Aerospace **2015**, *xx*, 1-x; doi:10.3390/——

OPEN ACCESS
aerospace
ISSN 2226-4310
www.mdpi.com/journal/aerospace

Article

Position Estimation using Image Derivative

Daniele Mortari [1],*, Francesco de Dilectis [1] and Renato Zanetti [2]

[1] Aerospace Engineering, Texas A&M University, College Station, Texas 77843, USA
[2] NASA Johnson Space Center, Houston, Texas 77058, USA

* Author to whom correspondence should be addressed; mortari@tamu.edu; Tel: +1 (979) 845-0734; Fax: +1 (979) 845-6051

Received: xx / Accepted: xx / Published: xx

Abstract: This article describes an image processing algorithm to identify size and shape of a spherical reflecting celestial body prominently depicted in images taken from a spacecraft with an optical camera, with the purpose of estimating the relative distance between target and observer in magnitude and direction. The approach is based on the fact that in such images the pixels belonging to the target's hard edge have the highest values of the image derivative, therefore they are easily recognizable when the image is processed with a gradient filter. Eventual extraneous points polluting the data set (outliers) are eliminated by two methods applied in sequence. The target center and radius are estimated by non-linear least squares using circular sigmoid functions. The proposed image processing has been applied to real and synthetic Moon images. An error analysis is also performed to determine the performance of the proposed method.

Keywords: Navigation; Image Processing; Autonomy

1. Introduction

Autonomous positioning has been a topic of research for many years [1] but its implementation has proven more complicated than autonomous attitude determination. As missions become more complex, it is desirable for a spacecraft to be able to estimate its position independently from ground, if not all the time, at least as a back up in case of loss of communication, either accidental or foreseen. In particular, spacecraft orbiting or en route to a satellite or a planet other than the Earth are periodically out of line-of-sight from any ground station and thus have to rely on their on-board capabilities. Utilizing pre-existing sensors to obtain a navigation solution is an attractive solution to the backup autonomous navigation problem. Among those, optical cameras have been a mainstays of spacecraft sensor arrays since the beginning of space exploration [2]. The availability of onboard cameras, together with the advancement of the fields of image processing and feature identification, has led to the proposed approach of an algorithm for autonomous spacecraft positioning via analysis of optical images of a reflective celestial body in proximity of the spacecraft itself. This work refines a previous study [3,4], focusing on the analysis of Moon images, both real and synthetic. However the basic idea is applicable to any spherical reflective celestial body within relative proximity to the spacecraft.

The method is described in detail in the following sections. Further, several application examples are illustrated, together with an error analysis.

2. Concept

The starting point of the method is the realization that in an image taken by a spacecraft of a reflective celestial body, if the illumination conditions are right and the pointing appropriate, the target will figure prominently in the image, against a mostly black background. Therefore, processing the image with a gradient filter, pixels belonging to the hard edge have the highest magnitudes, which makes them easily identifiable. These points can then be best fitted to estimate the apparent target radius and center, and thus the relative distance between observer and target. Several steps need to be taken to ensure the data points are properly selected and the best fitting is accurate enough. The proposed algorithm can be summarized as follows:

- The geometrical illumination conditions and the expected size of the observed body are first computed. These data allow to determine whether the target is sufficiently illuminated and of sufficient size to perform the estimation process.
- The image differentiation is obtained with a fourth order Richardson extrapolation.
- The pixels of the gradient image are sorted in descending mode. Eventual outliers present in this set, due to noise or background stars, are then eliminated by two sequential filters.
  - The first filter validates an edge pixel based on eigenanalysis of a small gradient box around the pixel and by investigation of the same box in the original image.
  - The second filter is a modified version of the RANSAC algorithm.
- The edge point selected are then used to obtain a first estimate of center and radius. The Taubin best fitting algorithm [7] is used.
- The first estimate is refined via non-linear least-squares using a circular sigmoid function to describe the edge transition.

Required inputs to the image processing software are (see Fig. 1):

Camera. This includes: focal length (f) in mm and imager data (number and size of rows and columns).

Image. This is the image (made of N_r rows and N_c columns) taken at the specified time.

Time. This is the time instant at which the image is taken. This time is used to compute the Moon and Sun positions in J2000.

Position. This input consists of a raw estimate of the spacecraft position, $E\{\boldsymbol{r}\}$, and its accuracy, $\sigma_{\boldsymbol{r}}$. These data are used to estimate the Moon radius on the imager and the expected illumination condition. This position estimate is very rough, having typical accuracy of less than $10,000$ km.

Attitude. This input consists of the Spacecraft attitude, $E\{\boldsymbol{q}\}$, and attitude accuracy, $\sigma_{\boldsymbol{q}}$. These data are used to compute the Spacecraft-to-Moon vector in the J2000 reference frame.

Figure 1 illustrates the algorithm.

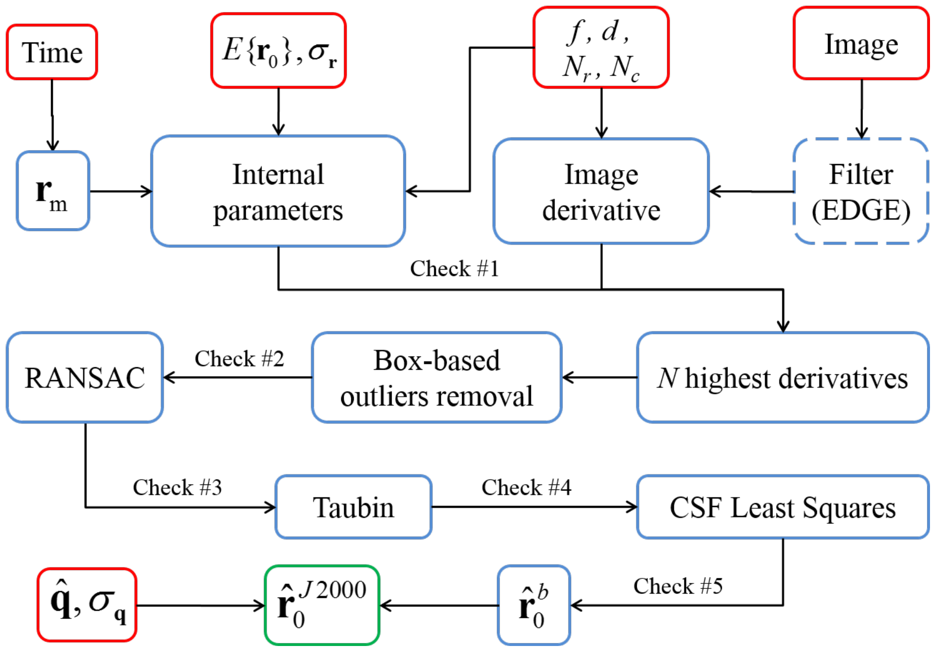

Figure 1. Image Processing General Flowchart

A first check on the data is done using rough expected value for the parameters, the following section describes these calculations done for the Moon, however they are applicable to any (almost) spherical celestial body. The following is based on geometric considerations alone and thus does not involve processing of the image data. This is why the parameters are described as "expected" values.

3. Expected Radius and Illumination Parameter

As the relative positions of observer, target, and light source change, the visible part of the target varies. These changes can lead to cases in which the target is too thin or completely dark from the point of view of the observer, and therefore no analysis can take place. The algorithm catches these degenerate cases before they can cause critical errors further down in the process. Two parameters are introduced to describe the target's shape (from now on, for simplicity, the term "target" will refer just to the illuminated portion rather than the whole object, as this is what the camera is actually able to

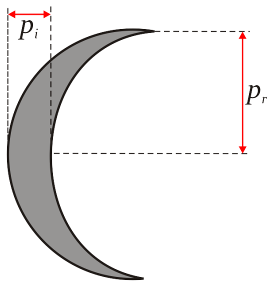

Figure 2. Expected Moon radius (p_r) and illumination parameter (p_i).

observe): one is the expected radius of the target p_r, and the other is the cross-length measured at its thickest point p_i (see Figure 2), which is referred to as the *illumination parameter*. Both these distances are measured in pixels, and they are estimated from the camera parameters (focal length f, pixel size d) and the observation geometry. To this purpose, the observer's position is needed, however, even a high uncertainty ($\sigma_D \geq 10,000$ km) has very small impact on p_r and p_i, as long as the spacecraft is further from the target than a certain threshold. Such a rough position estimate can be considered known *a priori*, to start the analysis.

To begin, expected radius and illumination angle are computed as:

$$p_r = \frac{f}{d} \cdot \frac{R_m}{\sqrt{|\boldsymbol{r}_o|^2 - R_m^2}} \qquad \text{(pixels)} \tag{1}$$

$$\vartheta = \arccos(\hat{\boldsymbol{r}}_s \cdot \hat{\boldsymbol{r}}_o). \tag{2}$$

where $\hat{\boldsymbol{r}}_s$ is the Sun-to-Moon rays direction, $\boldsymbol{r}_o$ the Observer-to-Moon vector, and R_m the Moon radius. The expected radius is of great importance, not only because the illumination parameter is measured in relation to it, but also because it predicts the overall size of the target in the image, which is used in subsequent parts of the algorithm. The value of p_i is determined differently depending on the values of p_r and ϑ.

