AAS 02-224

Optical Landmark Detection for Spacecraft Navigation

Yang Cheng*, Andrew E. Johnson*, Larry H. Matthies*, Clark F. Olson**

Optical landmark navigation using craters on the surface of a central body was first used operationally by the Near Earth Asteroid Rendezvous (NEAR) mission. It has proven to be a powerful data type for determining spacecraft orbits above the target for close flybys and low altitude orbiting. Tracking individual landmarks, which are small craters, enables orbit determination accuracies on the order of the camera resolution or several meters. This exceeds the accuracy that can be obtained from radiometric data alone. Currently, most of optical landmark navigation operations, such as crater detection, tracking, and matching etc, are done manually, which is extremely time consuming, tedious and sometime unmanageable. Because of the lengthily operation time and the deep-space communication delay, manual operation cannot meet the requirements of rapid and precise spacecraft maneuvers such as close orbiting, fast flybys and landing. Automating this operation can greatly improve navigation accuracy and efficiency and ultimately lead to an on-board autonomous navigation capability. In this paper, a new crater detection algorithm is suggested. Experimental studies show that this new algorithm can achieve sub-pixel accuracy in position, its detection rate is better than 90% and its false alarm rate is less than 5%. These good characteristics indicate that it is an ideal crater detection algorithm for spacecraft optical navigation.

INTRODUCTION

Craters are landforms commonly found on the surface of planets, satellites, asteroids, and other solar system bodies. A crater, in general, is a bowl shaped depression created by collision or volcanic activities. Because of their different geological ages and magnitudes of impact, craters may have a wide range of appearances. For instance, younger craters may have sharper and regular rims while aged craters might have very vague rims. Spatial densities of craters also form the primary basis for assessing the relative and absolute ages of geological units on planetary surfaces.

Besides their geological and astronomical interests, craters are ideal landmarks for spacecraft navigation [1-7]. Optical landmark navigation using craters on the surface of a central body was first used operationally by the Near Earth Asteroid Rendezvous (NEAR) mission. It has been shown to be a powerful data type for determining spacecraft orbits about the body for close flybys and low attitude orbiting. In the navigation filter, manually detected and identified craters were combined with Deep Space Network (DSN) radio metric tracking (Doppler and range) to estimate both orbital and asteroid physical parameters. The crater locations were also estimated in this process. The direct benefit of optical landmarks in the NEAR navigation solutions has

* Member of Mobility Systems Concept Development Section, Jet Propulsion Laboratory
** Professor of Computing and Software Systems, University of Washington, Bothell, WA

been enhanced navigation performance; i.e., increase orbital position accuracy in the 10 to 20 meter range, faster estimate convergence after maneuvers, and better solutions for dynamical parameters such as spacecraft non-gravitational accelerations and Eros gravity perturbations. Another benefit of using landmark tracking has been the rapid determination of poorly know physical parameters of Eros which affect navigation such as spin pole direction and spin state [4]. In NEAR mission, only a few dozens small, isolated and sharp craters out of thousand craters were used directly in orbit determination and the rest of them were used for contextual identification purpose.

Currently, most of optical landmark navigation operations, such as crater detection, identification, and matching etc, are done manually, which is extremely time consuming, tedious and sometime unmanageable. Usually, thousands of craters could be found on a small planetary body such as Eros. To identify and categorize them manually from thousands of images is a forbidding task. In addition, the manual operations lack consistency and large geometrical errors (position and shape) can be easily introduced. Furthermore, the long data processing time and the communication delay cannot satisfy many requirements of special spacecraft maneuvers such as fast flybys or small body landing. Finally, manual operation cannot be used for on-board autonomous navigation.

The navigational and scientific significance of craters have generated a lots of interests in autonomous crater detection algorithms. Because of their relative simple and regular geometries, the Hough transform constitutes the primary approach for crater detection. Olson [8][9] suggested a constrained Hough transform with a randomization technique yielding an algorithm with a worst case complexity of $O(n)$ where n is the number of edge pixels in the image. This algorithm is able detect relative simple geometric feature, such as straight lines and circular curves. However, when applying these techniques to conic curves with many degrees of freedom, such as an ellipse (5 degrees of freedom) a large number of trials are needed to achieve a useful detection ratio. The large number of trials makes this approach too expensive.

Leroy *et al* [10] suggested a tensor voting based crater detection algorithm. The edges are extracted by applying to the image to an edge detection algorithm. The tangent direction associated to each edge point in computed using the tensor voting technique. The curvature at each input site is estimated. All pixels considered belonging to the same crater is grouped together. Finally, an ellipse fitting algorithm is applied to delineated the crater. However, this technique only had a very limited success. Its false alarm ratio and omission ratio are still too large for spacecraft navigation purpose.

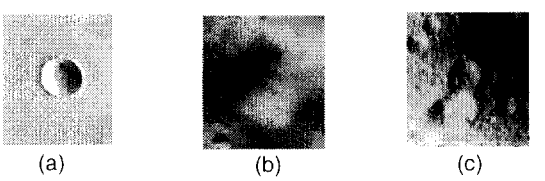
Mark Burl et al [11][12] have suggested an automated volcano (crater) detection algorithm, which classifies the candidate regions by the correlation with synthesized feature templates. Preliminary test showed performance comparable to trained human observers. Since this technique was designed for catalog inventory purpose, the geometrical property such as shape and position of the craters has not been well addressed. In addition, the false alarm rate is still too high [11]. For example, when the

detection rate is about 80 and 90%, the false alarm rate is about 100% and 1000% respectively. Such a high false alarm rate really cannot be used for any navigation application. Furthermore, the method of Burl et al. cannot accommodate elliptical craters that are often generated by oblique viewing directions and the 3-D shape of small bodies.

For the navigation application, it is highly desirable that the crater detection algorithm achieves a high positional accuracy (sub-pixel), high tolerance to large variations in crater appearance (e.g. lighting and viewing angle, image types and crater types), real time operation and a low false alarm ratio. In following section, such an algorithm is presented.

