Learning to Detect Small Impact Craters

Conference Paper · January 2005

DOI: 10.1109/ACVMOT.2005.68 · Source: DBLP

6 authors, including:

Philipp G. Wetzler
Google Inc.
10 PUBLICATIONS **154** CITATIONS
SEE PROFILE

Rie Honda
Kochi University
66 PUBLICATIONS **473** CITATIONS
SEE PROFILE

William J. Merline
Southwest Research Institute
227 PUBLICATIONS **3,622** CITATIONS
SEE PROFILE

Learning to Detect Small Impact Craters

P.G. Wetzler[1], R. Honda[2,1], B. Enke[3], W.J. Merline[3], C.R. Chapman[3], M.C. Burl[1]
[1]University of Colorado at Boulder, [2]Kochi University, [3]Southwest Research Institute

Abstract

Machine learning techniques have shown considerable promise for visual inspection tasks such as locating human faces in cluttered scenes. In this paper, we examine the utility of such techniques for the scientifically-important problem of detecting and catloging impact craters in planetary images gathered by spacecraft. Various supervised learning algorithms, including ensemble methods (bagging and AdaBoost with feed-forward neural networks as base learners), support vector machines (SVM), and continuously-scalable template models (CSTM), are employed to derive crater detectors from ground-truthed images. The resulting detectors are evaluated on a challenging set of Viking Orbiter images of Mars containing roughly one thousand craters. The SVM approach with normalized image patches provides detection and localization performance closest to that of human labelers and is shown to be substantially superior to boundary-based approaches such as the Hough transform.

1. Introduction

Craters are the most abundant landform in the solar system. Their size-frequency distribution, especially at the more abundant smaller sizes, is used extensively by scientists to judge the relative ages of planetary and asteroidal surfaces. Due to similarities in their initial morphologies, craters provide significant clues about the other processes (wind, lava, etc.) that have acted at various times and locations on the surface. Craters also provide convenient landmarks for visual motion estimation [17].

The surface of Mars is of particular interest, especially given the recent rover data that shows the planet was once wet for an extended period of time. High-resolution orbital imagers are providing unprecedented views of Mars, but the data volume is so large that the returned images cannot be thoroughly analyzed manually. The Barlow *Catalog 1.0 of Martian Impact Craters* [2], which was created manually during the mid 1980's from Viking imagery, contains approximately $40,000$ craters down to size $\sim$ 5km; however, in a heavily-cratered region one might find thousands of craters *in a single image* if the analysis is carried down to the smallest sizes supported by the Viking image resolution and recent missions yield resolutions one to two orders of magnitude better than that of the Viking missions.

More recent efforts toward manual cataloging have focused on using large groups of volunteer labelers (clickworkers) [18] distributed over the internet [23]. While such an approach may be an expedient path for creating catalogs in the short term, it suffers from the problem that the entire human effort must be repeated for each new dataset and each new feature type, e.g., if the target is changed from craters to gullies. A large manual effort on one dataset provides little leverage to another dataset of different type, e.g., manual detection of craters in a planetary dataset would have essentially no value for other visual inspection tasks such as counting blood cells in biomedical slides.

Hence, there is strong motivation for exploring methods to automate or assist in the visual inspection/detection process. Inspired by the considerable success of learning techniques applied to human face detection [25, 24, 27, 29, 31, 22], we have attempted to utilize a learning approach to facilitate comprehensive scientific analysis of the cratering record.

2. Related Work

Previous attempts to automate crater detection have largely focused on looking for circular or elliptical arrangements of edges along the crater boundary, e.g., using the Hough transform. Examples include the works of Cross [12], Cheng [11], Honda [15], and Leroy [21]. These boundary-based approaches seem to perform well under certain conditions, for example, for detecting medium to large craters (relative to the image resolution) when there is limited texture in the background due to other features or processes. However, the techniques break down severely for detecting smaller craters or when applied in more challenging terrain.

