AAS 13-778

TRAJECTORY DETERMINATION WITH UNKNOWN PERTURBATIONS

Francesco de Dilectis[*], Daniele Mortari[†], and Renato Zanetti[‡]

In this paper a new approach to estimate the trajectory of a spacecraft on a cislunar trajectory, based on measurements obtained with a camera pointing at the Moon, is discussed. The basic idea is to approximate the trajectory with a Bézier curve, whose control points can be found with a least square best fitting, and successively improve the accuracy of this curve by using the measurements. The effect of enforcing boundary conditions (initial and final position of the spacecraft) is also explored. This approach has the advantage of being independent from the problem's physics, which gives it great generality. For comparison, a Least Square Optimization method that effectively integrates the trajectory is also used. Main motivation is to make position estimation autonomous on Earth/Moon trajectories, to guarantee operativity in case of communications loss. While the image processing is the subject of the companion paper, this paper presents an analysis of the estimator observability using JPL-SPICE and GSFC-GMAT software for simulations. Estimation of Earth trajectory will also be provided using real sky-night Moon images.

INTRODUCTION

Since the beginning of modern astrodynamics, orbit determination has been one of the most interesting problems of the discipline. While for Keplerian orbits this problem can be considered fully solved, for 3 or more bodies a general solution has not been found, but rather a multitude of approximate or restricted solutions exist. Particularly in the case of the three-body problem, the special solutions due to Lagrange are well known. For higher number of bodies however, not even such solutions are available. It is therefore of interest to look for a method that can tackle all of these cases efficiently.

In Earth-Moon trajectories spacecraft are usually autonomous for attitude determination while the position estimation still relies on communication with ground bases. Rendering spacecrafts autonomous for positioning aswell is important, at least as a backup resource in case of lost communications.

The method proposed in this paper consists of an estimation algorithm that uses measurements obtained by data-processing of images taken by a visible camera observing the closest body. In Earth-Moon trajectories, this body can be the Earth or the Moon, depending on the distance. In

[*]PhD Graduate Student, 301B Reed McDonald, Aerospace Engineering, Texas A&M University, College Station, TX 77843-3141. E-mail: F.DE.DILECTIS@NEO.TAMU.EDU

[†]Professor, 746C H.R. Bright Bldg, Aerospace Engineering, Texas A&M University, College Station, TX 77843-3141, AAS Fellow, AIAA Associate Fellow. IEEE Senior member. E-mail: MORTARI@TAMU.EDU

[‡]Senior Member of the Technical Staff, Charles Stark Draper Laboratory, 17629 El Camino Real, Suite 470, Houston, TX 77058, AIAA Member. E-mail: RZANETTI@DRAPER.COM

fact, observation of the closest body implies greater variations of the observed parameters, thus improving state observability. A companion paper, Ref. [1], is dedicated to the image processing while this paper focuses on the trajectory estimation. This estimation can then be used as initial guess in a Kalman (or similar) more accurate filtering estimation process.

To simulate the real case of operative software, Earth and Moon positions and orientations are computed using SPICE with specific frequencies and stored in an on-board database. This database will be used to avoid the computational costs of obtaining high accuracy positions and attitudes of Earth and Moon. To obtain all data with the same level of accuracy, the selection of the sampling frequencies is chosen as a function of the repetition orbital and spinning periods. Interpolated positions are obtained using a quadratic Bézier function, while interpolated attitudes are obtained with a linear quaternion interpolation.

A simulated trajectory has been used to validate the estimation process, obtained using the GSFC-GMAT software, which is consistent with SPICE. All the estimation process is performed in Earth-centered J2000 inertial reference frame.

SIMULATION

To test the validity of the algorithm, a simulation environment has been created in MATLAB. This relies on two external sources for the necessary data; the information about planetary motion and orientation (namely of Earth and Moon) are generated with the SPICE toolkit developed by NAIF at JPL, and are used as "true" measurements, while the spacecraft trajectory is generated via the General Mission Analysis Tool (GMAT), developed by NASA. Both these programs compute the planetary motions using the same ephemeris database, compiled by NASA, which ensures the results will be consistent.