3.1. Full or New Moon With reference to Figure 3, the angle α_m can be derived from simple trigonometry

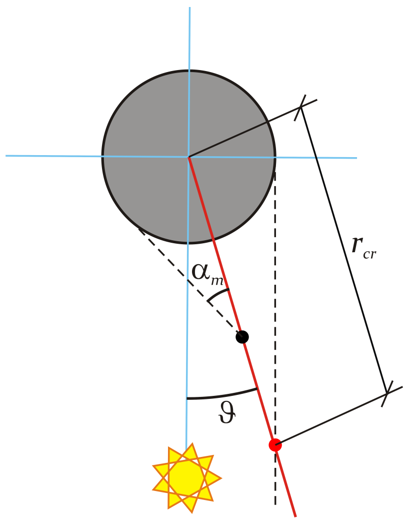

Figure 3. Full/New Moon geometry.

$$|\boldsymbol{r}_o| \, \sin\alpha_m = R_m. \tag{3}$$

This equation also allows to define a critical distance, r_{cr}, as the distance at which $\alpha_m = \vartheta$:

$$r_{cr} \, \sin\vartheta = R_m \tag{4}$$

Whether $|\boldsymbol{r}_o|$ is greater or smaller than r_{cr}, together with the value of ϑ, determines the case, according to the following scheme:

$$\text{if} \quad |\boldsymbol{r}_o| < r_{cr} \quad \text{and} \quad \begin{cases} \vartheta < \pi/2 & \text{then} \quad p_i = 2p_r \quad \text{(Full Moon)} \\ \vartheta > \pi/2 & \text{then} \quad p_i = 0 \quad\;\; \text{(New Moon)} \end{cases} \tag{5}$$

3.2. Gibbous Moon Gibbous Moon occurs when

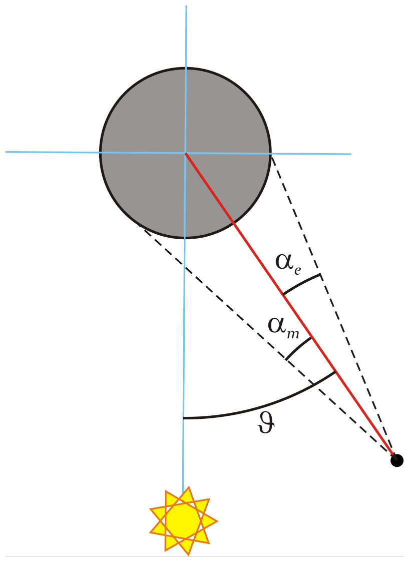

Figure 4. Gibbous Moon geometry.

$$|\boldsymbol{r}_o| > r_{cr} \qquad \text{and} \qquad \vartheta < \frac{\pi}{2}. \tag{6}$$

For this configuration (see Figure 4), p_i can be computed by finding the auxiliary variables x and α_e:

$$x^2 = |\boldsymbol{r}_o|^2 + R_m^2 - 2\,|\boldsymbol{r}_o|\,R_m \,\sin\vartheta \quad \rightarrow \quad x \tag{7}$$

$$\sin\alpha_e = R_m \,\frac{\cos\vartheta}{x} \quad \rightarrow \quad \alpha_e \tag{8}$$

which allows to find the “excess” factor p_e:

$$p_e = p_r \,\frac{\tan\alpha_e}{\tan\alpha_m} \tag{9}$$

and, therefore,

$$p_i = p_r + p_e = \left(1 + \frac{\tan\alpha_e}{\tan\alpha_m}\right) p_r. \tag{10}$$

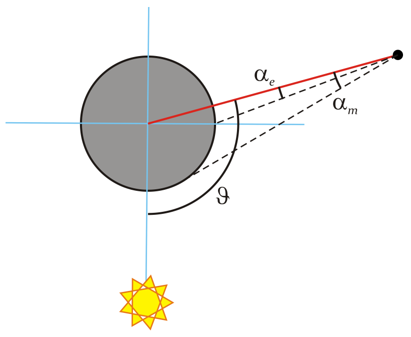

Figure 5. Crescent Moon geometry.

3.3. Crescent Moon Crescent Moon occurs when

$$|\boldsymbol{r}_o| > r_{cr} \qquad \text{and} \qquad \vartheta > \frac{\pi}{2}. \tag{11}$$

In this case (see Figure 5) the two variables x and α_e can be found according to:

$$x^2 = |\boldsymbol{r}_o|^2 + R_m^2 - 2\,|\boldsymbol{r}_o|\,R_m\,\sin\vartheta \quad \rightarrow \quad x \tag{12}$$

$$\sin\alpha_e = \frac{-\cos\vartheta}{x}\,R_m \quad \rightarrow \quad \alpha_e \tag{13}$$

and therefore:

$$p_i = p_r - p_e = \left(1 - \frac{\tan\alpha_e}{\tan\alpha_m}\right) p_r \tag{14}$$

Differentiating among these cases is important to properly initialize the algorithm. To this purpose, the analytical relations described above are supplemented with thresholds to ensure the system works properly; for instance, when p_i is lower than a certain value, the crescent is deemed too thin to be a reliable source of information, and the target is considered invisible.

3.4. Special Case: Cropped Target

It is also possible for the target not to be completely within the field-of-view. The images' boundaries are sept with a rectangular mask proportional to the dimensions of the image itself. At each step the cumulative brightness within the mask is computed and compared with a threshold value, for example a valued tested to provide good performance is $1/10$ of the maximum greytone. From the amount of threshold crossings, it is possible to determine if and how many times the target has been cropped by the frame. An example application is depicted in Figure 6.

The values of p_r and p_i are used to initialize the following parameters:

- Number of pixels in the gradient image to be analyzed (N). This number has to be proportional to the length of the visible Moon arc. However, at this stage it is impossible to know the angular width of the target; moreover, the presence of the terminator introduces a number of high contrast pixels that are not that are not part of the Moon edge. Ultimately N is chosen to be proportional to p_r and p_i, according to the following:

$$N = k\,\pi p_r$$

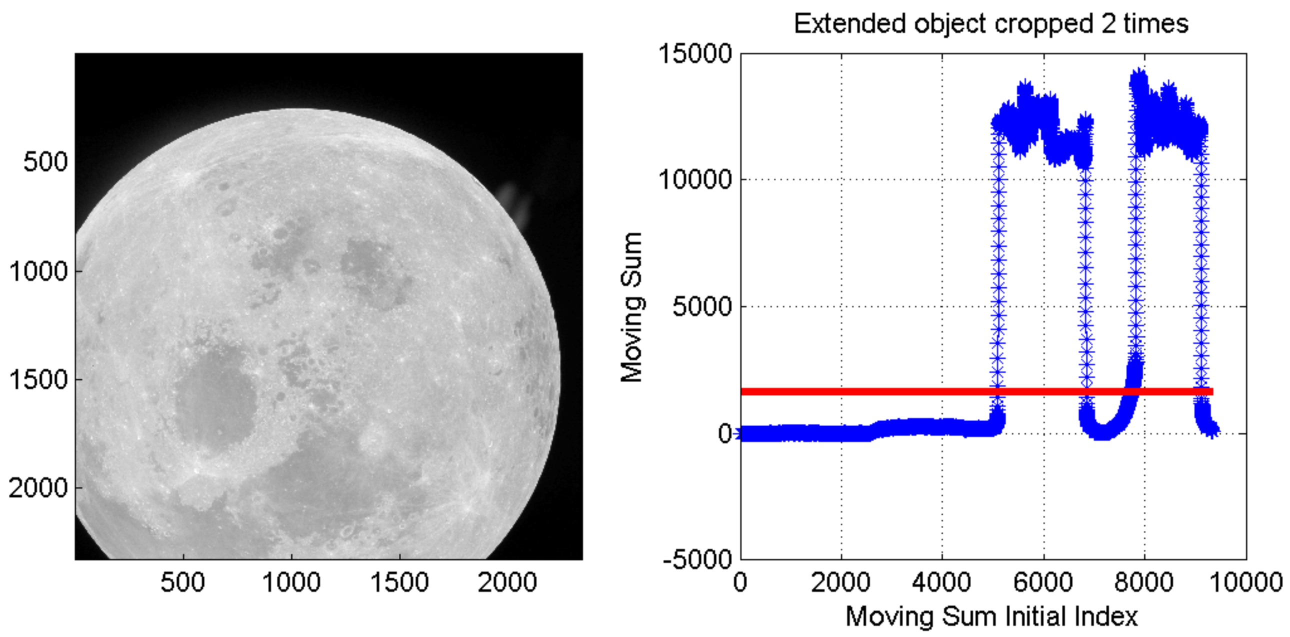

Figure 6. Processing of cropped image. Left: cropped image of the Moon. Right: results of the cropped detection.

where k is a factor that varies linearly with p_i, from a minimum value (2) when $p_i \leq 10$ to a maximum equal to twice the minimum, when $p_i \approx 2p_r$.

$$k = 2\left(1 + \frac{p_i - 10}{2\,p_r - 10}\right)$$

- Size of the box for box based outliers removal (s). Because of the way the code works, the box has to be chosen small enough that the segment of Moon edge within the box appears almost straight. Extensive testing has shown that the best choice is to take $s = \lceil 0.8p_r \rceil$, with a lower limit of $s = 3$ and an upper limit of $s = 21$.

