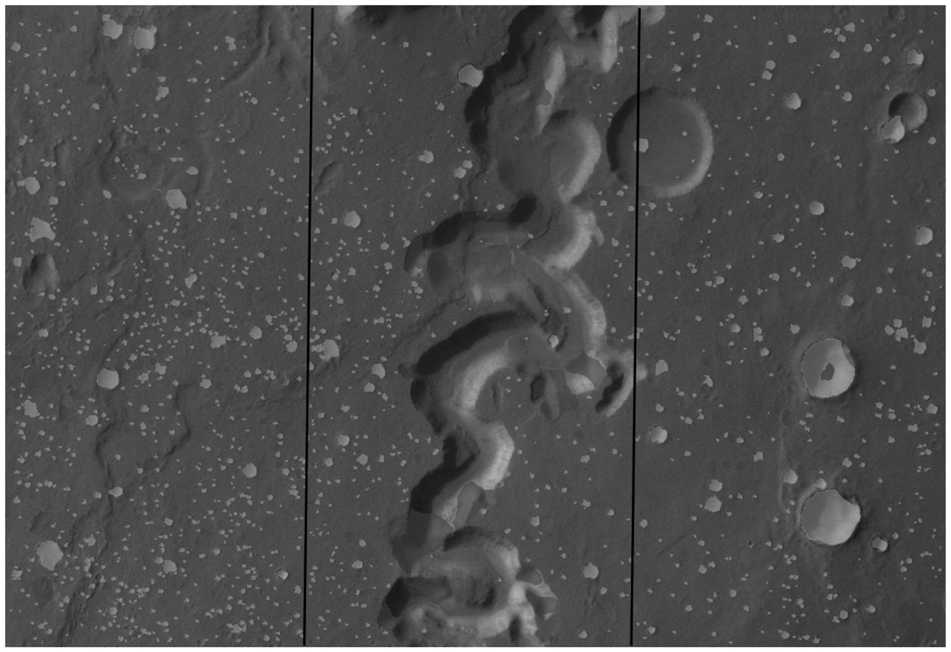


Fig. 15. Craters (<= 5000-meter in diameter) detected in a 37,500 × 56,250 m^2 test image. (Best viewed in color.) Green: True detections, Red: False detections.

4.6. Comparative Performance with Existing Algorithms

Table II provides a in-depth evaluation of the TL method with the crater detection method proposed by Urbach and Stepinski [2009]. Our method outpeforms their method on precision, recall, and F1 measure on all regions and each individual region.

We have also tested three representative algorithms for the purpose of a thorough comparative performance study: AdaBoost [Freund and Schapire 1995] with C4.5 as the base leaner for an example of boosting algorithms, SVM [Boser et al. 1992; Joachims 2002] with a linear kernel as an example of kernel-based learning algorithms, and TrAdaBoost [Dai et al. 2007] with C4.5 as the base leaner for an example of transfer learning algorithms. Using all the 1089 features, the F1 score of SVM on all regions is

Table II. Performance of our TL Method (Ours) vs. the Performance of Urbach and Stepinski's Method (Theirs)

Evaluation Metrics								
Type	All Regions		West Region		Central Region		East Region	
	Ours	Theirs	Ours	Theirs	Ours	Theirs	Ours	Theirs
Precision	0.919	0.801	0.911	0.794	0.902	0.802	0.954	0.813
Recall	0.791	0.635	0.779	0.593	0.769	0.615	0.843	0.755
F1	0.851	0.709	0.840	0.680	0.830	0.696	0.895	0.783

Improvement in Classification Performance				
Improvement	All Regions	West Region	Central Region	East Region
on Precision (%)	14.7	10.0	12.0	20.0
on Recall (%)	24.6	30.0	25.0	10.0
on F1 (%)	20.0	20.0	19.0	10.0

Parameter values: Ours–TL with 150 features selected, $\mu = 0.500$, TL-Random select 253 new test instances into the same-distribution set; Theirs–same parameter values proposed in Urbach and Stepinski's paper.