Generating the measurements with SPICE

The SPICE toolkit is an ancillary information system that provides scientists and engineers the capability to include space geometry and event data into mission design, science observation planning, and science data analysis software. It consists of a series of C++ routines accessible via a MATLAB interface; for the present paper, SPICE is used to generate a database of the following:

- position and velocity vectors of the Sun, in the J2000 reference frame centered on Earth ($\hat{\boldsymbol{S}}$).
- position and velocity vectors of the Moon, in the J2000 reference frame centered on Earth ($\hat{\boldsymbol{M}}$).
- orientation of the Moon, w.r.t. the J2000 reference frame, expressed as a quaternion ($\hat{\boldsymbol{q}}_M$).
- orientation of the Earth, w.r.t. the J2000 reference frame, expressed as a quaternion ($\hat{\boldsymbol{q}}_E$).

The database can cover any time interval up to the year 2050. Since characteristic periods of the celestial bodies of interest are very different, in order to save memory and computational time, data relative to Moon and Sun are generated with a time step multiple of the one used to sample Earth's quaternion. Specifically, the Sun's position and velocity are sampled with $t_s = 365\ t_e$ and the Moon's quaternion is sampled with $t_m = 27\ t_e$. The Moon's position and velocity are however sampled at the smallest time step, to ensure the highest precision possible. It is important to emphasize how the database is compiled off-line and accessed as requested during the mission.

To this end a separate routine has been designed to search the database for an entry associated with a specific Julian date and to return the requested data point. Whenever the entry is not readily available, it is generated with a proper interpolation process.

Time-linear Position Interpolation

The position vectors of Moon and Earth are stored in the mission database at discrete time steps. Image processing and trajectory estimation processes require estimation of position at any time. Therefore, data points not originally present in the database are interpolated, to obtain position and velocity vectors, $[\, E\{\boldsymbol{r}(t)\}, E\{\boldsymbol{v}(t)\}\,]$, at time t, using the knowledge of position and velocity vectors, $[\, \boldsymbol{r}_i, \boldsymbol{v}_i\,]$ and $[\, \boldsymbol{r}_f, \boldsymbol{v}_f\,]$, at times t_i and t_f, respectively, where $t_i \leq t \leq t_f$.

A time-linear quadratic nonrational Bézier curve is adopted to interpolate. This implies that position and time are evolving according to

$$\begin{cases} \boldsymbol{r} = \boldsymbol{r}_i\,(1-s)^2 + 2\boldsymbol{r}_1\,(1-s)s + \boldsymbol{r}_f\,s^2 \\ t = t_i\,(1-s) + t_f\,s \end{cases} \tag{1}$$

where s is the implicit parameter, and $s = 0$ for $t = t_i$ and $s = 1$ for $t = t_f$. The velocity can be written as

$$\boldsymbol{v} = \frac{d\boldsymbol{r}}{ds}\left(\frac{dt}{ds}\right)^{-1} = \frac{2}{t_f - t_i}\left[\boldsymbol{r}_i\,(s-1) + \boldsymbol{r}_1\,(1-2s) + \boldsymbol{r}_f\,s\right] \tag{2}$$

To be consistent with Eq. (2) the end velocities, $\boldsymbol{v}_i$ and $\boldsymbol{v}_f$, have the expressions

$$\boldsymbol{v}_i = \frac{2}{t_f - t_i}\,(\boldsymbol{r}_1 - \boldsymbol{r}_i) \qquad \text{and} \qquad \boldsymbol{v}_f = \frac{2}{t_f - t_i}\,(\boldsymbol{r}_f - \boldsymbol{r}_1) \tag{3}$$

from which we obtain an average estimation of the middle control point $\boldsymbol{r}_1$

$$\bar{\mathbf{r}}_1 = \frac{1}{2}\,(\boldsymbol{r}_i + \boldsymbol{r}_f) + \frac{t_f - t_i}{4}\,(\boldsymbol{v}_i - \boldsymbol{v}_f) \tag{4}$$