An alternative to boundary-based detection is to look directly at the pixel-level pattern in an image patch. Along these lines, Chapman [10] made an early attempt to automatically detect Lunar craters imaged at low sun angles by looking for adjacent bright-dark regions of the proper relative size. We have also used image-based techniques including principal components analysis (PCA) [6] and continuously-scalable template models [7] to detect various planetary features (volcanoes, craters, blocks). The learn-

Proceedings of the Seventh IEEE Workshop on Applications of Computer Vision (WACV/MOTION'05)
0-7695-2271-8/05 $ 20.00 IEEE

ing techniques we consider here are similar in that they also work directly with image patches.

3. Problem Formulation

In this section, we formulate the crater detection problem as one of supervised machine learning. We presume the existence of a training set $\mathcal{T}$ consisting of tuples $(\mathbf{x}, y)$, where $\mathbf{x}$ is a fixed-length vector of observations and y is a label from the set $\{-1, +1\}$. The label -1 indicates that $\mathbf{x}$ is an instance of the "negative class" (`not-crater`) whereas $+1$ indicates that $\mathbf{x}$ is an instance of the "positive class" (`crater`). The learning algorithm examines the training set and attempts to produce a function $g(\cdot)$ that will correctly map future instances $\mathbf{x}$ to the correct class. Most of the learning algorithms we consider produce a continuous-valued output that can be thresholded to provide a class decision. By varying the threshold, one varies the tradeoff between detection and false alarm rates.

3.1. Handling Size Variation

The size of the craters in an image can range from a few pixels in diameter to 400 pixels (or more). However, the standard learning techniques require a fixed-length vector of observations $\mathbf{x}$. To manage this problem, we use a multiscale "pyramid" representation of the image [1], so that a crater with a particular diameter d_0 in the original resolution data will have an apparent diameter at level l of the image pyramid that is given by:

$$d_l = \frac{d_0}{\lambda^l} \quad (1)$$

where λ is the magnification factor between adjacent pyramid levels (original resolution is $l = 0$).

When choosing positive instances for the training set, a fixed-size window that is centered on the crater is selected from pyramid level l, where l is such that $d_l \in [d_{\min}, d_{\max}]$. At detection time, the window is scanned[1] across each level of the pyramid and the pattern of pixel values under the window is used to classify the patch as positive or negative.

In setting $d_{\min}$ and $d_{\max}$ it is important to note that if the crater diameter relative to the patch size is too small, then important details of the pattern will be lost (also, much of the patch will consist of irrelevant background pixels). If the crater diameter is too large, then the patch will not contain enough context information. (Experimentally, we know that such information is helpful for making the correct classification.) In our experiments, we used $d_{\min} = 7.125$, $d_{\max} = 8.55$, and $\lambda = 2^{0.25}$, with the window size fixed at (19×19). The magnification factor λ insures that each crater falls within $[d_{\min}, d_{\max}]$ in exactly one of the pyramid levels.

[1] While this is conceptually the case, several of the implementations use more efficient FFT-based techniques.

Note that detecting craters at the smallest size (say $d_0 = 5$ pixels) requires supplying "negative" pyramid levels in which the original resolution data is interpolated to produce a larger image. We have shown experimentally that using negative pyramid levels is better performance than simply using smaller window sizes (e.g., (11×11)) to detect the smallest craters.

3.2. Contrast Normalization

In addition to size variation, individual image patches may exhibit very different overall average brightness and contrast. Although one can leave it to the algorithms to learn to ignore such differences, it is also possible to explicitly normalize the patches. Given an image patch $\mathbf{x}$, we define the normalized version of the patch by:

$$\tilde{\mathbf{x}} = \frac{\mathbf{x} - \mu\mathbf{1}}{\sigma\sqrt{n}} \quad (2)$$

where μ is the mean of the values in $\mathbf{x}$, σ is the standard deviation, n is the number of values, and $\mathbf{1}$ is a vector of ones. Normalization, when used, is applied to both the training and test patches.