- RANSAC maximum distance. Chosen equal to $d_{\max} = 5\,p_r/100$.

- Number of tests for RANSAC (N_T). RANSAC randomly selects for testing a subset of all the possible triplets of pixels remaining after the box based outliers removal. Testing has shown that the best choice is to have

$$N_T = \frac{1}{5}\binom{N_1}{3} = \frac{N_1\,(N_1-1)(N_1-2)}{30}$$

where N_1 is the number of selected pixels remained after the box-based outliers filter rejection. The value of 500 is enforced as an upper limit.

- Convergence tolerance for Circular Sigmoid Function (CSF) Least Squares [3]. At k-th iteration, the convergence check is performed on the solution variation $\Delta X_k = \sqrt{|\Delta r_k^2 + \Delta c_k^2 - \Delta R_k^2|}$. Assuming the requirement accuracy is 0.3 pixel (in center and radius estimation), the CSF-LS converge criteria is assumed to be 1/50 smaller than that, $\Delta X_{\min} = 0.3/50$.

- Number of iterations for CSF Least Squares is $N_{iter} = 50$.

As indicated in Figure 1, several checks are implemented throughout the code. Failure to pass any of the tests cause the process to abort.

- **Check #1** (after Internal Parameters computation).
    1. Moon is too small (apparent diameter less than a set number of pixel) or too big (Moon completely or almost completely filling the image).
    2. Moon is new (or otherwise its illuminated fraction is negligible) or out of frame altogether.
- **Check #2** (after box based outliers removal, see Section 4.2).
    - Too many outliers have been identified, leaving the pool of valid data below a set limit.
- **Check #3** (after RANSAC based outliers removal, see Section 4.3).
    - Too many outliers have been identified, leaving the pool of valid data below a set limit.
    - The estimated radii found by RANSAC are not consistent with the expected apparent radius previously found.
- **Check #4** (after Taubin, see Section 5.1).
    - The radius found is not consistent with the expected value.
    - Computed center position of the Moon is out of the imager and too far away form the optical axis.
- **Check #5** (inside and after CSF-LS).
    - Exceeded maximum number of iterations.
    - Estimates of radius and center are too different from the results of Taubin.

4. Gradient-based image processing algorithm

The gradient-based image processing approach is based on the expectation that the part of the image with strongest variation of gray-tones is the Moon observable hard edge. Pixels with highest derivatives are the most likely to belong to the hard edge. However, the presence of bright stars, Moon surface roughness and other unexpected objects within the field-of-view, creates several bright gradient pixels which are not in fact part of the Moon hard edge. These points, called *outliers*, must therefore be eliminated from the data set prior to using a best fitting algorithm to determine the sought geometrical properties. In this work, two separate outliers removal method are implemented in sequence.

4.1. Image differentiation

The gradient of the discrete greytone function is computed by numerical differentiation, in particular the gradient is computed using four point central difference with a single Richardson extrapolation.

The row and column four point central partial differences computed at pixel $[r, c]$ are

$$\left\{\begin{array}{lcl} g_r'(r,c) & = & \dfrac{8[I(r+1,c)-I(r-1,c)]-[I(r+2,c)-I(r-2,c)]}{12\,h} \\ \\ g_c'(r,c) & = & \dfrac{8[I(r,c+1)-I(r,c-1)]-[I(r,c+2)-I(r,c-2)]}{12\,h} \end{array}\right. \tag{15}$$

These derivatives are accurate with order h^4, where h is the pixel dimension. The image gradient is then computed as

$$g'(r,c)=\sqrt{g_r'^2(r,c)+g_c'^2(r,c)} \tag{16}$$

A single Richardson extrapolation requires computing Eq. (16) one time with step $2\,h$ (getting g_{2h}') and one time with step h (getting g_h'). Therefore, the true derivative can be written as

$$\left\{\begin{array}{lcl} g'|_{true} & = & g_{2h}'+c\,(2h)^4 \\ g'|_{true} & = & g_h'+c\,h^4 \end{array}\right. \tag{17}$$

Equating these two equation we obtain

$$g_h'-g_{2h}'=c\,h^4(2^4-1) \quad \rightarrow \quad c\,h^4=\frac{1}{2^4-1}\,(g_h'-g_{2h}') \tag{18}$$

and therefore

$$\boxed{g'|_{true}=g_h'+\frac{1}{2^4-1}\,(g_h'-g_{2h}')} \tag{19}$$

An example of application of a single Richardson extrapolation is done using the six-hump Camel back function

$$f(x,y)=\frac{x^2}{4}\left(4-\frac{21}{40}x^2+\frac{x^4}{8}\right)+\frac{xy}{4}+y^2\left(\frac{y^2}{4}-1\right) \tag{20}$$

Fig. 7 shows the accuracy of a single Richardson extrapolation

In this figure the errors with respect to the true analytic values are show for the four cases:

- Top-left: 2-point central differentiation and no Richardson extrapolation;
- Top-right: 4-point central differentiation and no Richardson extrapolation;
- Bottom-left: 2-point central differentiation and one Richardson extrapolation;
- Bottom-right: 4-point central differentiation and one Richardson extrapolation;

These results show that: 1) the accuracy provided by 4-point central differentiation and no Richardson extrapolation is equivalent to that provided by 2-point central differentiation and one Richardson extrapolation and 2) the 4-point central differentiation and one Richardson extrapolation provides almost the machine error accuracy. For this reason the 4-point central differentiation and one Richardson extrapolation is selected to evaluate which are the pixels with greatest gradient.

The image gradient pixels are sorted by gradient value in descending mode, the outliers are then eliminated with two methods discussed below.

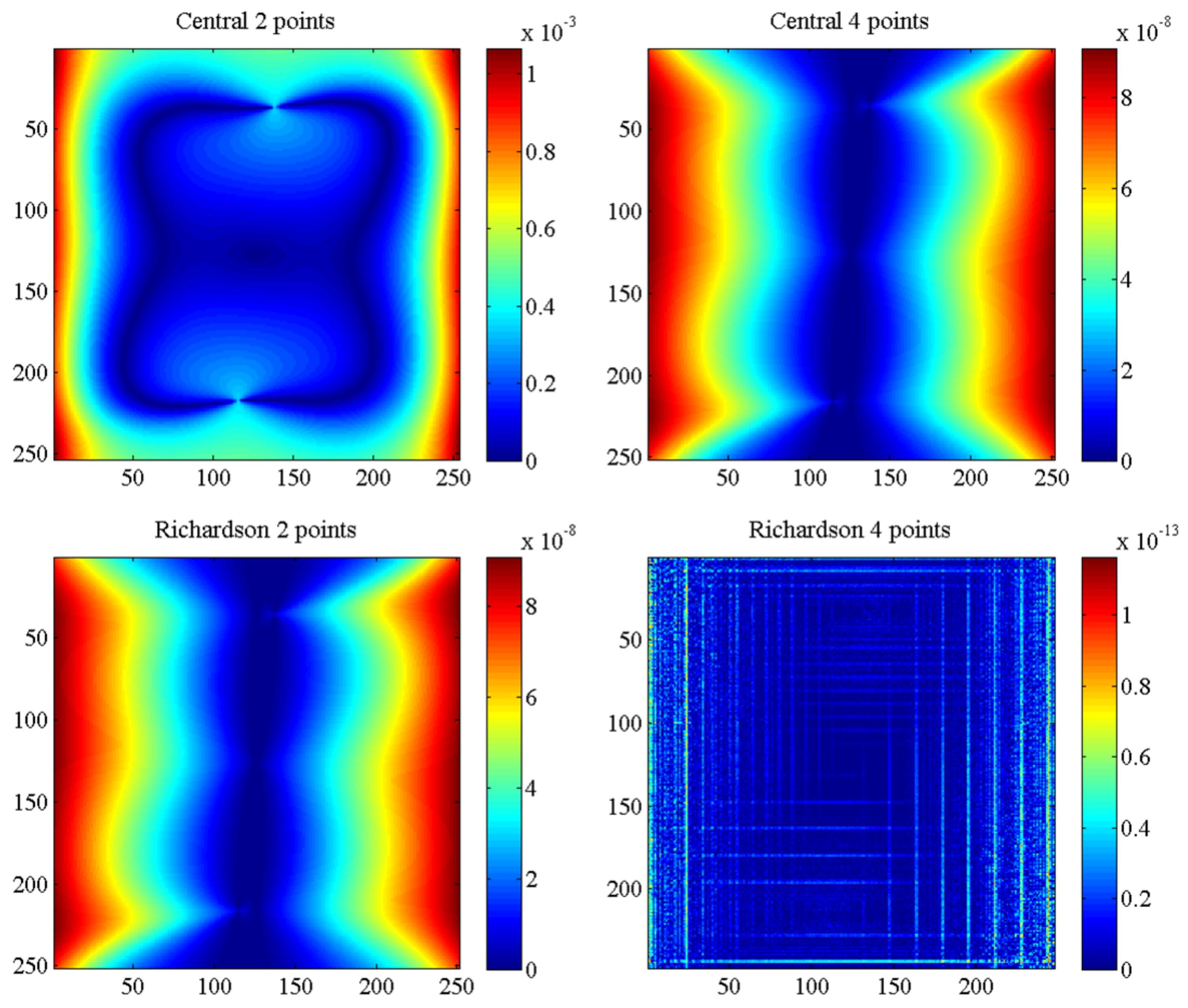

Figure 7. Accuracy results for the six-hump Camel back function example

4.2. Box-based outliers Identification The first approach is based on a local analysis around the pixel being considered as potentially belonging to the target's edge. A mask like the one in Figure 8, whose size depends on the parameter p_r, is centered on a candidate pixel. If the pixel does indeed belong to the target's edge, then it is likely to be part of a sequence of pixels of comparable brightness distributed along a preferred direction. Moreover, the pixel should sit on the border between two areas where the graytone "flattens" (the inside and outside of the target) and which therefore should appear as dark in the gradient image. If the pixel does not satisfy all of these conditions, it is rejected. To perform this analysis, first the tensor of inertia of the gradient image within the mask is computed and then the eigenvectors and eigenvalues are found. The eigenvector associated to the minimum eigenvalue (red line direction in Figure 8) is inclined by an angle γ with respect to the row axis direction and represents the direction of the supposed edge. If $\gamma \mod \pi < \pi/2$ then the farthest pixels from the edge line are those identified in green while if $\gamma \mod \pi > \pi/2$ the farthest pixels are the red ones. Several conditions have to be satisfied for the pixel to be accepted:

- The ratio between the eigenvalues must be lower than a certain threshold, chosen experimentally at 0.3. This indicates the distribution of bright pixels has a preferential direction.