Therefore, the position and velocity estimation problem becomes

$$\begin{Bmatrix} \boldsymbol{r}(t) \\ \boldsymbol{v}(t) \end{Bmatrix} = \begin{Bmatrix} \boldsymbol{r}_i(1-s)^2 + 2\bar{\mathbf{r}}_1(1-s)s + \boldsymbol{r}_f s^2 \\ \dfrac{2}{t_f - t_i}[\boldsymbol{r}_i(1-s) + \bar{\mathbf{r}}_1(1-2s) + \boldsymbol{r}_f^2] \end{Bmatrix} \quad \text{where} \quad s = \frac{t - t_i}{t_f - t_i} \tag{5}$$

and where $\bar{\mathbf{r}}_1$ is provided by Eq. (4). It is worth nothing that while this interpolation procedure uses the same family of Bézier curves that will later be used to approximate the spacecraft's trajectory, its application here is completely independent from later discussion.

Time-linear Quaternion Interpolation

For the quaternion data the constant pure spin condition has been adopted. This is the equivalent of linear interpolation on the surface of a 4 dimensional unit-sphere. Given two quaternions $\hat{\boldsymbol{q}}_i$ and $\hat{\boldsymbol{q}}_f$, any intermediate quaternion can be found easily using:

$$\hat{\boldsymbol{q}}(t) = \frac{1}{\sin\alpha}\;(\hat{\boldsymbol{q}}_i\,\sin\alpha_i + \hat{\boldsymbol{q}}_f\,\sin\alpha_f) \tag{6}$$

where $\sin\alpha = \sqrt{1 - (\hat{\boldsymbol{q}}_i^{\mathrm{T}}\hat{\boldsymbol{q}}_f)^2}$, $\alpha_i = \dfrac{t_f - t}{t_f - t_i}\,\alpha$, and $\alpha_f = \dfrac{t - t_i}{t_f - t_i}\,\alpha$.

The tradeoff between a small, portable database and an accurate interpolation has been studied so to find the best compromise. To this end, several databases with different sample times have been generated and then interpolated to match a "standard" database with a sample time of 10 mins. Some of the results are illustrated in the following plots. The angle between "true" and interpolated values of Sun direction as been taken as a measure of the relative error, according to $\alpha = \cos^{-1}(\hat{\boldsymbol{s}}_i \cdot \hat{\boldsymbol{s}}_r)$. Similarly, the relative error between quaternions has been taken equal to $\phi = 2\cos^{-1}(\hat{\boldsymbol{q}}_i \cdot \hat{\boldsymbol{q}}_r)$. In the previous formulas, the subscript "r" indicates a reference value, and the subscript "i" indicated an interpolated value.