3.3. Defining the Training Set

To create the training sets for the learning algorithms, we start from a set of images for which all of the true craters have been labeled (circled). In most cases, the absolute ground truth is not known; therefore, labelings provided by a human expert are used as a surrogate for true ground truth. Since the size-frequency distribution of craters at the smallest sizes is of most interest for assessing relative ages of surfaces, we want to detect craters down to the smallest sizes. In this study a minimum target diameter of 5 pixels was used (any craters significantly smaller than this size are removed from the ground truth). Given the training images and ground truth labelings, it is straightforward to extract the patches that are true positives from the image pyramids. To provide for some translation invariance, patches that are shifted by one pixel from the true crater centers are also included.

The choice of negative examples for the traning set is more difficult. If the image size is $(z_r \times z_c)$, then taking every image patch that doesn't contain a centered crater of the proper size as a negative example would result in millions of negative examples (compared to thousands of positive examples). These numbers are problematic for two reasons: (i) the size of the training set (even learning algorithms that perform linearly in the number of examples may be close to infeasible to run with millions of examples) and (ii) the huge imbalance between positives and negatives (the learning algorithms will be tempted to simply classify everything as a negative). We deal with these problems by selecting a random subset from all the potential negative

Proceedings of the Seventh IEEE Workshop on Applications of Computer Vision (WACV/MOTION'05)
0-7695-2271-8/05 $ 20.00 IEEE

examples. The size of this set is chosen to be a factor of six or so larger than the set of positive examples to capture the greater variability among the negative patches, but still leave enough incentive for the learning algorithm to use the positive examples.

We also note that one of the algorithms, the continuously-scalable template model (CSTM), is trained only from positive examples; hence, this algorithm ignores any negative examples in the training set.

3.4. Spatial Clustering

Suppose we have a window centered over a crater, which has the right apparent size relative to the window. If we shift the window by a pixel or two in any direction, the pattern within the shifted window is still going to look like a crater. Thus, it is to be expected that a single crater may cause multiple detections. Of course, if the goal is to count craters, then these duplicate detections are a problem. To remedy this situation, the collection of outputs from each detector (a set of circles of different sizes and locations) are passed through a spatial clustering algorithm. Detections for which the relative area of overlap[2] between circles exceeds a predetermined threshold are presumed to be due to the same object. The circle within a cluster that receives the highest confidence score from the classifier is considered to be the true object and the others are discarded as duplicates.

4. Learning Algorithms

We applied a number of supervised learning algorithms to the crater detection problem, including: feed-forward neural networks, ensemble methods (bagging and AdaBoost with feed-forward neural networks as base learners), support vector machines (SVM), and continuously-scalable template models (CSTM). In this section, we briefly outline these techniques and give pointers to more detailed descriptions in the literature.

4.1. Feed-Forward Neural Networks

Feed-forward neural networks consist of a set of *units* that produce an output value by applying a nonlinear squashing function (typically sigmoid or tanh) to a weighted combination of their inputs [14]. The units are commonly arranged in layers so that the outputs from one layer of units serve as the inputs to the next layer of units. The back-propagation algorithm [26] offers a convenient way to train the network by adjusting the network weights so as to minimize the error between the actual outputs of the network and the target outputs (y-values from the training set). Feed-forward neural networks have been used successfully in a number of visual pattern recognition applications [25, 20]. Initially, we considered both the standard topology networks with a single hidden layers and convolutional networks similar to those used by LeCun on the MNIST data. Although we were able to reproduce the MNIST results with our implementation of convolutional neural networks, they did not perform well in initial experiments on the crater data, so they were not used in the full scale experiments.

[2]Since craters can be nested (a small crater can occur inside a larger crater), the relative area of overlap measures the ratio between the area of intersection between the two circles and the area of the larger of the two.

4.2. Bagging and Boosting

Ensemble methods offer a way to improve accuracy and reliability by training and combining multiple classifiers in an intelligent way. Bagging [4] simply trains multiple classifiers on different randomly selected subsets of the training set and averages or majority votes the individual decisions to produce a composite classifier. Boosting is a more directed approach in which examples that prove difficult to one classifier are reweighted and given enhanced attention by the next classifier. In our experiments, we used AdaBoost [13], which has previously been used successfully for face detection [27, 31]. The Kullback-Leibler boosting algorithm [22] also appears promising and may be evaluated in future work. In our experiments, we used the feedforward neural network as the base learning algorithm and performed six rounds of boosting (we also experimented with more rounds, but did not find significantly better results). The weightings placed on the examples by the boosting algorithm were directly incorporated into the backpropagation/gradient descent algorithm used to train the base neural networks.