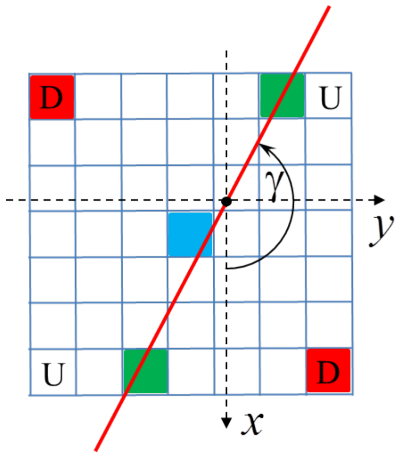

Figure 8. Mask for the box-based outlier identification technique (mask size: 7×7).

- The values of the gradient in the pixels furthest from the edge line should be small, as both should be far from areas of variable brightness. Moreover, their graytones should differ noticeably, as one belongs to the target and the other to the night sky.
- For the pixels highlighted in green in Figure 8, the ratio of their gradient values should be close to 1 ($g_{\min}/g_{\max} > 0.6$) as they both supposedly belong to the edge.
- For the pixels highlighted in red in the same figure, the ratio of their graytone values should be close to 0 ($g_{\min}/g_{\max} < 0.3$) as one belongs to the nught sky and the other to the target.

4.3. RANSAC for circle and ellipse The second technique is the standard RANSAC algorithm which is

an abbreviation for "RANdom SAmple Consensus" [5]. It was developed for robust fitting of models in the presence of many data outliers. The algorithm is very simple and can be applied in the fitting of many geometric entities, such as lines, circles, ellipses, etc. It is a non-deterministic iterative algorithm, in the sense that it produces a reasonable result only with a given probability, which increases as more iterations are allowed. It works by selecting random subsets of size equal to the number of unknown parameters, and then checks whether the remaining datapoints lie further than a certain threshold from the curve determined by the initial subset. Repeating this process for many randomly chosen subsets is likely to find and remove the outliers. More formally, given a fitting problem with n data points and unknown m parameters, starting with $N_{\min} = n$, the classic RANSAC algorithm perform N_{test} times the following steps:

1. selects m data points at random.
2. finds the curve passing through these m points, thus estimating the m parameters associated.
3. counts how many data items are located at a distance greater than a minimum value $d_{\min}$ from the curve. They are N_k.
4. if $N_k < N_{\min}$ then set $N_{\min} = N_k$.

The basic cycle is depicted in Figure 9.

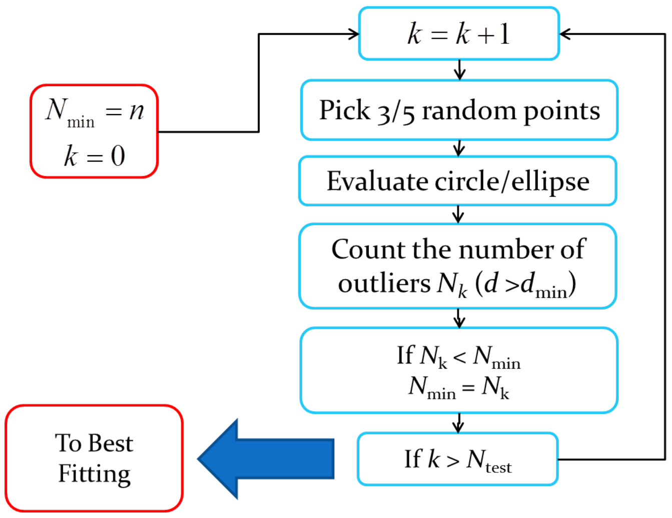

Figure 9. RANSAC flowchart.

Crucial to a successful implementation of RANSAC is the method used for the random selection of subsets. A purely random selection approach may persist choosing outliers, with great waste of computational time. On the other hand, a procedure based on deterministic inner loops can also be inefficient, if by chance one outlier is selected by the outmost loop. Therefore, as an improvement on the basic method, a technique originally developed for Pyramid Star-Identification [6], has been adopted, which always selects different subsets by continuously rotating the dataset indexes.

5. Circle Best Fitting

Once the pixels in the image have been selected, the algorithm determines the geometric characteristics of the observed body, that is, apparent radius and center position within the image. The problem of recognizing geometrical entities and reduce them to a minimal set of parameters, sometimes called *feature extraction*, is very important in image processing for a number of applications in computer graphics and computer vision. Consequently a great amount of research exists into developing methods to efficiently and unequivocally recognize geometrical shapes within an image; in particular, much literature is available regarding circle identification. Identifying geometrical figures within a discrete representation of the physical world reduces essentially to a best fitting problem. Currently, feature recognition techniques for conic curves fall within one of the following categories:

Geometric Fit: given a set of n data points, iteratively converge to the minimum of

$$\mathbb{L}_g = \sum_{i=1}^{n} d_i^2(\boldsymbol{x}) \tag{21}$$

where d_i are geometric distances from the data points to the conic, and $\boldsymbol{x}$ is the vector of the parameters to be found.

Algebraic Fit: d_i are replaced with some other function of the parameters f_i, and thus a new cost function is defined:

$$\mathbb{L}_a = \sum_{i=1}^{n} f_i^2(\boldsymbol{x}) \tag{22}$$

These functions typically are algebraic expression of the conic in question, which are usually simpler than the expression for geometric distances.

Historically, geometric fit has been regarded as being the more accurate of the two. However, this accuracy comes at the price of added computational cost, and possibility of divergence. Algebraic fit is typically reliable, simple and fast. A common practice is to use an algebraic fit to generate an initial guess for a subsequent iterative geometric fit, but in recent years, new algebraic fitting routines have shown such a degree of accuracy that geometric fits cannot improve it noticeably [8].

5.1. Taubin best fitting The following is a description of the Taubin best fitting method, first described in [7], but using a formalism found in [8]. This algorithm is used to obtain an initial estimate of the sigmoid best fitting algorithm. Given the circle implicit equation as

$$a\,(x^2+y^2)+b\,x+c\,y+d=a\,z+b\,x+c\,y+d=0 \tag{23}$$

the Taubin Best Fit minimizes the function:

$$\mathcal{F}_T\,(a,b,c,d)=\frac{n\sum_i(a\,z_i+b\,x_i+c\,y_i+d)^2}{\sum_i(4a^2\,z_i+4ab\,x_i+4ac\,y_i+b^2+c^2)} \tag{24}$$

which is equivalent to minimize

$$\mathcal{F}_T\,(a,b,c,d)=\sum_i(a\,z_i+b\,x_i+c\,y_i+d)^2 \tag{25}$$

subject to the constraint

$$4a^2\,\bar{z}+4ab\,\bar{x}+4ac\,\bar{y}+b^2+c^2=1 \tag{26}$$

where the bar sign indicates simple average of the variables.

$$\bar{x}=\frac{1}{n}\sum_{i=1}^{n}x_i,\qquad \bar{y}=\frac{1}{n}\sum_{i=1}^{n}y_i,\qquad \text{and}\qquad \bar{z}=\frac{1}{n}\sum_{i=1}^{n}(x_i^2+y_i^2)=\frac{1}{n}\sum_{i=1}^{n}z_i. \tag{27}$$

One important property of this method is that the results are independent of the reference frame. By reformulating this problem in matrix form, it becomes apparent how it is equivalent to a generalized eigenvalue problem. Indeed, define

$$A=\begin{Bmatrix}a\\b\\c\\d\end{Bmatrix},\quad Z=\begin{bmatrix}z_1&x_1&y_1&1\\\vdots&\vdots&\vdots&\vdots\\z_n&x_n&y_n&1\end{bmatrix},\quad\text{and}\quad M=\frac{1}{n}Z^{\mathrm{T}}Z=\begin{bmatrix}\overline{zz}&\overline{zx}&\overline{zy}&\bar{z}\\\overline{zx}&\overline{xx}&\overline{xy}&\bar{x}\\\overline{zy}&\overline{xy}&\overline{yy}&\bar{y}\\\bar{z}&\bar{x}&\bar{y}&1\end{bmatrix}, \tag{28}$$

then the best fitting problem can be rewritten as

$$A^{\mathrm{T}}\,M\,A=0\quad\text{subject to}\quad A^{\mathrm{T}}\,N\,A=1, \tag{29}$$

where

$$N=\begin{bmatrix}4\bar{z}&2\bar{x}&2\bar{y}&0\\2\bar{z}&1&0&0\\2\bar{y}&0&1&0\\0&0&0&0\end{bmatrix}. \tag{30}$$

Therefore, the optimization problem is equivalent to the following generalized eigenvalue problem:

$$M\,A = \eta\,N\,A. \tag{31}$$

Since for each solution, $[A, \eta]$

$$A^{\mathrm{T}}\,M\,A = \eta\,A^{\mathrm{T}}\,N\,A = \eta \tag{32}$$

then to minimize $A^{\mathrm{T}}\,M\,A$ the smallest positive eigenvalue is chosen. Because matrix N has rank three, such eigenvalue is found by explicitly solving the cubic characteristic equation without the necessity to implement SVD or other computationally intensive algorithms.