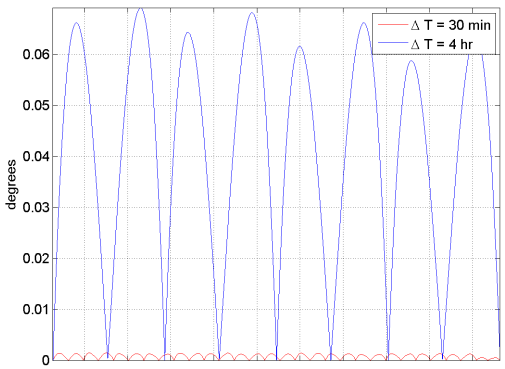


Figure 1. Sun direction error

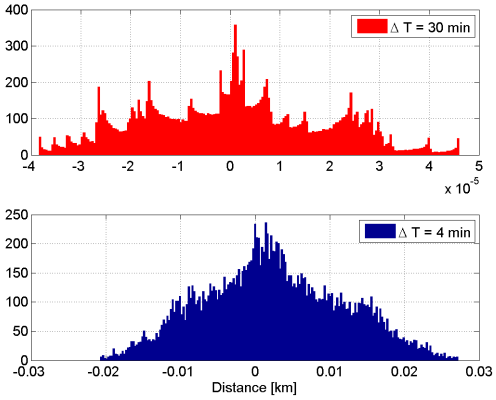


Figure 2. Moon position error

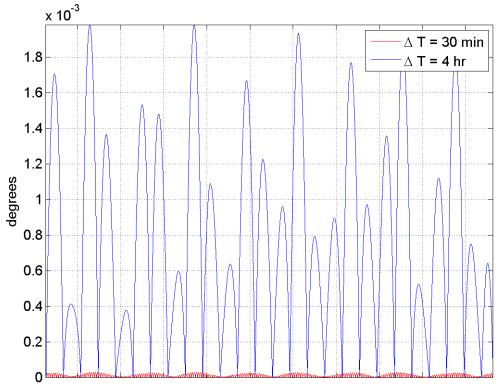


Figure 3. Moon quaternion error

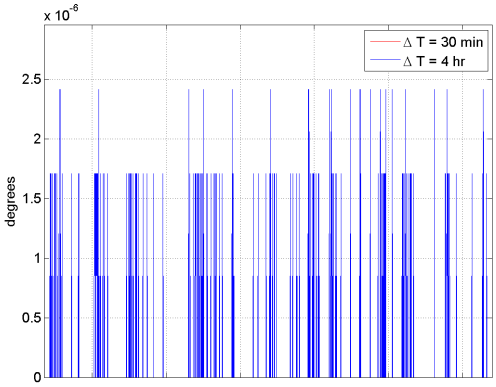


Figure 4. Earth quaternion error

Generating the trajectory with GMAT

GMAT propagators can take into account gravitational effects of higher order due to Earth, Moon and Sun, together with Solar Radiation Pressure. Output from GMAT can later be read by MATLAB and integrated with information provided by SPICE. As GMAT uses an optimized integration step, a manual procedure has been implemented to obtain data points equally spaced in time.