2 CRATER DETECTION

Because of the different geological ages and magnitudes of collision impacts, craters may have very wide range of variation. For example, some might have very fuzzy rim (Fig 1 b) and others might have broken rims (Fig 1 c) etc. To make an algorithm work well across the wide range of crater appearances is not easy. However, a typical crater in an image has an elliptical rim and a bright to dark shading pattern, which is dictated by the lighting azimuth and elevation as well as its own topography. These distinguishing characteristics were used extensively in the crater detection here.



(a) (b) (c)

Figure 1: Craters appear very differently from image to image.

The crater detection algorithm is broken down into six steps:

1. Edge detection: This step detects all edges including rim edges from a given image;
2. Crater anchor point detection: This step selects all probable crater edges from the edge database.
3. Rim edge grouping: This step groups together edges considered belonging to the same crater.
4. Ellipse fitting: This step fits grouped crater edges into an ellipse.
5. Ellipse refinement: This step adjusts the detected crater's geometry directly on the image domain to remove any error introduced in the edge detection and the ellipse fitting.
6. Crater confidence evaluation: This step evaluates the quality of each detected crater.

2.1 Edge Detection

The edges of an image are extracted by applying the Canny edge detection algorithm [14]. In order to extract both sharp and fuzzy rims, two rounds of edge detection are executed. In the first execution, a smaller kernel (< 5) is used, which aims to detect small and sharp edges. In the second round, a large kernel (~9) is used to detect large and less sharp edges. Experimental study shows this strategy is very effective when image quality

is poor such as NEAR imagery. The image gradient (g_x, g_y) of each edge pixel is extracted. Both data sets are stored separately for the future usage.

2.2 Rim Edge Grouping

2.2.1 Crater Topography Constrains

After the edge detection algorithm detects indiscriminately all edges from an image, we have to pick up those belonging to rims. Here some topographic constrains are used for the crater edge selection process:

- The angle between the lighting source direction and gradient vector of crater rim should be less than 90 degrees (Fig. 2 a). In order to reduce false alarm ratio, we only select the edges whose gradient vectors are less the 50 degrees off the lighting source direction as

$$(g_x S_x + g_y S_y)/(g_x^2 + g_y^2)^{1/2} > 0.64 \qquad (1)$$

Where g_x and g_y are the gradients of the image. S_x and S_y are the lighting direction on the image.

- A rim edge should be a convex curve (Fig. 2 b). The shaded side of rim should be facing the lighting source and lit side should be facing the opposite direction.
- Inside a crater, the image intensity profile along lighting direction should be an monotonously decreasing function.

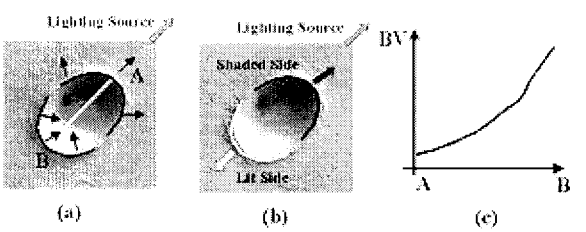


Figure 2: Crater topography constrains help in crater edge selection.

2.2.2 Convex Analysis for Long Edges

If a given edge (E) is part of a crater rim, we would like to know whether it lies on lit or shaded side of the crater. For a large crater, because its rim edge is preserved fairly well, it can be determined just based on its convex orientation. In order to do so, we have to detect two anchor points, the furthest point (p_f) and closest point (p_c) on the edge to the lighting resource (Fig. 3. a):

$$p = \begin{cases} p_c & if\,(ax_c + by_c + c = \min) \\ p_f & if\,(ax_f + by_f + c = \max) \end{cases} \qquad (2)$$

Where (a, b, c) are the coefficients of a baseline, which is perpendicular to lighting direction and lies outside of the images.

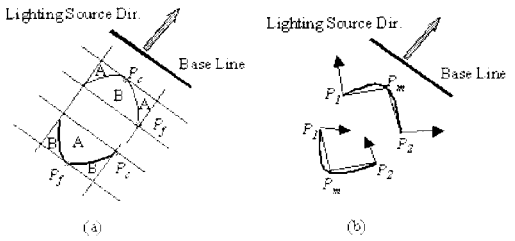


Figure 3: Convex analysis will figure out an edge belonging to either shaded or lit side of a crater.

Two straight lines passing through the anchor points and perpendicular to lighting direction and another two straight line passing the two end points of the edge and parallel to the lighting direction form a rectangular. This rectangular is divided into two regions by the edge E and the area ratio of the two regions indicates the convex orientation as

$$E = \begin{cases} lit \quad side \quad if(A/B > 1.0) \\ Shade \quad Side \quad otherwise \end{cases} \tag{3}$$

2.2.3 Convex Analysis for Short Edges

Because of relative large kernel of edge detection operator, the shapes of smaller craters (< 20 pixels) are note well preserved. In this case, the method in 2.2.2 becomes ineffective. In this case their convex orientation here is detected by their image gradients.

Assume the middle point of an edge is P_m and two end points are P_1 and P_2 and their gradients are G_1 and G_2. Therefore. The edge is on lit side of a crater if

$$\sum_{i=1}^{2} \frac{(P_m - P_i) \bullet G_i}{\| (P_m - P_i) \| \times \| G_i \|} > 0.0 \tag{4}$$

Otherwise, it is on shaded side (Fig. 3 b).

2.2.4 Edge Grouping

All candidate edges are labeled as either lit or shaded side of craters by the methods in section 2.2.2 and 2.2.3. Now we will group them up into lit and shaded pairs. Several constrains are used to perform this task:

- The ratio between the lit and shaded edge lengths should be close to 1 as: $1/t < (l_l / l_s) < t$; where l_l and l_s are the length of lit and shaded side edges and t is the threshold greater than 1.0;
- The image intensity profile between the two edges should be a monotonously decreasing function. We fit the image intensity profile by a straight line and its slope is used to test its decrement.
- The distance between the edges (the distance between two anchor points) should not be too close or too far. In this algorithm, the distance must be less than twice the length of the longer edge and larger than the half-length of the shorter edge.

In order to speed up the edge grouping process, we combined of vector and raster data search scheme. All marked edges' IDs are mapped to an integer 2-D array. For a given edge, the search area is a 30 degrees conic area on the opposite direction of the edge convex orientation. The size of search area is dependent on the desired crater size specified by users and the length of current edge. Any edge found in the search area will be tested using the three constraints listed above. If they pass the test, the pair of edges will be used to fit an ellipse.