6. Numerical tests

To support the capability of the developed image processing code, three numerical examples of partially observed Moon are here presented. All these tests have been performed with no information about time, camera data, and observer, Moon, and Sun positions. The image derivative is obtained using the 4-point central approach with no Richardson extrapolation. These information clearly help to define the best values of the internal parameters. Since this help is missing, the internal parameters have been adopted with same values in all three tests. These values are:

1. Maximum number of highest derivatives selected: $N = 1,500$;
2. Box-based outliers elimination: box size = 21×21;
3. Box-based outliers elimination: derivative ratio > 0.6;
4. Box-based outliers elimination: corner ratio < 0.3;
5. Box-based outliers elimination: eigenvalue ratio < 0.3;
6. RANSAC: minimum distance $d_{\text{min}} = 4$ pixels;
7. RANSAC: number of tests = 100.

6.1. Example #1: Real image, four-times cropped

The original image of example #1, shown in Fig. 10, was intentionally cropped from a real image (taken from Earth) to validate the capability of the software to face a multiple cropped Moon images. Four times cropped is the also the maximum number of cropped segments possible. The four figures of Fig. 10 show the four main phases of the image processing. Once the image differentiation has been computed, the first 1,490 points with highest derivatives have been selected. This number differs from $N = 1,500$ because the image is small and the number of points selected is also a function of the total number of pixels ($N = 1,500$ is also the maximum number).

The box-based outliers elimination approach discarded 375 points while RANSAC eliminated an additional 5 points. The resulting points are shown in the bottom right figure of Fig. 11 using blue markers.

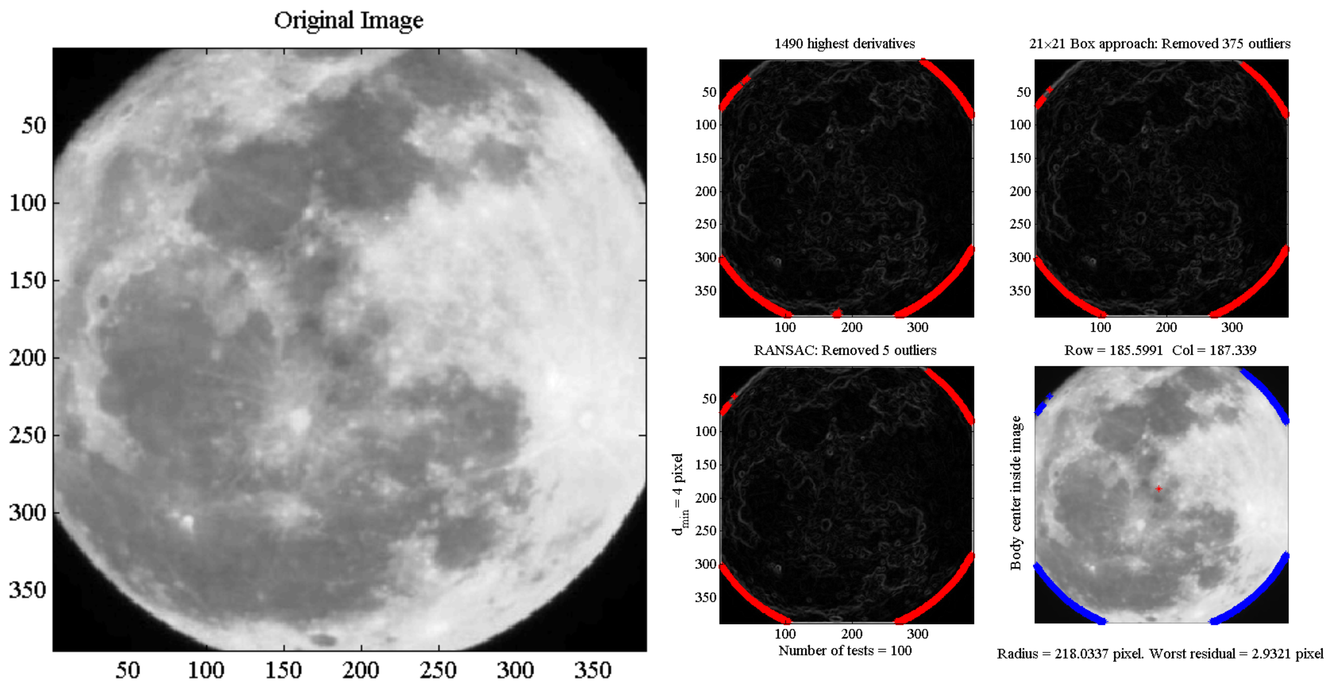

Figure 11. Example #1: Image processing phases

The bottom right figure of Fig. 11 shows the results obtained by Taubin fitting approach to estimate the circle parameters (Moon center and radius). The maximum distance from the estimated circle (worst residual) of the selected points was 2.9 pixels. The Taubin estimation gives a Moon center located at row 185.56 and column 187.34 and radius 218.03. Using this estimate, the set of points shown by red and blue markers in Fig. 12 over of Moon edge have been selected. These points are then used by CSF-LS to perform the nonlinear least-square estimation of Moon center and radius using circular sigmoid function. The new estimate implies a variation of -0.7, -0.5, and 0.4 pixels for row and column center coordinates and for radius, respectively.

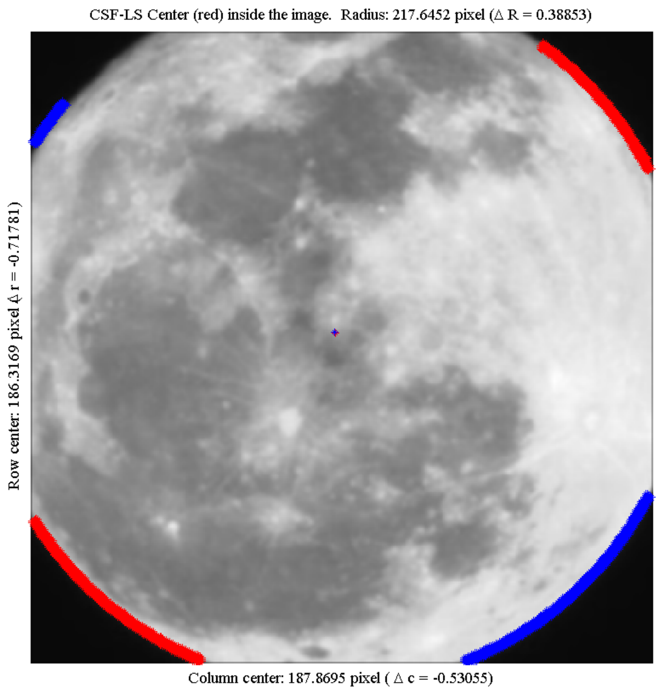
CSF-LS Center (red) inside the image. Radius: 217.6452 pixel (△ R = 0.38853)
Row center: 186.3169 pixel (△ r = -0.71781)
Column center: 187.8695 pixel (△ c = -0.53055)

Figure 12. Example #1: CSF-LS final estimation

6.2. Example #2: Synthetic image cropped

In the second example the test image is a synthetically generated image. All the relevant data and results are shown in Figs. 13, 14, and 15.