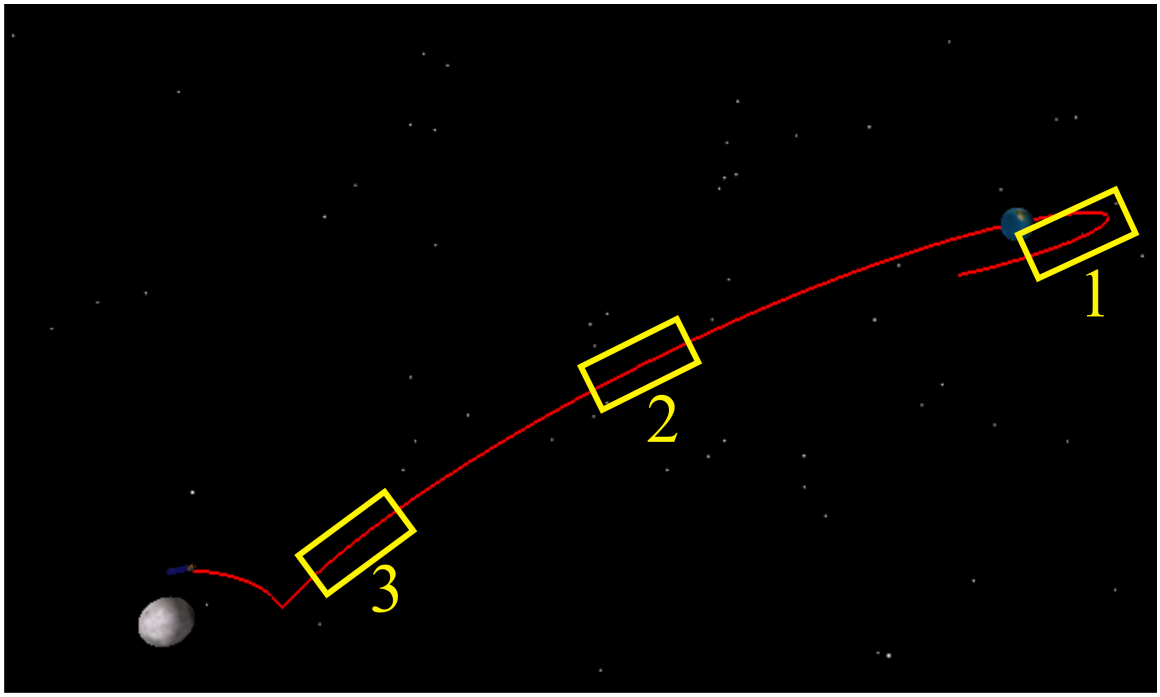


Figure 5. Earth-to-Moon trajectory simulated by GMAT. Circled in yellow are the segments used for the testing.

DATA ACCURACY ESTIMATION

According to SPICE, Moon is a sphere and the Earth an axial-symmetric ellipsoid. This means that the Moon's hard-edge contour is a circle while the Earth's one is an ellipse. The data obtained by image processing are:

Moon
- Center $[r, c]$ or unit-vector, $\hat{\boldsymbol{b}}$,
- Axis of symmetry, $\hat{\boldsymbol{s}}$,
- Radius, [R]

Earth
- Center $[r, c]$ or unit-vector, $\hat{\boldsymbol{b}}$,
- Semi-major axis, a,
- Ellipse orientation, ϑ.

Reference [1] describes the image processing to estimate the body center direction (corrected by the offset for the Earth), as well the radius (Moon) or the semi-major axis (Earth). These estimations have been performed by iterative least-squares using circular (Moon) and elliptical (Earth) sigmoid functions. To quantify how accurate they are, Monte Carlo tests can be done using image translations and rotations. This is implemented following Ref. [2], whose theory is summarized in the next two sub-sections.

Image translation

Consider a given image, for which the estimation of the parameters (e.g., Moon direction, $\hat{\boldsymbol{b}}_m$, and Moon radius, R_m) has been performed, as described in Ref. [1]. This is a single image and, in

order to quantify the standard deviations of the errors associated with these estimations we do need 1) the true value of these parameters and 2) statistical estimation of them under identical conditions, namely, same observer and observed positions! Obviously, this is not possible. Therefore, the accuracy must be computed by a different approach, which is described in these two subsections.

The body center coordinates and radius are provided at fraction of pixel level of accuracy. Let consider to translate this image by δr rows and δc columns, not necessarily an integer number of pixels. If the translation is accurate, then by applying the image processing to the translated image we obtain measurements whose values differ from the expected. Meaning, the direction to the body center should be estimated by δr and δc variations while the radius should be the same. The new estimated values differ from the expected because of: 1) the image processing precision (what we want to quantify) and 2) the image translation precision. If the image translation is performed with zero-mean noise, then, from statistical point of view, Monte Carlo tests performed using various values of δr and δc provide the image processing precision. Same idea stands for image rotation. Therefore, particular attention must be given to perform rotation and/or translation with zero-mean noise.

Therefore, a fraction of pixel continuous translation algorithm is adopted and explained here. The translation of an image by a non integer number of pixels can be performed in two steps: a translation of an integer number of pixels, followed by a translation of a fraction of pixel. The first operation is simple since the image is stored in a matrix; it is sufficient to move the matrix elements of the given integer number of rows and columns, while the case of fractional translation the problem is more complicate.