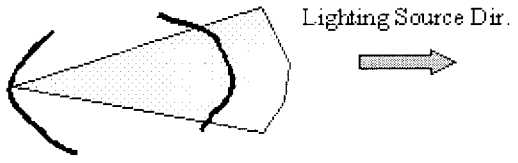


Figure 4: The edge grouping algorithm searches an edge on other side of a crater in a 30 degree conic area.

2.3 Ellipse Fitting

A generic conic representation of an ellipse is

$$F(A,X) = a^T X = a_1 x^2 + a_2 xy + a_3 y^2 + a_4 x + a_5 y + a_6 = 0 \tag{5}$$

$$A = [a_1, a_2, a_3, a_4, a_5, a_6]^T \text{ and } \boldsymbol{X} = [\, x^2, xy, y^2, x, y, 1]^T$$

A conic polynomial could represent an ellipse, hyperbole or parabola. An equality constraint $4a_1a_3 - a_2^2 = 1$ will force $F(\boldsymbol{A}, \boldsymbol{X})$ to be an ellipse for any case. We rewrite this in a matrix form $a^t ca = 1$, where c is a 6 by 6 sparse matrix, in which $c_{13} = c_{31} = -c_{11} = 1$ and all other entries are zero.

Once the candidate rim edges are selected, the ellipse should be satisfy

$$E = \| a^t x \|^2 = \min \;\&\quad a^t ca = 1 \tag{6}$$

We have implemented the Least-Median-Square regression [need reference] and iterative reweighting techniques [12] for ellipse fitting and found that the later one has better performance in term of accuracy and convergence speed. In this case, the maximum likelihood estimators is

$$\begin{aligned} E &= \sum w_i \, \| a^T x \| = \min \;\&\quad a^t ca = 1 \\ w_i &= 1/(1 + d_i^3) \end{aligned} \tag{7}$$

Where d_i is the distance of an edge pixel to the computed ellipse.

2.4 Crater Refinement

The crater detection algorithm described in the previous sections provides relatively successful crater detection. However, it cannot achieve sub-pixel positional accuracy because it relies solely on edge pixels for ellipse fitting. It is well known that edge detection often introduces unwanted artifacts. These artifacts will introduce errors into

the geometry (position and shape) of detected ellipse. The crater refinement algorithm removes the artifacts.

Let's represent an ellipse by five parameters: its center x_0, y_0, major axis a, minor axis b and orientation t_0. The equation of the ellipse under this parameterization is

$$\begin{aligned} x(t) &= a\cos t\cos t_0 - b\sin t\sin t_0 + x_0 \\ y(t) &= a\cos t\sin t_0 + b\sin t\cos t_0 + y_0 \end{aligned} \tag{8}$$

If the lighting direction is given, the image of the crater should satisfy a merit function $\boldsymbol{M}$

$$M = \frac{\oint \| (g \bullet n)(s \bullet n) \| \, dt}{\oint (x_t^2 + y_t^2) dt} = \max \tag{9}$$

Where $\boldsymbol{g}$ is the image gradients at $(\boldsymbol{x}, \boldsymbol{y})$, $\boldsymbol{s}$ is the lighting direction vector and $\boldsymbol{n}$ is the customized unit normal of crater rim as.

$$\begin{aligned} n(n_x, n_y) &= \begin{cases} (x_{tt}k_1 - x_t k_2, & y_{tt}k_1 - y_t k_2) \quad if\,(n \bullet s) > 0 \\ (-x_{tt}k_1 + x_t k_2, & -y_{tt}k_1 + y_t k_2) \quad otherwise \end{cases} \\ x_t &= -a\sin t\cos t_0 - b\cos t\sin t_0 \\ y_t &= -a\sin t\sin t_0 + b\cos t\cos t_0 \\ x_{tt} &= -a\cos t\cos t_0 + b\sin t\sin t_0 \\ y_{tt} &= -a\cos t\sin t_0 - b\sin t\cos t_0 \\ k_1 &= \frac{r^2}{ab}, \\ k_2 &= \frac{(a^2 - b^2)\sin t\cos t}{abr^2} \end{aligned} \tag{10}$$

By adjusting the five parameters to maximize the merit function (9) will eventually remove the error and lead to locking on the crater rim precisely. At here, we assume the image gradient direction and rim normal directions are very close each other. $(\boldsymbol{s} \bullet \boldsymbol{n})$ in Equation (9) is a weight factor, which reaches its maximum(1) when $\boldsymbol{s} = \boldsymbol{n}$ and its minimum (0) when $\boldsymbol{s} \perp \boldsymbol{n}$.

In order to do so, we rewrite Equation (9) into a discrete form. In this form the denominator is omitted because it is a constant here.

$$M = \sum \| (g \bullet n)(s \bullet n) \| = \sum (n_x^2 g_x s_x + n_x n_y (g_y s_x + g_x s_y) + n_y^2 g_y s_y) \tag{12}$$

To perform the maximization, we use a multidimensional iterative nonlinear minimization algorithm based on conjugate gradient [15]. This algorithm requires computation of the partial derivatives of $\boldsymbol{M}$ with respect to the five parameters in Equation (12). These are relative straightforward to compute using the chain rule. For example,

$$m_a = \sum [2n_x g_x s_x n_{xa} + n_x^2 g_{xa} s_x + (n_y n_{xa} + n_x n_{ya})(g_y s_x + g_x s_y)$$
$$+ n_x n_y (g_{ya} s_x + g_{xa} s_y) + 2n_y g_x s_x n_{ya} + n_y^2 g_{ya} s_y]$$

$$\frac{\partial k_1}{\partial a} = \frac{\sin^2 t}{bd} - \frac{d}{a^2 b} \qquad \frac{\partial k_2}{\partial a} = \frac{2}{bd} - (a^2 - b^2)(\frac{\sin^2 t}{bd^3} + \frac{1}{a^2 bd})$$
$$\frac{\partial k_1}{\partial b} = \frac{\cos^2 t}{ad} - \frac{d}{ab^2} \qquad \frac{\partial k_2}{\partial b} = -\frac{2}{ad} - (a^2 - b^2)(\frac{\cos^2 t}{ad^3} + \frac{1}{ab^2 d})$$