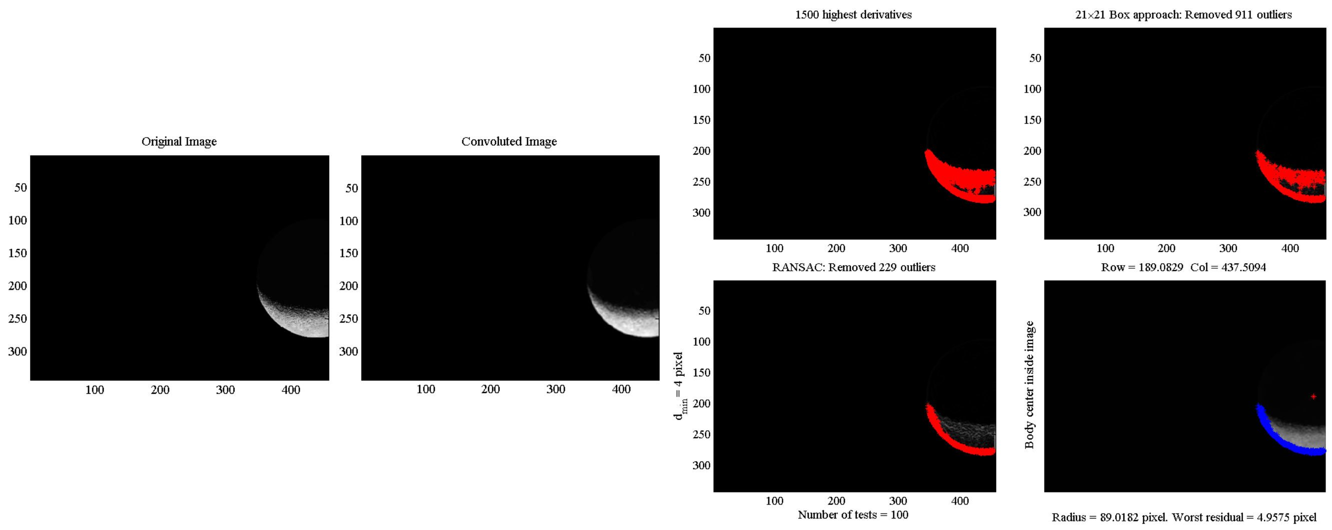

Figure 14. Example #2: Image processing phases

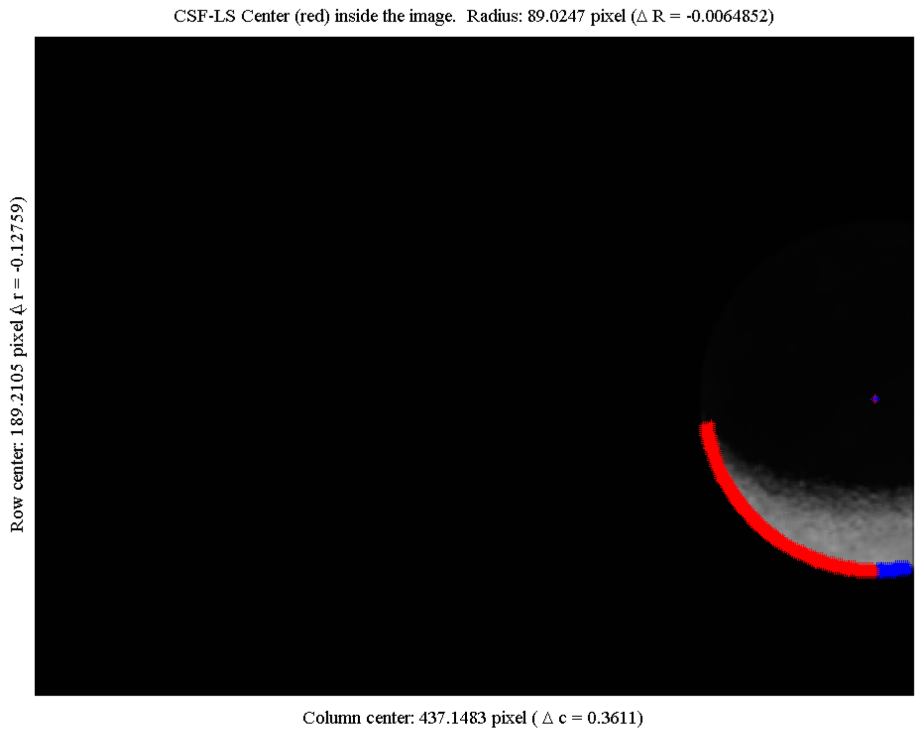

Figure 15. Example #2: CSF-LS final estimation

6.3. Example #3: Real image with center far outside the imager

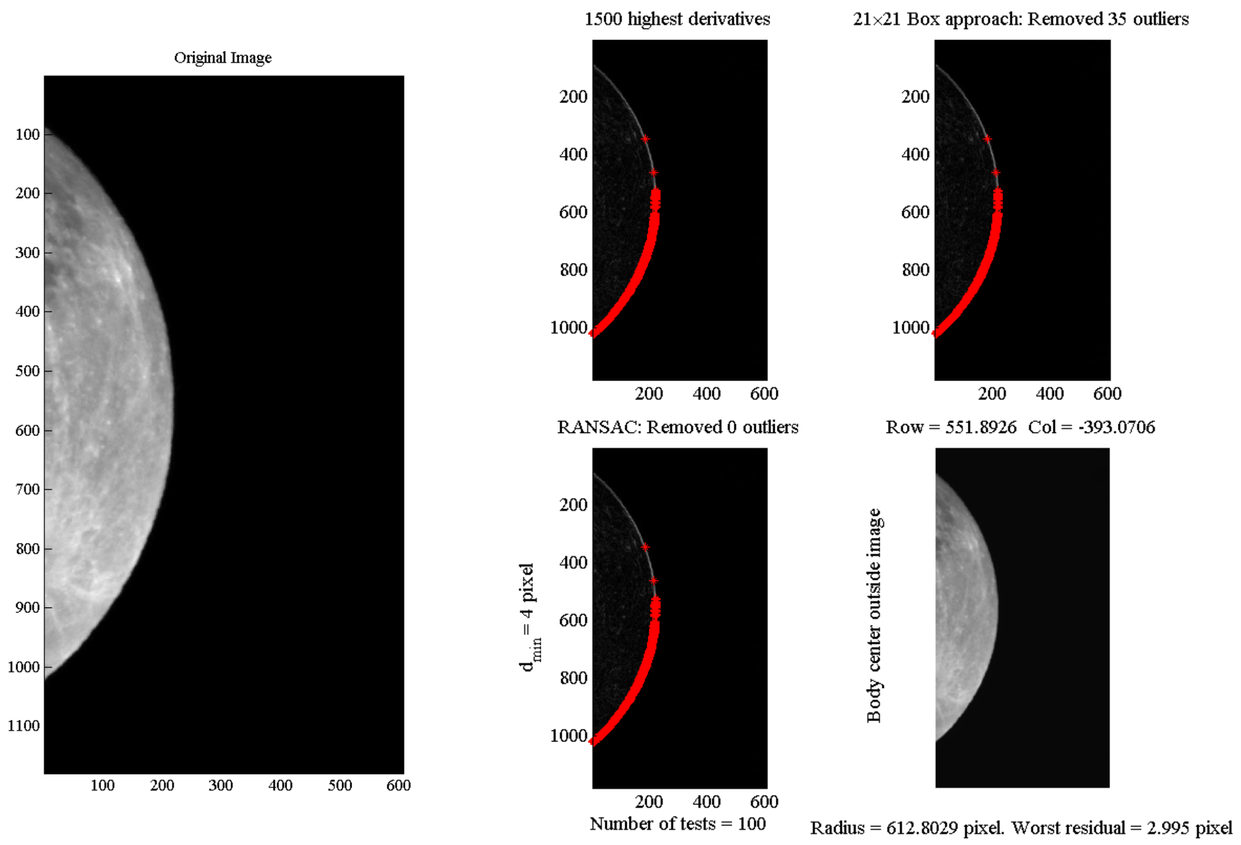

Figure 17. Example #3: Image processing phases

7. Error analysis

This section is dedicated to the error analysis associated to the following sources:

1. **Estimated position error**. This analysis quantifies how the error in the measured Moon radius (in pixel) affects the Spacecraft-to-Moon distance computation (in km).
2. **Estimated attitude error**. This analysis quantifies how the error in the camera attitude knowledge affects the Spacecraft-to-Moon direction (unit-vector).
3. **Image processing errors**. This Montecarlo analysis shows the distribution of the Moon radius and center estimated by the image processing software when the image is perturbed by Gaussian noise. This is done in the three cases of crescent, gibbous, and full Moon illumination.

7.1. Position and attitude uncertainty propagation

This section investigates the sensitivity of the Moon estimated radius to the position estimation error. It also investigates the error in the Moon center direction because of the attitude estimation error.

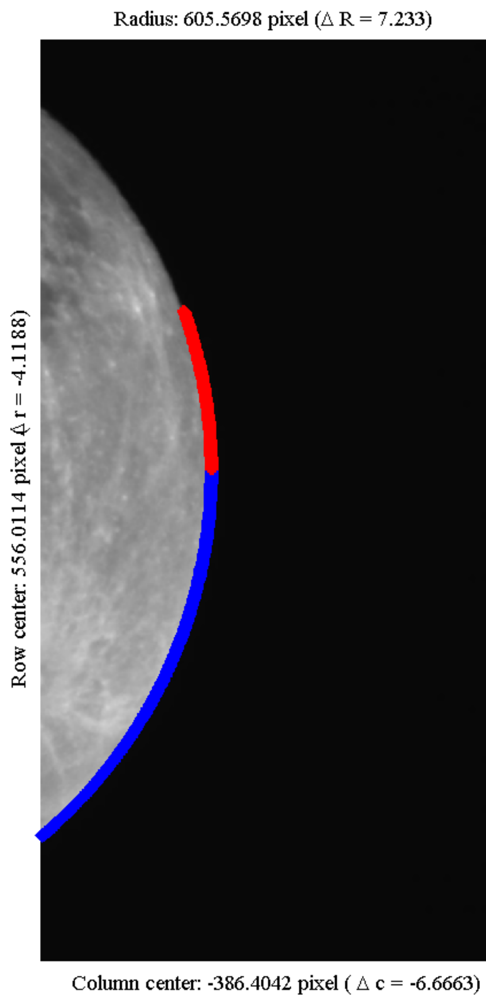
Radius: 605.5698 pixel (Δ R = 7.233)
Row center: 556.0114 pixel (Δ r = -4.1188)
Column center: -386.4042 pixel (Δ c = -6.6663)