4.3. Support Vector Machines

The underlying idea of Support Vector Machines [30, 5] is to find a hyperplane that optimally separates the positive from the negative examples, where optimality is defined by the size of the margin, i.e., the smallest distance between a positive and a negative example when mapped onto the hyperplane's normal vector. Using *kernels* [19], the input examples can be implicitly lifted into a higher-dimensional space in which the examples can be more easily separated. The hyperplane decision surface in this higher dimensional space corresponds to a non-linear surface in the original input space. These decision surfaces are specified in terms of a subset of the training examples (the support vectors) and a set of weights (LaGrange multipliers or alpha values). Support vector machines have provided very favorable results on a number of visual detection/classification problems, including face detection [24] and the MNIST digit database.

In our experiments, we used the `libSVM` implementation [9] to carry out the training. The kernel type (RBF) and other parameters were selected through cross-validation. The SVM models that were produced in training were far more complex than the models produced by the other meth-

Proceedings of the Seventh IEEE Workshop on Applications of Computer Vision (WACV/MOTION'05)
0-7695-2271-8/05 $ 20.00 IEEE

ods. We typically found SVM models with ∼5000 support vectors. Applying such models to image data is computationally demanding (due both to the large number of support vectors and the large number of test patches). Alternative formulations of the SVM objective function, such as the ν-SVM proposed by [28], allow more explicit control over the number of support vectors. We conducted several experiments with ν-SVM to see if useful models with fewer support vectors could be found. We also developed an exact blocked-FFT implementation of the SVM decision function [8] so that performance of the full scale models could be assessed in a feasible amount of time.

4.4. CSTM

For comparison, we also evaluated an approximate implementation of the continuously-scalable template matching algorithm (CSTM) described in [7]. This algorithm, which has previously been applied to crater detection, is based on the idea of generating a family of matched filters at densely-sampled scales using a single example crater as the prototype for the family. The resulting family of filters is then applied to the images in an efficient way by applying PCA to compress the family into a smaller number of basis functions. We approximate this technique by omitting the PCA step and instead performing brute-force matching (calculation of correlation coefficient between each family member and each template in the family). This modification sacrfices some speed, but does not significantly alter the results. The choice of the prototypes for generating the filter families is made from the training set by sequentially choosing the positive example that covers the largest number of positive examples not already covered by an earlier prototype, where an example *covers* another if the correlation coefficient between the two exceeds some pre-determined threshold.

5. Experiments

Although there are a large number of planetary images available on the internet, detailed ground truth in machine readable form is generally not available. Hence, we tested the various algorithms on a set of seven Viking Orbiter images of size (1280×1280), for which ground truth labelings were established several years ago by one of the authors. The images contain ∼1100 craters down to 5 pixels in dimaeter.

All experiments were conducted in a cross-validation manner. Six of the seven images were used to generate a training set. Each learning algorithm derived a detector (the function $g(\cdot)$) from the training set. Each detector was then applied to the seventh image to produce a labeling (set of circles of various sizes at various locations). The raw labeling was processed by the spatial clustering algorithm described earlier to remove duplicate detections. This process

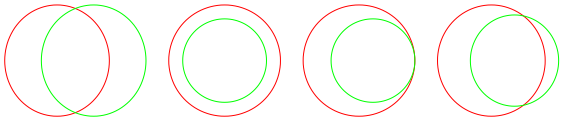

Figure 1: Circles that "just match" according to their relative area of overlap.

was repeated with each image taking a turn as the holdout (test image). The final labelings were automatically scored as detailed below.