$$x_{ta} = -\sin t \cos t_0$$
$$x_{tb} = -\cos t \sin t_0$$
$$x_{tt_0} = a \sin t \sin t_0 - b \cos t \cos t_0$$

$$y_{ta} = -\sin t \sin t_0$$
$$y_{tb} = \cos t \cos t_0$$
$$y_{tt_0} = -a \sin t \cos t_0 - b \cos t \sin t_0$$
$$x_{tta} = -\cos t \cos t$$
$$x_{ttb} = \sin t \sin t_0$$
$$x_{tt_0} = a \cos t \sin t_0 + b \sin t \cos t_0$$
$$y_{tta} = -\cos t \sin t_0$$
$$y_{ttb} = -\sin t \cos t$$
$$y_{tt_0} = -a \cos t \cos t_0 + b \sin t \sin t_0$$

$$g_{xa} = g_{xx} x_a + g_{xy} y_a$$
$$g_{ya} = g_{yy} y_a + g_{xy} x_a$$
$$g_{xb} = g_{xx} x_b + g_{xy} y_b$$
$$g_{yb} = g_{yy} y_b + g_{xy} x_b$$
$$y_a = \cos t \sin t_0$$

$$n_{xa} = -\sin t * \sin t_{0/} / r - n_x r_a / r^2 \quad (n * s) > 0$$
$$n_{xa} = \sin t * \sin t_{0/} / r - n_x r_a / r^2 \quad otherwise$$
$$r_a = a * \sin^2 t / r^2$$

In order to improve the convergence and avoid local maximum, we decompose the optimization problem into two groups. First we will try to find the best location of crater by maximizing equation (12) with respect to x_0 and y_0. Then we will try to determine the best ellipse shape by maximizing equation (12) with respect to a, b and t_0. This interlacing approach is repeated several times until a threshold is reached. When we are dealing with large and regular object at closer range, such as Moon, Mars, the lighting

direction on an image is fairly consistent. However, when we are working on the small and irregular shape object, such as Phobos, Eros etc. the lighting direction in an image varies due to the local surface normal and viewing direction. In order to precisely locking on the craters, the local lighting direction will be determined as

$$Angle = \arctan\left(\frac{\sum g_x}{\sum g_y}\right)$$

A new lighting direction is recomputed between two iterations.

An experimental study has shown that the crater refinement algorithm can tolerate up to 30% translation, 10% scale and 30 degrees of rotation error. These bounds easily contain the largest error generated by the crater detection algorithm from the previous section (Fig. 5).

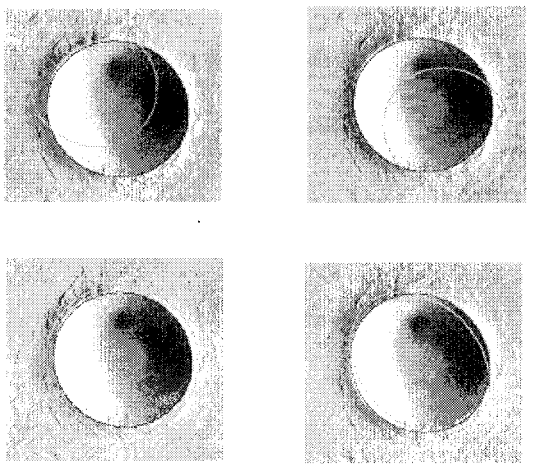


Figure 5: The crater refinement algorithm can tolerant large errors. The green circles are the initial craters and the red circles are the final craters.

2.5 Crater Confidence Evaluation

Our confidence metric is derived from Equation (9). N points are evenly selected on both sides of a crater. No point is selected at the lit and shaded transition area (30 degrees on each side) because of large ambiguity in image gradients around those areas. The confidence of a crater is computed by

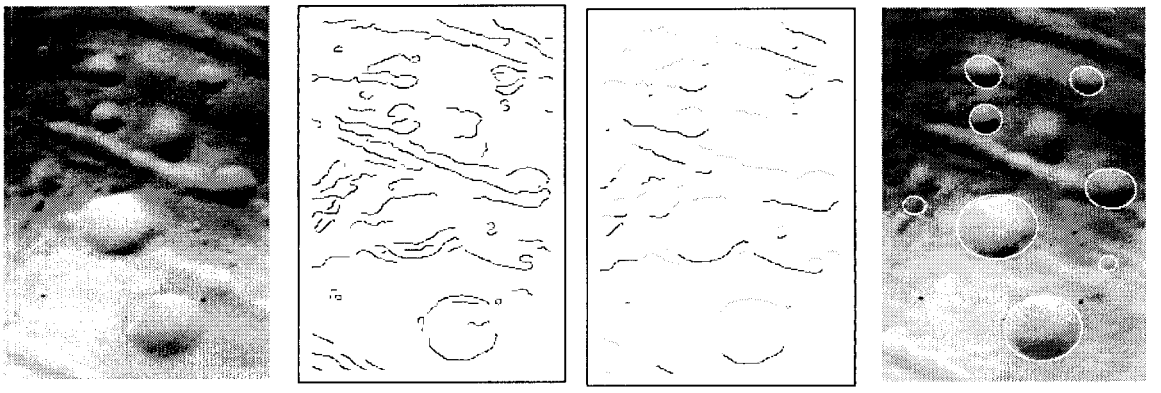
$$C = \frac{\sum_{i=1}^{n} K_i}{N}$$

$$K_i = \begin{cases} 1 & if(n_i \bullet g_i) > T \\ 0 & otherwise \end{cases} \tag{12}$$

Where n_i is the customized normal of ellipse (equation (10)) and g_i is the gradients at point i.

If a crater's confidence is greater than a threshold, it will be added to the crater database and the edges associated with this crater will be removed from the edge database and the 2-D edge ID array.

The crater detection algorithm is composed of the steps described in sections 2.1 to 2.5. Fig. 6 shows some intermediate and final results of this algorithm.