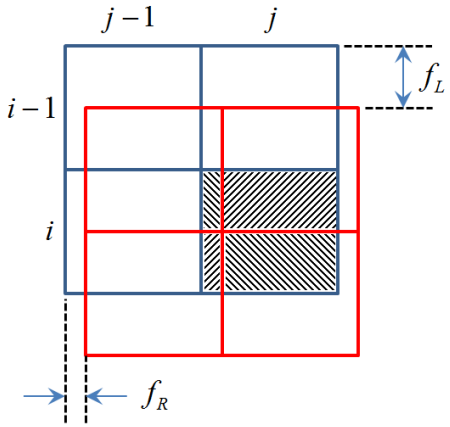


Figure 6. Translation model

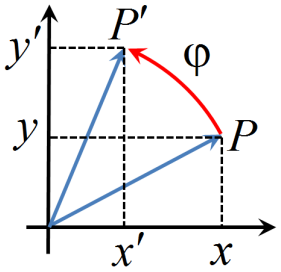


Figure 7. Rotation model

Let $-1 < f_R < 1$ and $-1 < f_L < 1$ be the fractional translations in row and column, respectively (see Fig. 6). In this case there is a partial overlap of four neighboring pixels over the "new" pixel (i, j); for positive translations, the overlapping pixels are $(i-1, j-1)$, $(i-1, j)$, $(i, j-1)$, and (i, j). The intensity of pixel (i, j), after translation, can be computed as a weighted sum of four contributions

$$\begin{aligned} M_n(i,j) \;=\; & M_o(i-1,j-1) f_R f_L + M_o(i-1,j)(1-f_R) f_L + \\ & + M_o(i,j-1) f_R (1-f_L) + M_o(i,j)(1-f_R)(1-f_L) \end{aligned} \tag{7}$$

where $M_n(i,j)$ and $M_o(i,j)$ are the gray tone of the (i,j) pixel in the translated and original images, respectively. Using Eq. (7) for every pixel in the original image, and rounding the results to the nearest gray levels, we obtain the translated image. A consequence of analog translation is a smoothing effect, due to the distribution of pixel intensities on the neighboring pixels. *This gray tone redistribution based on overlapping areas is exact if the gray tone level for each pixel is constant inside the pixel*. This is certainly a zero-mean approximation.

Image rotation

As for image rotation algorithm, let us consider the (x,y) pixel gray tone a lumped mass of weight $\mathbb{M}$ centered at the pixel center (position P in Fig. 7). Consider the rotation performed about the center $O(0,0)$ by the counterclockwise angle φ. Then, the (x,y) pixel mass is moved into the new position (x',y'). Then, we have the following formulae

$$\begin{cases} x' = x\cos\varphi - y\sin\varphi \\ y' = x\sin\varphi + y\cos\varphi \end{cases} \tag{8}$$

being (x,y) the coordinates of the pixel center P, with respect to O. The angle φ is assumed positive if counterclockwise.

The original pixel "affects" four pixels in the rotated image; Let (m,n), $(m+1,n)$, $(m,n+1)$, and $(m+1,n+1)$ be their coordinates, where $m = \lfloor x' \rfloor$, and $n = \lfloor y' \rfloor$, and where $\lfloor u \rfloor$ denotes the maximum integer less than u. The pixel "mass" $\mathbb{M}$ (i.e. the gray tone) is distributed among the four pixels, so that the center of gravity of the resulting four masses coincides with the rotated center P'. Let,

$$\begin{cases} \alpha = m + 1 - x' \\ \beta = n + 1 - y' \end{cases} \tag{9}$$

then, the new gray levels due to the rotation of P, are computed as follows. First, we distribute the mass M in the horizontal direction

$$\begin{cases} M_f(m+1,n+1) &=& M_f(m+1,n) = (1-\alpha)\mathbb{M} \\ M_f(m,n+1) &=& M_f(m,n) = \alpha\mathbb{M} \end{cases} \tag{10}$$