6. Scoring

To quantitatively assess the performance of different algorithms, an automatic scoring procedure was developed to compare the labeling produced by an algorithm with the ground truth labeling. For each ground truth crater, the best matching detection above the detection threshold T and with overlap greater than the overlap threshold O is identified. Any other detections that match but are not the best match are reported as false alarms. Detections that are not matched to any ground truth marker are also counted as false alarms. The criteria for matching a detection and a ground truth marker is the relative area of overlap described in Section 3.4. The overlap threshold for deciding that there is a match is set to require fairly accurate localization and sizing of the crater as shown in Figure 1.

7. Results

The overall results are presented as receiver operating characteristics (ROC), i.e, as curves showing the trade-off between probability of detection (y-axis) and number of false alarms (x-axis) as the detection threshold T is varied. These curves can be translated into recall-vs-precision by noting that:

$$\text{precision} = \frac{N_+ y}{N_+ y + x}; \qquad \text{recall} = y \qquad (3)$$

where N_+ is the number of positive examples in the test set (∼1100 here).

7.1. Algorithm Performance

The ROC performance for selected algorithms in our experiments is shown in Figure 2. The curves for the algorithms that are not shown (bagging, boosting, neural networks, and ν-SVM fall below the CSTM curve, but above the Hough transform curve; they have been omitted from the diagram for clarity. The curve labeled `SVM-step4` corresponds to an initial SVM experiment in which we scanned the image pyramids by shifting the (19×19) window by four pixels at a time in order to reduce the computation time required. This spatial subsampling was only performed for

Proceedings of the Seventh IEEE Workshop on Applications of Computer Vision (WACV/MOTION'05)
0-7695-2271-8/05 $ 20.00 IEEE

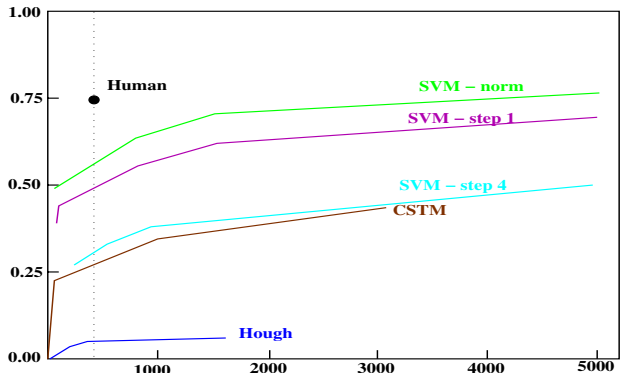

Figure 2: ROC performance for selected algorithms on the test set. The performance for the ensemble methods, neural networks, and ν-SVM have been omitted for clarity; those curves fall below the CSTM curve but above the Hough transform curve.

the SVM detector; the other detectors shifted the window by one pixel at a time. After observing that the SVM approach, even with step 4, was slightly better than the other approaches, we went back and developed a more efficient implementation based on blocked-FFTs [8] that allows the exact SVM detector to be applied at every pixel. This curve is labeled `SVM-step1` and does indeed significantly outperform `SVM-step4`. In a final experiment, we trained the SVM detector with normalized patches and also normalized the test patches, which produced further gains in performance as shown by the `SVM-norm` curve.

7.2. Human Performance

Since the current practice for crater cataloging is manual labeling [2, 18, 3], it is useful to understand human performance on our test set. One of the authors, who provided the original ground truth labeling several years ago, recently generated a second ground truth labeling of the same images. The consistency between the old and new labelings provides an estimate of the consistency with which humans can perform the crater labeling task. The human performance point (treating the old labeling as ground truth and scoring the new labeling against that) is at approximately $(x = 300, y = 0.75)$. More detailed comparisons between the two labelings show that the inconsistency is due to several factors: oversight, inconsistent handling of partial craters on the image boundaries, localization and sizing errors on the smaller craters, and genuine ambiguity in the image patches. The human performance point is shown on Figure 2 as a small, filled circle. The vertical line through this point allows comparison of the algorithms at the same false alarm rate as the human labeler.