(a) (b) (c) (d)

Figure 6: Some intermediate and final results of the crater detection algorithm. (a) The original image; (b) The edge image by Canny edge detection; (c) The edge image after edge filtering and convex analysis. The black edges are the edge on shaded side and gray edges are the edges on lit sides; (d) The detected craters.

3 CRATER DETECTION VALIDATION

The following sections describe results on images collected in the lab for accuracy assessment and images from various deep space missions.

3.1 The Geometric Accuracy Assessment

Because a benchmark does not exist for real imagery, the accuracy assessment was conducted in a carefully arranged laboratory sitting. Six craters of difference sizes were created on a sand bed and eight fiducial marks were laid around the craters. A total 23 images were taken from different angles and distances. The eight fiducial marks are extracted manually from every image and they are used to construct homography transforms between any two images:

$$X = \frac{m_1 x + m_2 y + m_3}{m_7 x + m_8 y + 1}, Y = \frac{m_4 x + m_5 y + m_6}{m_7 x + m_8 y + 1}$$

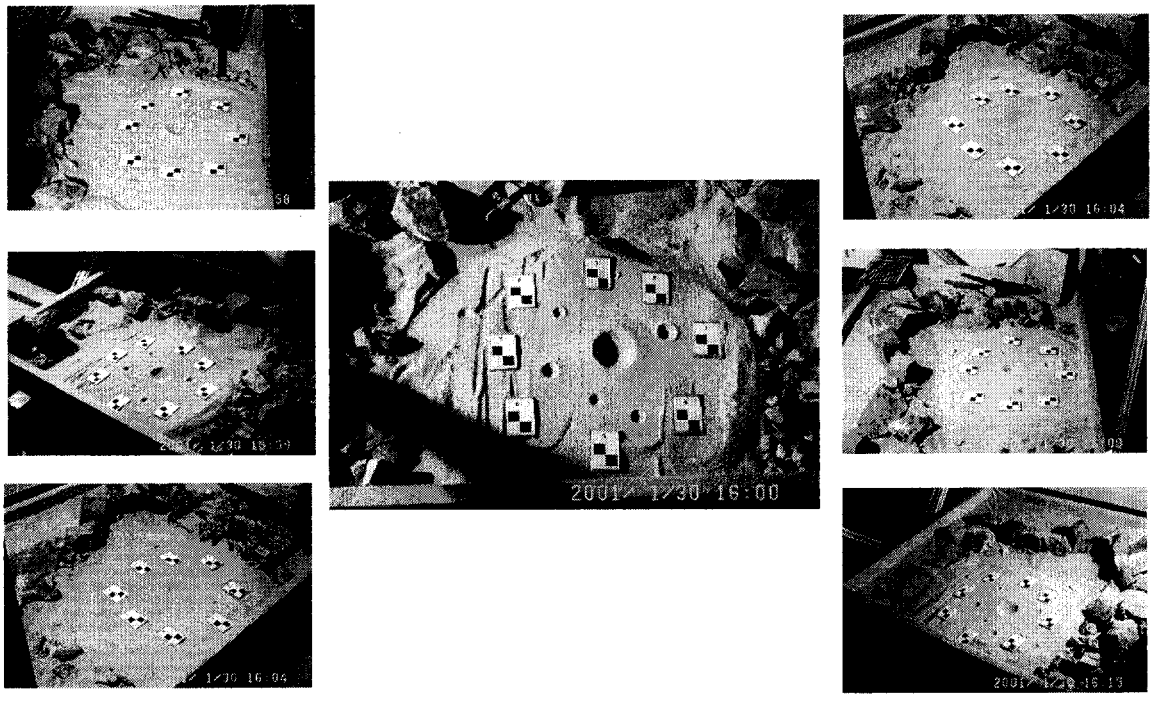


Figure 7: Total 23 images were taken in different angles and distances. The center image is the base image.

In order to remove the scaling uncertainty, an image with finest resolution is chosen as the base image (Fig. 7 and Table 1). The craters detected in the base image, using our detection algorithm, are transformed to the other images using the homography transforms. These "truth" craters are then compared to the craters detected locally. Four statistics are extracted from this study: positional error (*dx*, *dy*) and geometrical (major and minor axis) errors (*da*, *db*) (Table 2 and 3).

Table1: The six craters were detected from the base image. T_0 is in degree and the rest are in pixel.

ID	x	y	A	b	T_0
1	209.96	120.50	5.6524	5.4096	238.18
2	308.63	138.36	8.9467	7.8952	201.12
3	260.88	154.41	20.496	18.363	201.30
4	202.12	173.51	8.2422	7.7153	204.31
5	244.83	199.97	7.0682	6.8220	186.53
6	282.48	215.32	7.1588	6.3692	214.11

Table 2: The positional error statistics.

ID	E(dx)	std(dx)	E(dy)	Std(dy)
1	0.2421	0.2410	0.1119	0.2586
2	-0.0951	0.1442	0.0455	0.1420
3	0.0034	0.2438	0.1278	0.2153
4	-0.0165	0.1299	-0.0499	0.1719
5	-0.1972	0.1742	-0.2184	0.2193
6	0.0629	0.2010	-0.0359	0.2341
Total	0.00	0.2328	0.0	0.2368

Table 3: The geometrical error statistics

ID	E(da)	Std(da)	E(db)	Std(db)
1	-0.0846	0.3494	0.1942	0.3943
2	-0.1178	0.1924	0.2329	0.3346
3	-0.3254	0.4041	0.3102	0.3048
4	-0.1185	0.2061	0.3079	0.2876
5	-0.0258	0.2845	0.0727	0.4206
6	0.0125	0.2589	0.3137	0.5970
Total	-0.1101	0.3066	0.2412	0.4073

The RMS error of position is less than 0.3 pixel and the RMS error for the shape is less than 0.5 pixel. It indicates that this crater detection algorithm is able to achieve sub-pixel accuracy. In addition, the table 2 and 3 indicate that RMS error for each crater is not significantly different between different sizes of crater. We also have conducted some study on the effect of the lighting and viewing direction to the crater error and found out that their contribution to geometric errors was negligible.