where M_f indicates the image after the horizontal distribution. Then, we distribute the mass in the vertical direction

$$\begin{cases} M_s(m,n+1) &=& M_f(m,n+1)(1-\beta) &=& \mathbb{M}\alpha(1-\beta) \\ M_s(m+1,n+1) &=& M_f(m+1,n+1)(1-\beta) &=& \mathbb{M}(1-\alpha)(1-\beta) \\ M_s(m,n) &=& M_f(m,n)\beta &=& \mathbb{M}\alpha\beta \\ M_s(m+1,n) &=& M_f(m+1,n)\beta &=& \mathbb{M}(1-\alpha)\beta \end{cases} \tag{11}$$

where M_s indicates the image matrix after the vertical distribution. The sum of the four masses must be preserved, $\mathbb{M}\alpha(1-\beta) + \mathbb{M}(1-\alpha)(1-\beta) + \mathbb{M}\alpha\beta + \mathbb{M}(1-\alpha)\beta = \mathbb{M}$. This validates the correctness of the above formulae (equal to the "mass" of the rotated pixel). The coordinates of the center of gravity of the four masses are

$$\begin{cases} x_g = m + 1 - \alpha = x' \\ y_g = n + 1 - \beta = y' \end{cases} \tag{12}$$

Adding the effects of every pixel in the original image, and rounding the results to the nearest gray levels, we obtain the rotated image.

Distance estimation error

Information about the distance to the observed body can be derived from the body radius evaluation. This is shown in Fig. 8, where r is the body observed radius (mm), f is the lens focal length (mm), D is the distance to the body (km), and R the body radius (km).

$$\frac{r}{f} = \frac{R}{\sqrt{D^2 - R^2}} \tag{13}$$

by performing the derivative it is possible to quantify the error associated with the distance estimation using the computed body radius

$$\frac{dr}{dD} = -\frac{f\,R\,D}{(D^2 - R^2)^{3/2}} \qquad \rightarrow \qquad \sigma_D^2 = \frac{(D^2 - R^2)^3}{f^2\,R^2\,D^2}\,\sigma_r^2 \tag{14}$$

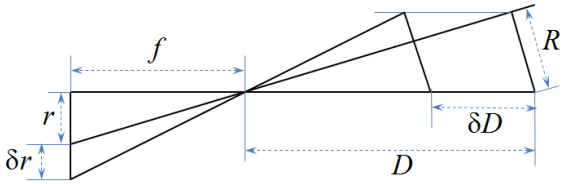


Figure 8. Distance estimation sensitivity geometry

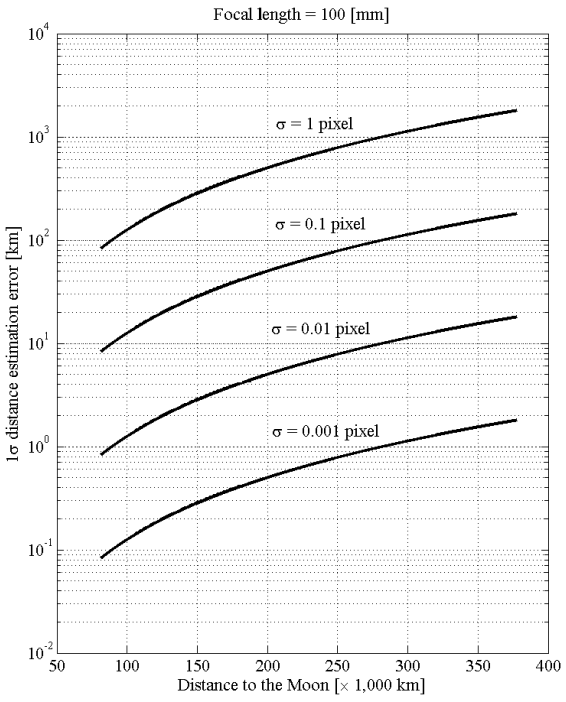


Figure 9. Distance estimation sensitivity for various radius estimation accuracy values

TRAJECTORY ESTIMATION USING NONRATIONAL BÉZIER FUNCTIONS

Gooding's IOD method [3] provides, in general, better performances than most alternative methods. This is because Gooding's method is based on the Lambert problem, which is a known *Two-Point Boundary Value Problem* (TPBVP), while most of the other methods can be considered falling into the *Initial Value Problem* (IVP) category. These two approaches have both different complexity and different obtained accuracy.