7.3. Discussion

The SVM algorithm (using radial basis function kernels) was significantly better than the other algorithms, although the performance comes at a relatively large computational cost. The number of support vectors in each of the learned models was $\sim$5000. As mentioned earlier, to even complete the initial experiments in a reasonable time, we found it necessary to slide the detection window by four pixels at a time. Using ν-SVM [28], which allows the number of support vectors to be more explicitly controlled, we found models with several hundred support vectors in our experiments, but the performance was significantly degraded. The development of a blocked-FFT method to implement the SVM detector allowed the full experiments to be carried out in time comparable to the direct spatial scanning with skip of four. The sequential reduced set method that is reported in [24] is also a possibility for speeding up the runtime of SVM on image data.

The CSTM performance curve shown in the figure is based on a single template selected automatically from the training set. Somehwat surprisingly, the performance of this algorithm was better than virtually all of the learning approaches except for SVM. Most likely this reflects the fact that a majority of the craters in these images have a bowl-floored morphology and are therefore very consistent in their appearance (except for size) so handling the scaling issues provides significant pay-off. We also conducted additional experiments that are not reported here using up to six CSTM prototypes and combining the results. This provided a modest increase in performance (slightly above the `SVM-step4` curve). Comparing the labelings produced by CSTM and SVM-norm shows very clearly the reason for the difference inperformance. The CSTM detector has a significant number of false alarms on the periphery of the large craters as well as on the bright linear features (rilles) that appear in the images. The SVM, by also considering the negative examples, is able to learn to reject these common sources of false alarms and therefore attains better overall performance.

We also applied the Hough transform [16] algorithm to this dataset (using the implementation of [15]). Although the Hough transform does a reasonable job of detecting some of the medium and large-sized craters, the overall performance is quite bad (comparable to a three hidden unit neural network).

8. Conclusion

Automatic crater cataloging is difficult due to the variability in appearance of the craters and surrounding terrain. We applied a variety of state-of-the-art learning algorithms, which have worked well in other pattern recognition and vision problems, to a challenging set of Viking Orbiter images of the surface of Mars. The best results (detection of $\sim$60% of

Proceedings of the Seventh IEEE Workshop on Applications of Computer Vision (WACV/MOTION'05)
0-7695-2271-8/05 $ 20.00 IEEE

the true craters with 300 false positives) were achieved using the support vector machine methodology with normalized image patches for both training and testing. This approach was significantly better than the other methods tried, including a boundary-based (Hough transform) algorithm. Although the performance does not quite reach human levels, the results are very significant as they provide for a useful "intelligent assistant" that can speed the labeling process by automatically marking craters and allowing a human operator to edit/correct the results. Larger scale experiments with high resolution data from the MGS and Mars Odyssey missions are planned.

References

[1] E.H. Adelson, C.H. Anderson, J.R. Bergen, P.J. Burt, J.M. Ogden, "Pyramid Methods in Image Processing", *RCA Engineer*, 29(6):33-41, (1984).

[2] N.G. Barlow, "Revision of the Catalog of Large Martian Impact Craters", Sixth Int. Conf. on Mars, (2003).

[3] E.B. Bierhaus, C.R. Chapman, W.J. Merline, "Europa's Small Crater Population Revealed", DPS-34, (2002)

[4] L. Breiman, "Bagging Predictors", *Machine Learning*, 24(1):123-140, (1996).

[5] C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition", *DMKD*, 2(2):121-167, (1998).

[6] M.C. Burl, L. Asker, P. Smyth, U.M. Fayyad, P. Perona, L. Crumpler, and J. Aubele, "Learning to Recognize Volcanoes on Venus", *ML*, 30(2/3):165–194, (Feb/Mar 1998)

[7] M.C. Burl, W.J. Merline, W. Colwell, E.B. Bierhaus, C.R. Chapman, "Automated Detection of Craters and Other Geological Features", *iSAIRAS*, (2001).

[8] M.C. Burl, P.G. Wetzler, "Resource-constrained Application of Support Vector Machines to Sensor Data", *Data Mining in Resource-Constrained Environments*, (Apr 2004).

[9] C.-C. Chang, C.-J. Lin, "LIBSVM: A Library for Support Vector Machines.", URL: `http://www.csie.ntu.edu.tw/ cjlin/libsvm`.