0.202, AdaBoost is 0.302, TrAdaBoost is slightly better than 0.4. As we can see from Figure 9, the three algorithms designed in this article can achieve an F1 score above 0.85.

The huge performance gain by the three algorithms (Boost, Naive, and TL) is because the proposed algorithms intelligently select and integrate subset of best features out of all 1089 features to build a strong ensembled classifiers using boosting. The 1089 features are overcompleted by contructing 9 maskes in different scales, stepwise, and positions. Without a built-in mechanism on feature selection to remove irrelevant and redunt features, the AdaBoost, SVM, and TrAdaBoost classifiers cannot perform well. Comparable results would be obtained on the crater detection, if similar feature set is used on those classifiers. However, this approach is less desirable as feature selection and classifier induction have already been simultaneously integrated into the learning process of the three proposed algorithms.

5. CONCLUSIONS

The aim of this article is to present a robust and reliable framework for autodetection of small craters in high-resolution images of planetary surfaces. This is one of the most challenging problems in planetary science: effective and automatic crater detection from extremely large orbiter images. The framework uses an innovative method that integrates improved techniques on embedding feature selection with supervised classification, and transfer learning. First, we have demonstrated that our method identifies craters with high accuracy. The test site is an HRSC image of Martian scene that presents a heterogeneous region of $37{,}500 \times 56{,}250\ m^2$, and detecting craters in various forms is challenging using regular algorithms. Our approach can achieve an F1 score above 0.85, and provides a reliable mechanism for planetary research. Second, we have demonstrated that a consistently accurate detection can be achieved through transfer learning. Without transfer learning the performance of our algorithms (Boost and Naive) decreases in the central region of the image where surface morphology differs as characterized by the training set. However, using the TL algorithm partially restores the level of performance. Third, we noticed that the Naive algorithm can perform well in the context of crater detection for a fraction of the computational cost of the Boost algorithm.

We contend that the robustness and reliability of our methodology make it an effective tool for planetary research. If adopted, our approach has great potential to produce surveys of small craters over entire surfaces of planets, thus revolutionizing certain aspects of planetary science. Our future research will address means of efficient selection of additional training samples for construction of the same-distribution for transfer learning. The goal is to intelligently select samples that exemplify differences between the existing training sets and new candidate sets.

REFERENCES

ANDERSSON, L. E. AND WHITAKER, E. A. 1982. Nasa catalog of Lunar nomenclature. In *NASA Reference Publication 1097*.

BANDEIRA, L., SARAIVA, J., AND PINA, P. 2007. Impact crater recognition on Mars based on a probability volume created by template matching. *IEEE Trans. Geosci. Remote Sensing*. 4008–4015.

BARLOW, N. G. 1988. Crater size-frequency distributions and a revised Martian relative chronology. *Icarus 75*, 285–305.

BOSER, B. E., GUYON, I. M., AND VAPNIK, V. N. 1992. A training algorithm for optimal margin classifiers. In *Proceedings of the 5th Annual Workshop on Computational Learning Theory (COLT'92)*. 144–152.

BURL, M. C., STOUGH, T., COLWELL, W., BIERHAUS, E. B., MERLINE, W. J., AND CHAPMAN, C. 2001. Automated detection of craters and other geological features. In *Proceedings of the International Symposium on Artificial Intelligence Robotics and Automated Space*.

CHENG, Y., JOHNSON, A. E., MATTHIES, L. H., AND OLSON, C. F. 2002. Optical landmark detection for spacecraft navigation. In *Proceedings of the 13th Annual AAS/AIAA Space Flight Mechanics Meeting*. 1785–1803.

CRATER ANALYSIS TECHNIQUES WORKING GROUP. 1979. Standard techniques for presentation and analysis of crater size-frequency data. *Icarus 37*, 467–474.