3.2 The detection rate and false alarm rate

The analysis of the omission and emission errors was conducted on real deep space mission imagery including MGS, NEAR, Voyager etc. It performs well on all the test images. Again, because benchmarks did not exist, the craters missed (omission error) or misidentified (emission error) by this algorithm were manually identified. For the missed crater category, we only accounted craters missed by our algorithm but still having navigational value. The craters, which were too large, too small, broken, or partially occluded, were not included. Most misidentified cases happen when multiple craters overlap or are close to shadow edges. Table 4 shows the results on four images (Fig. 8).

Table 4: Some statistics of the crater detection algorithm

Img	**R**	**C**	**T**	**N**	**M**	**F**
1	470	830	92.91	163	6	7
2	691	640	17.25	23	1	1
3	518	558	70.24	119	8	4
4	500	1280	60.19	100	2	0

Where **R** and **C** is the number of rows and columns in the input image; **T** is the wall clock execution time (s) on a SUN Ultra 10 workstation; **N** is the number of detected craters, **M** is the number of missed craters, and **F** is the number of misidentified craters.

This experimental study indicates that the detection rate of this algorithm is better than 90% and the false alarm rate is less than 5%, which are adequate for spacecraft navigation.

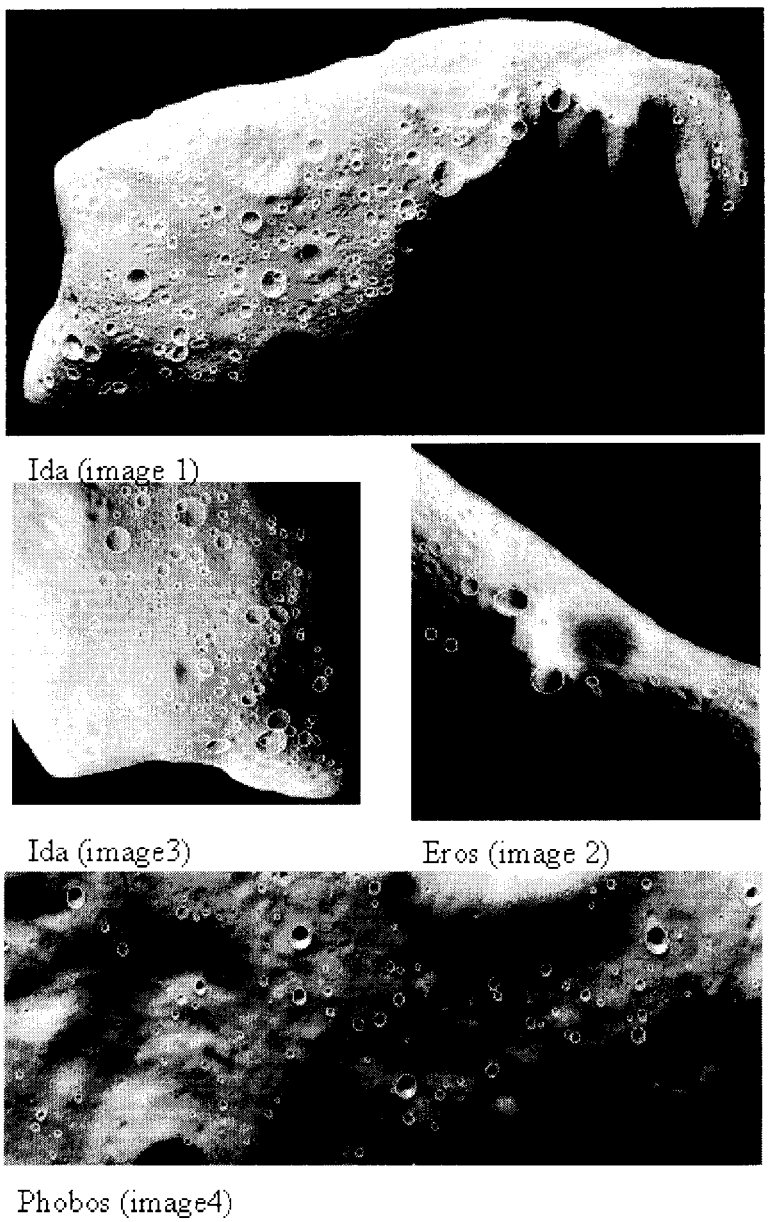


Figure 8: Some results of the crater detection algorithm.

4 3-D CRATER MATCHING

In order to determine the position of the spacecraft with respect to the asteroid, we use geometrical recognition techniques that perform matching between the craters that have been extracted from an image of the asteroid and a crater database containing the 3D locations of the craters on the asteroid surface. This problem is closely related to pose estimation of 3D objects from 2D image data. The basic algorithm that we use is an efficient method for pose estimation from points in such data [19]. Each crater is treated as an attributed point corresponding to the center of
the crater, where the attributes are the radius and orientation of the crater. The efficiency of the basic methodology is improved by two means. First, the crater attributes are used to remove matches that are incompatible. Second, an initial estimate of the spacecraft position is used to filter matches that are not feasible. The following subsections describe this method in more detail.

4.1 Point Matching

Given a set of 3D objects points, the pose of the object can be determined from a 2D image of the points, with most of the points missing and also with many spurious points using a technique based on the Hough transform.
The basic idea is to compute the pose(s) that bring small sets of object points into alignment with image points. If the sets contain three points each, then each match yields two poses that bring the points into alignment under a weak-perspective model [18] (and up to four under full perspective [16]). Therefore, the sets of matches map into points in the six-dimensional pose space. After some number of matches are examined, the correct pose will correspond to a cluster of points in the pose space, since each correct match should yield at least one pose that is close to the correct pose.

We use an efficient implementation of this idea that examines several trials of somewhat simpler problems [19]. In each trial, we select two image craters and two database craters that must match for the trial to succeed. Since we don't know which craters will match in advance, not every trial will succeed. A randomization technique is used to ensure that enough trials are examined such that the probability of them all failing is very low.

For each trial, we consider each additional match between a single image crater and a database crater. Since each such match gives us three overall (with the two were previously selected for the entire trial), we can determine the poses that bring the crater centers into alignment and determine if there is a large enough cluster in the pose space.