[10] C.R. Chapman, (private communication).

[11] Y. Cheng, A.E. Johnson, L.H. Matthies, C.F. Olson, "Optical Landmark Detection for Spacecraft Navigation", In *13th Annual AAS/AIAA Space Flight Mech Mtg.*, (Feb 2002).

[12] A.M. Cross, "Detection of circular geological features using the Hough transform" *International Journal of Remote Sensing*, 9(9):1519–1528 (1988).

[13] Y. Freund and R.E. Schapire, "A Decision-theoretic Generalization of On-line Learning and an Application to Boosting", *J. Comp. & Sys. Sciences*, 55(1):119-139, (1995).

[14] J. Hertz, A. Krogh, R. Palmer, *Introduction to the Theory of Neural Computation*, (1991).

[15] R. Honda, O. Konishi, "Data Mining System for Planetary Images Crater Detection and Categorization", URL: `citeseer.nj.nec.com/436870.html`, (2002).

[16] P.V.C. Hough, "Method and Means for Recognizing Complex Patterns", *U.S. Patent 3069654*, (Dec 1962).

[17] A.E. Johnson, L.E. Matthies, "Precise Image-Based Motion Estimation for Autonomous Small Body Exploration", *iSAIRAS*, pp. 627-634, (1999).

[18] B. Kanefsky, N.G. Barlow, V.G. Gulick, "Can Distributed Volunteers Accomplish Massive Data Analysis Tasks", *LPSC*, (2001).

[19] URL: `http://www.kernel-machines.org/`

[20] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, "Gradient-Based Learning Applied to Document Recognition", In *Proc. of the IEEE*, (Nov 1998).

[21] B. Leroy, G. Medioni, A.E. Johnson, L.H. Matthies, "Crater Detection for Autonomous Landing on Asteroids", In *Workshop on Perception for Mobile Agents*, (Jun 1999).

[22] C. Liu, H.-Y. Shum, "Kullback-Leibler Boosting", *CVPR*, pp. 587-594, (Jun 2003).

[23] National Aeronautics and Space Administration. National Space Science Data Center. *http://nssdc.gsfc.nasa.gov/*

[24] E. Osuna, R. Freund, F. Girosi, "Training Support Vector Machines: An Application to Face Detection", *CVPR*, pp. 130-136, (1997).

[25] H. A. Rowley, S. Baluja, T. Kanade. "Neural-Network Based Face Detection", *TPAMI*, 20(1):23-38, (1998).

[26] D.E. Rummelhart, G.E. Hinton, R.J. Williams, "Learning Representations by Backpropoagating Errors", *Nature*, 323:533-536, (1986).

[27] H. Schneiderman and T. Kanade, "Probabilistic Modeling of Local Appearance and Spatial Relationships for Object Recognition", *CVPR*, pp. 45-51, (1998).

[28] B. Schölkopf, A.J. Smola, R.C. Williamson, P.L. Bartlett, "New Support Vector Algorithms", *Neural Computation*, 12:1207-1245, (2000).

[29] K.-K. Sung, T. Poggio, "Example-based learning for view-based Human Face Detection, *TPAMI*, 20(1):39-51, (1998).

[30] V.N. Vapnik, *The Nature of Statistical Learning*, Springer, (1995).

[31] P. Viola and M. Jones, "Rapid Object Detection Using a Boosted Cascade of Simple Features", *CVPR*, (2001).

Proceedings of the Seventh IEEE Workshop on Applications of Computer Vision (WACV/MOTION'05)
0-7695-2271-8/05 $ 20.00 IEEE