DAI, W., YANG, Q., XUE, G.-R., AND YU, Y. 2007. Boosting for transfer learning. In *Proceedings of the 24th International Conference on Machine Learning (ICML'07)*. 193–200.

DING, W., STEPINSKI, T. F., BANDEIRA, L., VILALTA, R., WU, Y., LU, Z., AND CAO, T. 2010. Automatic detection of craters in planetary images: An embedded framework using feature selection and boosting. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management (CIKM'10)*.

FREUND, Y. AND SCHAPIRE, R. 1995. A decision-theoretic generalization of on-line learning and an application to boosting. In *Proceedings of the European Conference on Computational Learning Theory (EuroCOLT)*. Springer-Verlag, 23–37.

HONDA, R., IIJIMA, Y., AND KONISHI, O. 2002. Mining of topographic feature from heterogeneous imagery and its application to lunar craters. In *Progress in Discovery Science*, Final Report of the Japanese Discovery Science Project, Springer-Verlag, 395407.

HOUGH V, P. C. 1962. Method and means for recognizing complex patterns. United States Patent.

HRSC DATA BROWSER. 2009. http://europlanet.dlr.de/node/index.php?id=209.

JOACHIMS, T. 2002. Learning to classify text using support vector machines. Ph.D. thesis, Kluwer Academic Publishers.

KIM, J., MULLER, J.-P., VAN GASSELT, S., MORLEY, J., AND NEUKUM, G. 2005. Automated crater detection: A new tool for Mars cartography and chronology. *Photogram. Engin. and Remote Sensing 71,* 10, 1205–1217.

KULLBACK, S. AND LEIBLER, R. 1951. On information and sufficiency. *Ann. Math. Statist. 22,* 1, 79–86.

LEROY, B., MEDIONI, G., AND MATTHIES, E. J. L. 2001. Crater detection for autonomous landing on asteroids. *Image Vision Comput. 19*, 787–792.

MARTINS, R., PINA, P., MARQUES, J., AND SILVEIRA, M. 2009. Crater detection by a boosting approach. *IEEE Geosci. Remote Sensing Lett. 6*, 127–131.

PAN, S. J. AND YANG, Q. 2010. A survey on transfer learning. *IEEE Trans. Knowl. and Data Engin. 22,* 10, 1345–1359.

PAPAGEORGIOU, C., OREN, M., AND POGGIO, T. 1998. A general framework for object detection. In *Proceedings of the 6th International Conference on Computer Vision*. 555–562.

SALAMUNICCAR, G. AND LONCARIC, S. 2007. Open framework for objective evaluation of crater detection algorithms with first test-field subsystem based on mola data. *Adv. Space Res. 42,* 1, 6–19.

TANAKA, K. L. 1986. The stratigraphy of Mars. *J. Geophys. Res. 91*, E139–E158.

URBACH, E. R., ROERDINK, J. B. T. M., AND WILKINSON, M. H. F. 2007. Connected shape-size pattern spectra for rotation and scale-invariant classification of gray-scale images. *IEEE Trans. Patt. Anal. Mach. Intell. 29*, 272–285.

URBACH, E. R. AND STEPINSKI, T. F. 2009. Automatic detection of sub-km craters in high resolution planetary images. *Planet. Space Sci. 57*, 880–887.

VINOGRADOVA, T., BURL, M., AND MJOLSNESS, E. 2002. Training of a crater detection algorithm for Mars crater imagery. In *IEEE Aerospace Conference Proceedings*. Vol. 7. 7–3201–7–3211.

VIOLA, P. AND JONES, M. J. 2004. Robust real-time face detection. *Int. J. Comput. Vis. 57*, 137–154.

WETZLER, P., HONDA, R., ENKE, B., MERLINE, W., CHAPMAN, C., AND BURL, M. 2005. Learning to detect small impact craters. In *Proceedings of the 7th IEEE Workshops on Application of Computer Vision*. Vol. 1. 178–184.

Received September 2010; revised November 2010; accepted December 2010