Figure 18. Example #3: CSF-LS final estimation

The Moon radius (in pixel) is directly related to the Spacecraft-to-Moon distance. The Spacecraft-to-Moon vector is $\boldsymbol{r}_{om} = \boldsymbol{r}_m - \boldsymbol{r}$ where $\boldsymbol{r}_m$ and $\boldsymbol{r}$ are the Moon and Spacecraft position vectors in J2000, respectively. Vector $\boldsymbol{r}_m$ is simply derived from time while the Spacecraft position vector, $\boldsymbol{r}$, it can be considered a random vector variable with mean $\bar{\boldsymbol{r}}$ and standard deviation $\sigma_{\boldsymbol{r}}$, that is, $\boldsymbol{r} \sim \mathcal{N}(\bar{\boldsymbol{r}}, \sigma_{\boldsymbol{r}} \hat{\boldsymbol{u}})$, where $\hat{\boldsymbol{u}}$ is a random unit-vector variable uniformly distributed in the unit-radius sphere. Neglecting the ephemeris errors, the uncertainty in the Spacecraft vector position coincides with the uncertainty in the Spacecraft-to-Moon distance, $\sigma_D = \sigma_{\boldsymbol{r}}$.

The Moon radius in pixels in the imager is provided by

$$r = \frac{f}{d} \, \frac{R_m}{\sqrt{D^2 - R_m^2}} \tag{33}$$

where d is the pixel dimension (in the same unit of the focal length, f), $R_m = 1,737.5$ km is the Moon radius, and D is the Spacecraft-to-Moon distance. This equation allows us to derive the uncertainty in the Moon radius in the imager

$$\sigma_r = \left| \frac{\partial r}{\partial D} \right|_{\bar{\boldsymbol{r}}} \sigma_D = \frac{f \, D \, R_m}{d \, (D^2 - R_m^2)^{3/2}} \, \sigma_{\boldsymbol{r}} \tag{34}$$

The inverse of Eq. (34)

$$\sigma_D = \frac{d\,(D^2 - R_m^2)^{3/2}}{f\,D\,R_m}\,\sigma_r \tag{35}$$

provides an important information about the accuracy required by the radius estimation. This equation tells you how the radius estimate accuracy is affecting the Spacecraft-to-Moon vector length, and, consequently, how this estimate is affecting the estimation of the Spacecraft position.

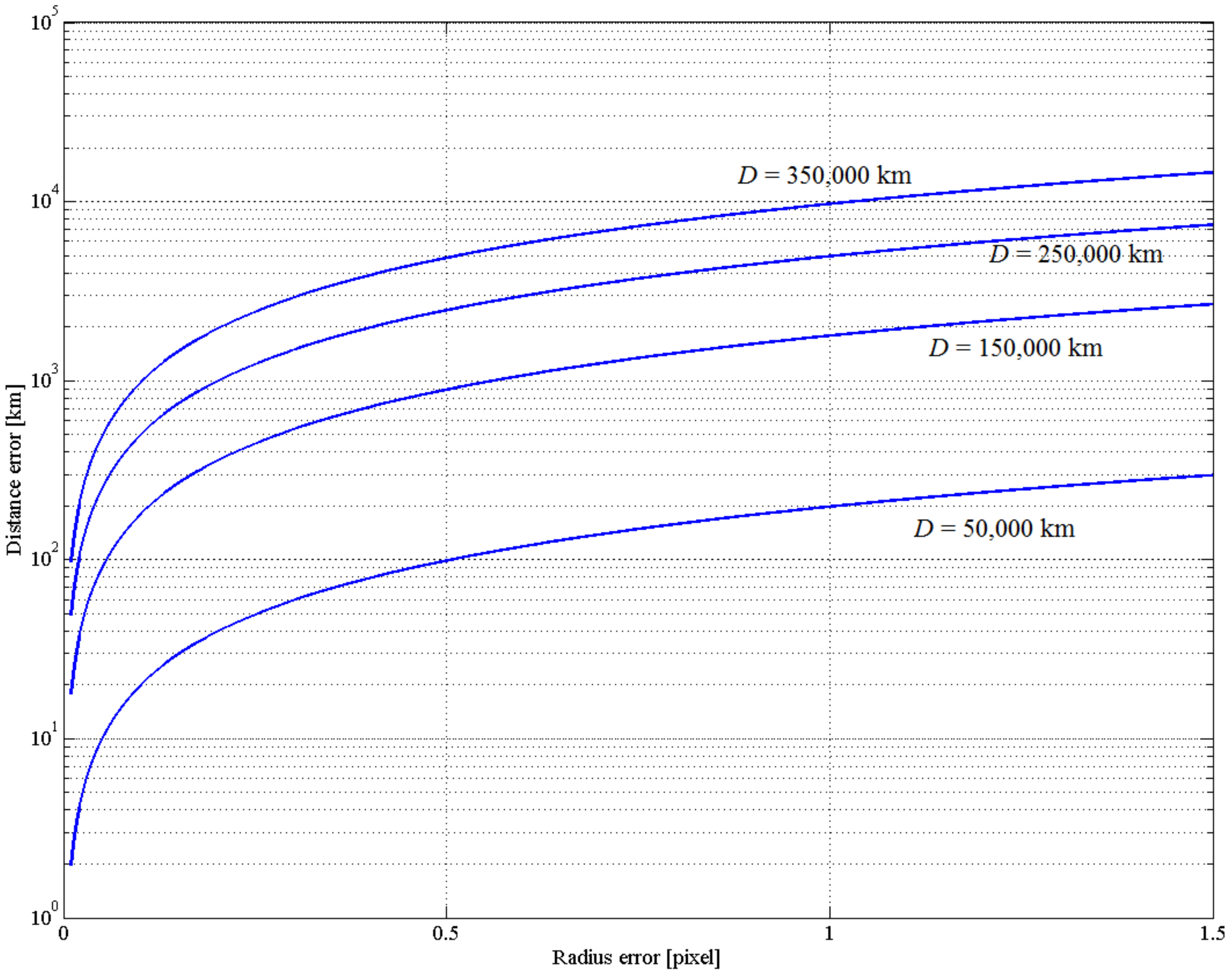

Figure 19. Moon distance error vs distance

Equation (35) has been plotted in Fig. 19 as a function of the Moon radius error and for four values of the Moon distance, $D = 50{,}000$ km, $D = 150{,}000$ km, $D = 250{,}000$ km, and $D = 350{,}00$ km.

The Moon direction uncertainty (in pixel) is caused by the uncertainty in the attitude knowledge. Consider the attitude $\mathbf{q} \sim \mathcal{N}(\bar{\mathbf{q}}, 2\sigma_{\mathbf{q}}\hat{\mathbf{u}})$ with mean $\bar{q}$ and standard deviation $2\sigma_q$. The displacement with respect to the camera optical axis is $c = \dfrac{f}{d}\tan\vartheta$, where ϑ is the angle between estimated Moon center and camer optical axis. Therefore, the radial uncertainty in the Moon center direction is provided by

$$\sigma_c = \frac{f}{d}\left|\frac{1}{\cos^2\vartheta}\right|_{\bar{q}}\sigma_q \tag{36}$$

7.2. Image processing error

Image processing error quantification requires performing statistics using a substantial set of images with known true data. Since no such ideal database is currently available, the analysis of the resulting estimated parameters (Moon radius and center) dispersion is performed by adding Gaussian noise to the same image. The Moon illumination scenario plays here a key role since when the Moon is fully illuminated the number of data (pixels) used to perform the estimation will be twice as much as in the partial illumination case (under the same other conditions: distance to the Moon and selected camera). For this reason two scenarios have been considered:

1. Partially illuminated Moon (results provided in Fig. 20)
2. Full illuminated Moon (results provided in Fig. 21)

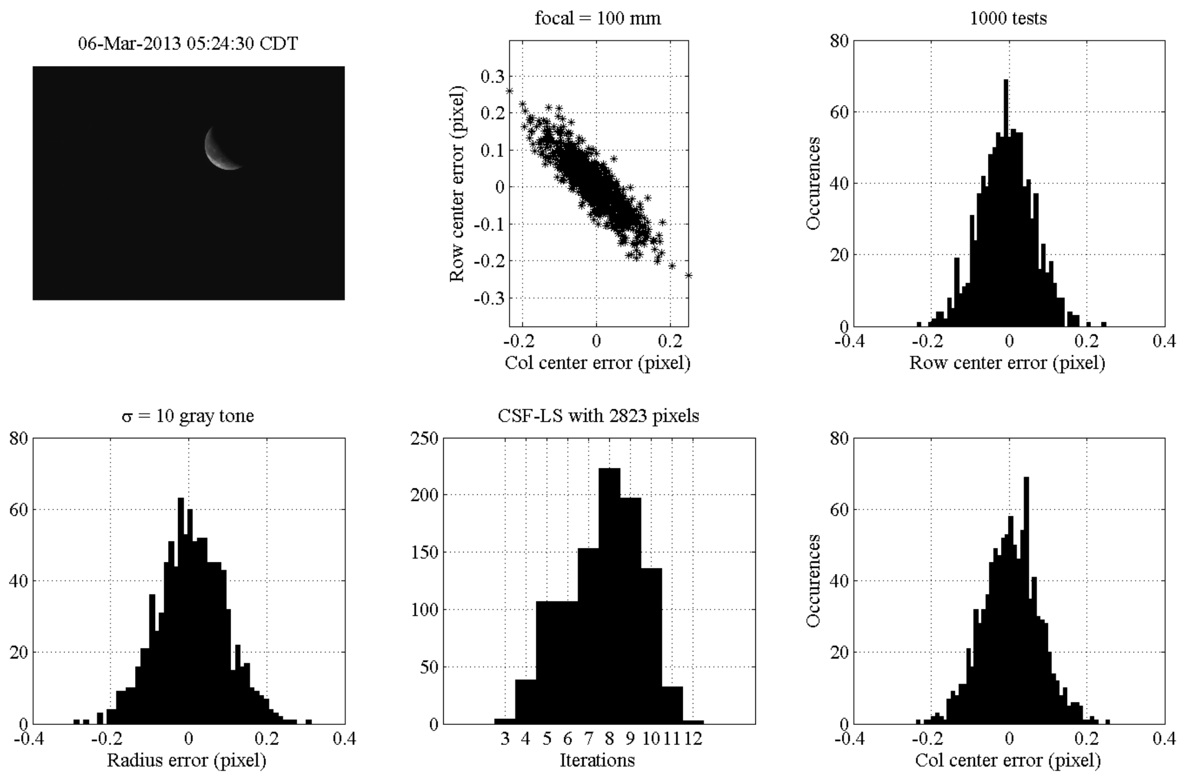

Figure 20. Image processing error results: partial illumination case

The results of the tests done using partial illumination Moon are shown in Fig. 20. The image selected is real and it was taken on March 6, 2013 at 05:24:30 using a focal length $f = 100$ mm. The statistics of the results obtained in 1,000 tests by adding Gaussian noise with zero-mean value and standard deviation $\sigma = 10$ gray tone are given. The code estimates the Moon radius and center (row and column) by least-squares using circular sigmoid function (CSF-LS). The results show the three parameters estimated as unbiased and with a maximum error of the order of 0.2 pixels (with respect to the original picture). The histogram of the number of iterations required by the least-squares to converge shows an average of 8 iterations. The central-upper plot in Fig. 20 shows the ellipsoid distribution of the Moon center error due to the fact that the estimation uses just data on one side of the Moon, on the illuminated part.

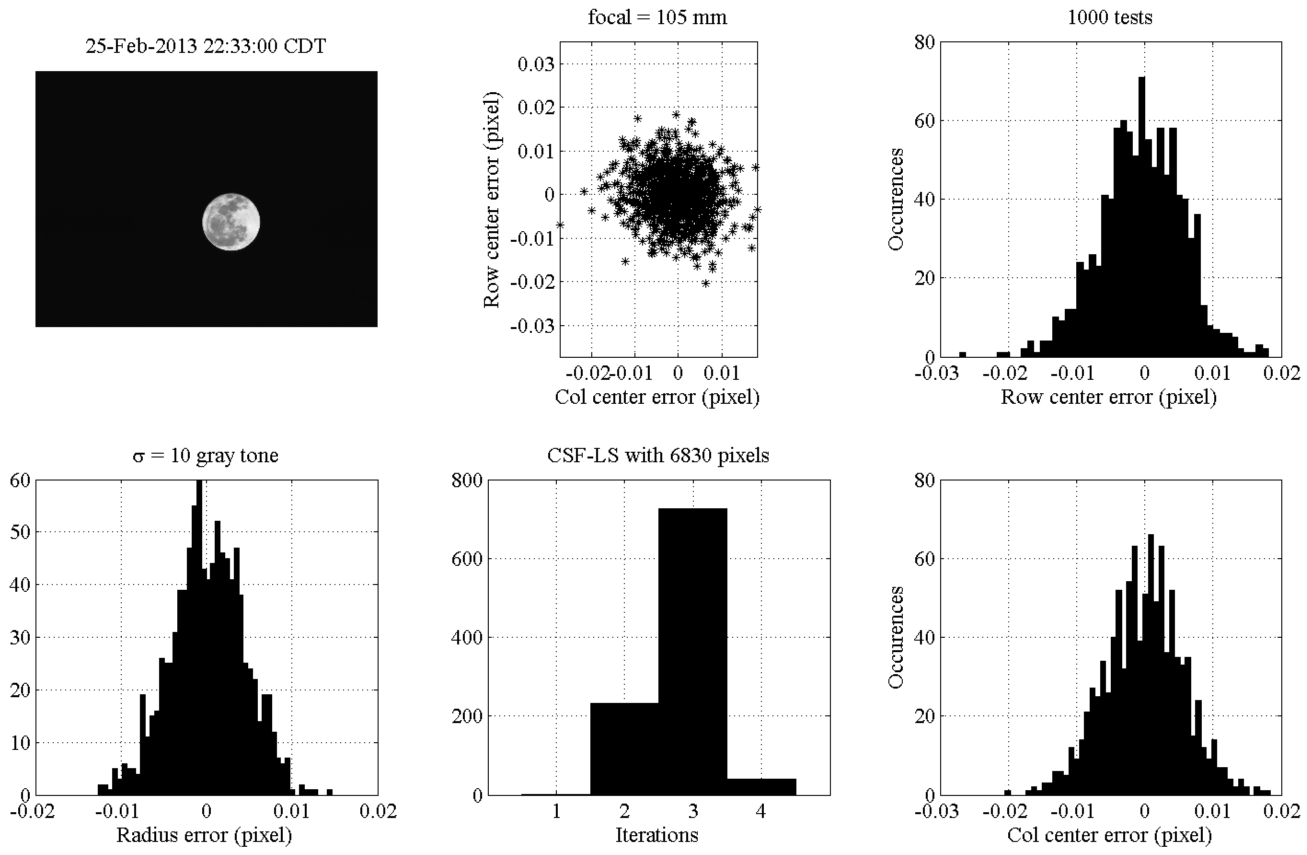

Figure 21. Image processing error results: full illumination case

The results of the tests done using full illuminated Moon are shown in Fig. 21. Also in this case the image is real and was taken in Houston on February 25, 2013 at 22:33:00 CDT using a focal length $f =$ 105 mm. The statistics of the results obtained in 1,000 tests by adding Gaussian noise with zero-mean value and standard deviation $\sigma = 10$ gray tone are given. The code estimates the Moon radius and center (row and column) by least-squares using circular sigmoid function (CSF-LS). The results show the three parameters estimated as unbiased and with a maximum error of the order of 0.01 pixels, one order magnitude more accurate than what obtained in the partial illuminated case. The histogram of the number of iterations required by the least-squares to converge shows an average of 3 iterations. The central-upper plot in Fig. 20 shows the unbiased Gaussian distribution of the Moon center error. These more accurate results are due to the fact that the estimation uses data all around the Moon edge.

8. Conclusions

This paper presents an image processing algorithm capable of determining the apparent location of the Earth or moon and its apparent diameter. This information can then be used for autonomous onboard trajectory determination in cislunar space. Two methods of removing potential outliers are introduced in order to increase the algorithm robustness. Preliminary analysis of the algorithm's performance is studied via processing both real and synthetic lunar images.

Acknowledgments

Main text.

Author Contributions

Daniele Mortari has conceived and developed the original theory. Francesco de Dilectis bas validated the idea by developing software. Renato Zanetti has supervised in details theory and validation and has improved the use of English in the manuscript.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Chorym, M.A.; Hoffman, D.P.; Major, C.S.; and Spector, V.A. Autonomous Navigation - Where we are in 1984, *Guidance and Control*, Keyston CO, **1984**, 27-37.
2. Shoemaker, M.A.; Shroyer, L.E. Historical Trends in Ground-Based Optical Space Surveillance System Design, The Boeing Company.
3. Mortari, D. Trajectory Estimation using Earth and Moon Images, NASA-JSC Contract NNX13AF30A-S02, 01/01/13-09/30/13.
4. Mortari, D.; de Dilectis, F.; and Zanetti, R. Position Estimation using Image Derivative, AAS 14-259, 2015 *AAS/AIAA Space Flight Mechanics Meeting Conference*, Williamsburg, VA, Jan. 12-15, **2015**.
5. Fischler, M.A.; and Bolles, R.C. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography, *Comm. of the ACM*, **1981**, *24*, 381-395.
6. Mortari, D.; Samaan, M.A.; Bruccoleri, C.; and Junkins, J.L. The Pyramid Star Pattern Recognition Algorithm, ION *Navigation*, **2004**, *51 (3)*, 171-183.
7. Taubin, G. Estimation Of Planar Curves, Surfaces And Nonplanar Space Curves Defined By Implicit Equations, With Applications To Edge And Range Image Segmentation, IEEE *Trans. PAMI*, **1991**, *13*, 1115-1138.
8. Chernov, N. Circular and Linear Regression. Fitting Circles and Lines by Least Squares, *Monographs on Statistics and Applied Probability*, 2011, CRC Press, *117*.