4.2 Crater Filtering

Given an initial pair of matches, our algorithm considers each possible additional match between an image crater and a database crater, yielding triples of matches. Not all such triples are viable. Prior to computing the poses that bring all three crater centers into

alignment, we filter pairs of matches using visibility constraints and using the ratio of the crater radii. After computing the poses, we filter further matches by checking the crater orientation and the ratio of the major and minor axes lengths in the image.

4.2.1 Crater pairs

For each crater detected in the image, the major axis of the ellipse detected corresponds to a cross-section of the crater. The ratio of these axes in the image must be the same as ratio of the crater radii in the database (modulo image noise and detection error). We eliminate pairs of craters if the ratio is not within 50% of the correct ratio from the database.

We set this threshold to filter only those craters that are clearly wrong, since other filters will also eliminate many cases. An experiment indicates that approximately 12\% of the crater pairs are filtered with this test.

In addition, each pair of craters must be mutually visible from some viewpoint. We filter any pair of craters that has more than a 60 degree difference in orientation. While this constraint will filter a few more crater pairs than necessary, those that are filtered are less likely to yield good results, since at least one crater will be considerably foreshortened in the image. Approximately 26% of the crater pairs are filtered with this test.

4.2.2 Crater triples

If all three pairs of matches in the triple pass the previous test, we compute the poses that bring the crater centers into alignment using the method of Huttenlocher and Ullman [18]. We can do further pruning on these poses. For example, if the pose requires that one or more of the craters is on the wrong side of the asteroid to be seen. We check to see whether the pose specifies that one of craters has an orientation greater than 75 degrees away from the camera. If so, then the pose is filtered, since the crater would be either on the wrong side of the asteroid or extremely foreshortened. This test filters 32% of the remaining poses.

The estimated pose also tells us what size the crater should be in the image, what the ratio of the major and minor axis lengths should be, and the orientation of the crater in the image.

If the size or the ratio of axis lengths is not within 50% of the correct size, we filter the pose. Finally, if predicted orientation of the crater from the axis lengths in the image is not within 40 degrees of the orientation predicted by the pose, we also filter the pose. These tests remove 31% and 20% of the poses, respectively.

4.3 Pose filtering

If we have an initial estimate of the spacecraft position and an error covariance matrix, the pose estimation process can be made much more efficient by pruning the matches that are not consistent with the position estimate.

We represent the spacecraft orientation using a quaternion q and the position with a 3-vector t, so the overall spacecraft position is represented by 7 values (4 for the quaternion and 3 for the position), although only 6 are independent. Given the error covariance matrix, with values c_{ij} , for $1 \leq i,j \leq 7$, we can use covariance propagation methods to project the error covariance into an ellipse in the image space centered at the position given by the projection of the database crater according to the estimated spacecraft position.

Let $\boldsymbol{p}$ be the vector [0 p], so that we can use quaternion multiplication to rotate the vector. The equation that takes points from the asteroid frame of reference to the spacecraft camera frame of reference is:

$$p' = qpq^* + t \tag{14}$$

If we view the point from the spacecraft camera with focal length f, the image coordinates are:

$$\begin{bmatrix} i_x \\ i_y \end{bmatrix} = \begin{bmatrix} \dfrac{fp'_x}{p'_z} \\ \dfrac{fp'_y}{p'_z} \end{bmatrix} \tag{15}$$

The position covariance is propagated into the image coordinates through linearization by taking the partial derivatives of this equation with respect to the pose parameters (i.e. the Jacobian).

Now, the error covariance matrix in the image space is given by $C_i = JC_pJ^T$ where C_p is the covariance matrix in the pose space. Now, we want to decide if an image crater is close enough to the estimated position of a database crater, so we calculate the error vector:

$$e = p_i - qp_dq^* + t \tag{16}$$

where p_i is the center of the crater in the image and p_d is the center of the crater in the database. If the errors yield a multi-variate normal distribution around the estimated point in the image, then we get a chi-squared test statistic (with 1 degree of freedom) using:

$$\chi^2 = e^T C_i^{-1} e \tag{17}$$

If this test statistic is above a threshold (we use 3.841), then the crater is eliminated from consideration.

This filtering procedure is used every time we consider a match between a particular image crater and a database crater. Over the course of the algorithm, matches are often considered several times. We further improve the efficiency by maintaining a look-up table for matches that have been previously considered, so that the computations need not be performed again. With a diagonal covariance matrix, with values of

0.2 for the position variables and 0.0001 for the quaternion values, this test eliminates 98.5\% of the crater matches from consideration.

4.4 Clustering

In this method, we need to be able to detect whether a cluster of points is present in the pose space for each trial. We perform this operation using a simple accumulator method, where there pose space is divided into bins and each possible pose votes for bins in this space.

First, note that since each trial fixes two matches between craters in the image and in the database, we have four constraints on the six-dimensional pose space (two for each match). This means that the poses considered in each trial are constrained to lie on a two-dimensional manifold of the six-dimensional pose space. We parameterize this manifold using the scale of the projection (which is linearly related to the distance of the spacecraft from the asteroid) and the rotation angle around the center of the two database craters used in the initial match for the trial.

We discretize the two parameters coarsely. Given these values, each pose that passes through the filters described above votes for nine bins in the pose space - the bin that the pose falls into and its 8 neighbors (to account for noise). A vote corresponds to incrementing the appropriate accumulator.

Once each pose in some trial has been considered, we look through the accumulator to determine if any bin has accumulated sufficient votes to correspond to the spacecraft position. In practice, this is performed by maintaining the score of the best position found so far and comparing against this score. When a high scoring bin is found in the accumulator, a new score (bounded by the count in the accumulator) is computed to more accurately reflect the quality of the pose. The new score is computed by determining how well each of the crater matches that contributed to poses in the bin is brought into alignment. Each match is given a score between 0 and 1 that encompasses position, scale, and orientation information. The sum of the scores is used as the score for the bin. There is also a question as to which pose should be used to test each bin with a large score. To ensure that a good pose is not missed, we test each of the poses that fell into the bin and keep the pose with the best score.

We must now decide how many random initial pairs of image craters should be examined by the algorithm. We can derive this value using some assumptions about the number of database craters detected in the image [19].
If we assume that some minimum number c of the database craters are detected (for example, three), then we can estimate the probability that none of k trials will examine a pair that are from the database:

$$p \approx \left(1 - \left(\frac{c}{n_i}\right)^2\right)^k \tag{18}$$

where n_i is the number of image craters detected.