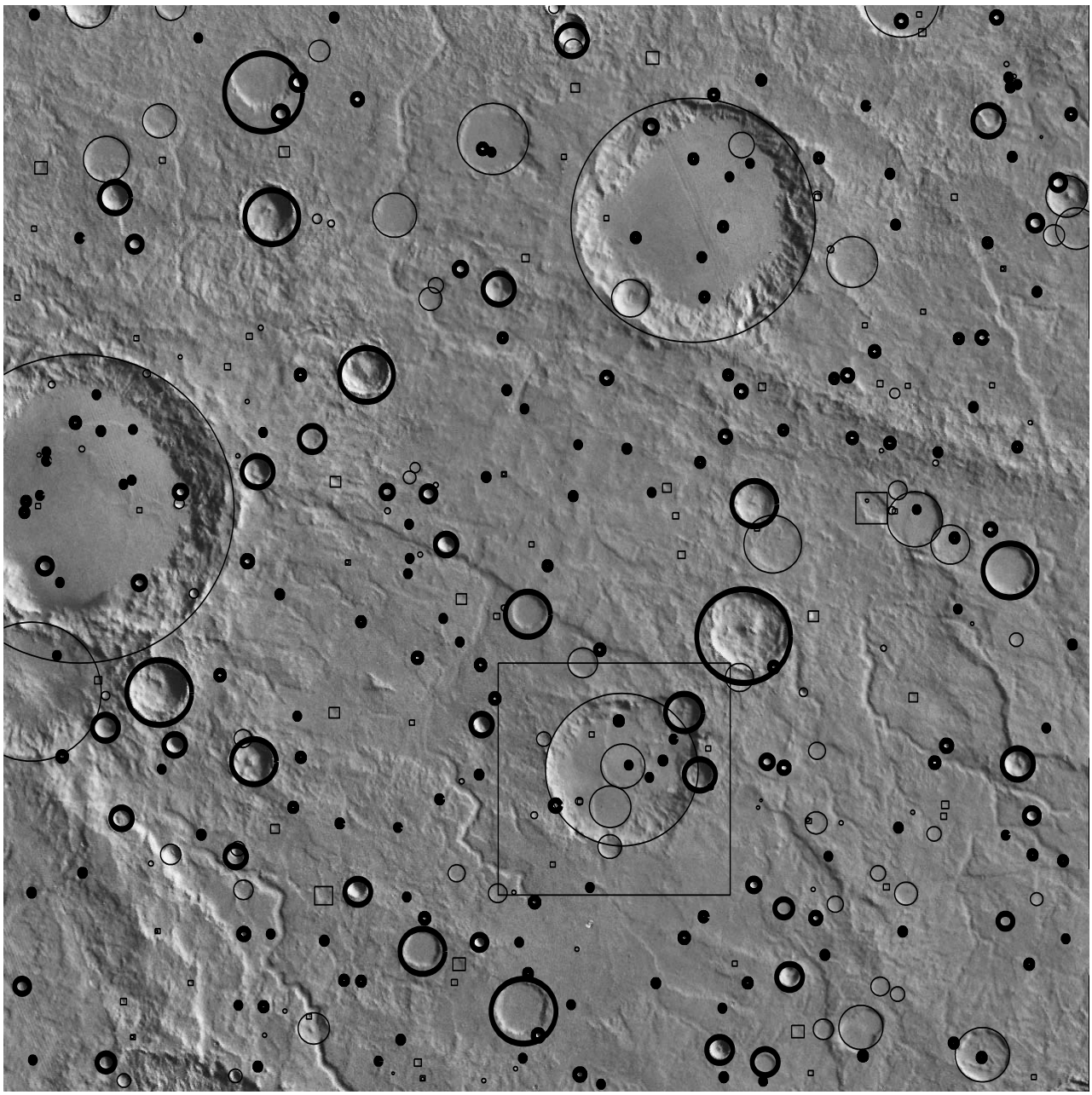

Figure 3: Detections obtained with the SVM-norm method on `mi00n0242`. Thick circles show detections judged to be correct by the automatic scoring procedure, while squares show false alarms. Missed craters are shown with thin circles. In this image there are 301 craters in the ground truth (with diameter $\geq$ 5 pixels). The SVM-norm detector succesfully finds and localizes 197 of these with 72 false alarms. This level of performance, although beneath that of a human labeler, offers a useful intelligent assistant since a human can quickly edit/correct the results.

Proceedings of the Seventh IEEE Workshop on Applications of Computer Vision (WACV/MOTION'05)
0-7695-2271-8/05 $ 20.00 IEEE