Setting p to be below some small threshold σ(for example, 0.0001), we can estimate the number of trials necessary as:

$$k \approx \left(\frac{n_i}{c}\right)^2 \log\frac{1}{\sigma} \tag{19}$$

For a problem with 18 image craters and 955 database craters, this algorithm requires approximately 2 seconds on a 333 MHz Sun Ultra 10. This example is illustrated in Fig. 9.

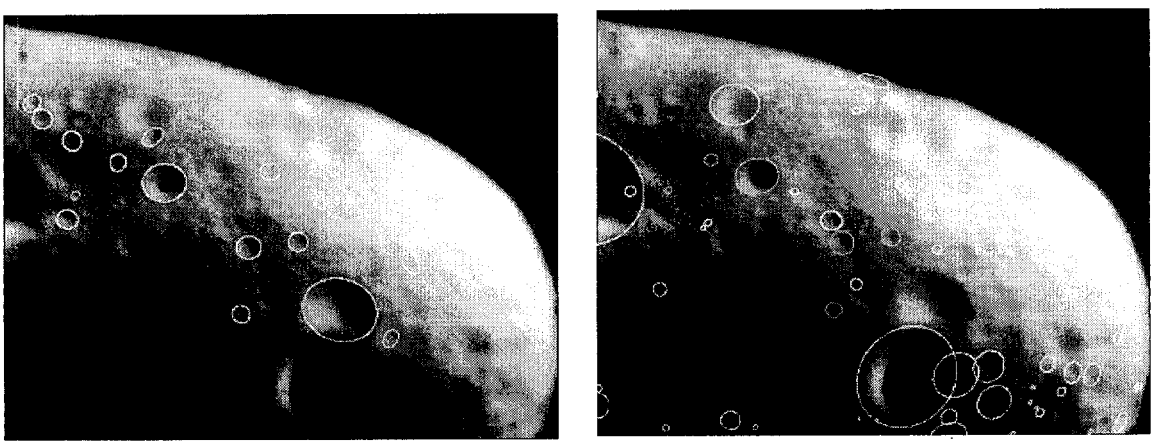


Figure 9: Example result using NEAR imagery of the Eros asteroid. (a) Crater detected. (b) Pose of asteroid computed after crater matching. Matched craters are shown in green. Unmatched craters are shown in yellow.

5 CONCLUSIONS

In this paper, a crater detection and crater match algorithms have been presented. By fusing multiple types of information, this algorithm is able to extract craters at very high detection rate (> 90%) and low false alarm rate (<%5). With the crater refinement algorithm, the sub pixel accuracy in term of position and shape can be achieved. The crater match algorithm presented in section 4 is able to match crater detected from an image to a crater database containing the 3D locations of the craters on the asteroid surface and compute the spacecraft pose. With these good properties, these algorithms are excellent tools for spacecraft optical navigation for both onboard and ground operation.

Additional work is needed in two directions. First, an autonomous method to construct 3D crater database is needed. When a large number of images have been taken around a central body, it is possible to reconstruct the craters' 3D positions and shapes. In order to do so, the same crater from different images has to be positively identified. Some techniques, such as affined geometry, conic invariance etc can be used to perform this task. Once all craters are positively matched across images, the crater positions and shapes can be determined by triangulations. The second direction of this research will be to fuse this algorithm to real navigation software and apply to a real mission such as DAWN and messenger mission.

ACKNOWLEDGEMENTS

The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

REFERENCES

1. J. E. Riedel, et al, "An Autonomous Optical Navigation and Control System for Interplanetary Exploration Missions", *2nd International conference on Low Cost Planetary Missions.*
2. L. A. Caugahuala, T.R. Drain, and P. D. Burkhart, " ARTSN: Prototype for an Automated Real-time Spacecraft Navigation System" *AIAA-98-4185.*
3. Marina Kolesnik and Gregory Baratoff, "3-D Interpretation of Sewer Circular Structures", *Proceedings of the 2000 IEEE ICRA.*
4. B. Williams, J.K. et al. "Navigation Results for NASA's Near Earth Asteroid Rendezvous Mission", *AIAA/AAS Astrodynamics Specialists Conference.*
5. J. Kawaguchi et al, "Autonomous optical guidance and navigation strategy around a small body", *Journal of Guidance, Control, and Dynamics*, Vol. 20, No.
6. Douglas Bernard et al "Spacecraft autonomy flight experience: the DS1 remote agent experiment", *AIAA-99-4512*
7. Clark Olson, "Constrained Hough Transforms for Curve Detection", *Computer Vision and Image Understanding*, Vol. 73, No. 3.
8. Clark Olson, "Improving the generalized Hough transformation through imperfect grouping", *Image and Vision Computing* 16, 1998.
9. Michael Burl et al, "Learning to recognize volcanoes on Venus", *Machine Learning*, 30, 1998.
10. Michael Burl et al, "Automating the hunt for volcanoes on Venus", 1063-6919/94 IEEE.
11. Marina Kolesnik and Gregory Baratoff, "3-D interpretation of sewer circular structures", *Proceedings of 2000 IEEE ICRA.*
12. Andrew Fitzgibbon et al, "Direct least squares fitting of ellipses", www.dai.ed.ac.uk/groups/mvu/ellipse-demo.html.
13. John Canny, "A computational approach to edge detection," *IEEE PAMI*, 8 (6) 1986.
14. W. Press et al, "*Numerical Recipes in C.*" Cambridge University Press, 1992.
15. M.A. Fischler, and R.C. Bolles. "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography", *Communication of the ACM*, 24:381-394, June 1981.
16. R.M. Haralick, "Propagating covariance in computer vision", *In Proceedings of the 12th International Conference on Pattern Recognition*, P 493-498, 1994
17. D.P. Huttenlocher and S. Ullman, "Recognizing soild object by alignment with an image," *International Journal of Computer Vision*, 5(2):195-212, 1990.
18. C.F. Olson, "Efficient pose clustering using a randomized algorithm," *International Journal of Computer Vision*, 23(2):131-147, June 1997.