In general, an IVP is easy to solve but sometimes the solutions provided are not very accurate. This is because all the information is concentrated in the starting point*. Small variations in initial conditions (i.e. at the same point) may cause huge variations in the solution's accuracy or even chaos. On the other hand a TPBVP relies on information provided in two distinct points, initial and final. This makes them usually more accurate and, more importantly, less sensible to small perturbations of the initial data; on the other hand, they are harder to solve than an IVP.

Based on the above discussion, this paper investigates the use of Bézier curves to approximate trajectories because, in these curves, the information is also provided at intermediate points (i.e., control points) along the curve, and not only at the endpoints, as in TPBVP. This is expected to increase stability and accuracy of the solution.

In addition, simulate cislunar trajectories requires numerical integration of a multi-body problem under solar pressure perturbation which is difficult to model as it depends on the unknown spacecraft attitude and on the unknown solar activity. Note that, even if the solar pressure is in general a weak perturbation its contribution cannot be neglected due tot he typical travelling times on cislunar trajectories. On the other hand, this approach based on Bézier curves is completely independent by any dynamic consideration, which means it has not only a much lighter computational load, but it is also universally applicable to any type of trajectory, as long as a sufficient number of measurements is known. In the following we investigate the capability to estimate a cislunar trajectory with "B_{ij}"-curves, where "i" is the degree of the Bézier curve used to describe the position vectors and "j" is the degree of the Bézier curve used to describe the time.

Trajectory Best Fitting using Nonrational Bézier Functions

A nonrational Bézier function can be written as linear combination of Bernstein polynomials,

$$\boldsymbol{r} = \sum_{k=0}^{n} \boldsymbol{c}_k B_k^n(s) \quad \text{where} \quad B_k^n(s) = \binom{n}{k} s^k (1-s)^{n-k}, \quad s \in [0,1]. \tag{15}$$

By using the binomial theorem, a generic Bernstein polynomial can be rewritten:

$$B_k^n(s) = \binom{n}{k} s^k \left[\sum_{j=0}^{n-k} \binom{n-k}{j} (-s)^{n-(k+j)} \right] = \binom{n}{k} \left[\sum_{j=0}^{n-k} \binom{n-k}{j} (-1)^{n-(k+j)} s^{n-j} \right]. \tag{16}$$

Therefore, any nonrational Bézier function has an expression of the form:

$$\boldsymbol{r} = \sum_{k=0}^{n} \left[\boldsymbol{c}_k \binom{n}{k} \left[\sum_{j=0}^{n-k} \binom{n-k}{j} (-1)^{n-(k+j)} s^{n-j} \right] \right]. \tag{17}$$

*A similar criticism can be addressed to the least-squares problem of fitting data points using truncated Taylor series.

Given m points along a curve and the corresponding values of the parameter s Eq. (17) can be rewritten in a more compact form using linear algebra:

$$\mathbb{R} = \mathbb{C}\,\mathbb{M}\,\mathbb{S} \tag{18}$$

where

$$\begin{cases} \mathbb{R} = [\,\boldsymbol{r}_1,\ \boldsymbol{r}_2,\ \ldots,\ \boldsymbol{r}_m\,] \\ \\ \mathbb{C} = [\,\boldsymbol{c}_0,\ \boldsymbol{c}_1,\ \ldots,\ \boldsymbol{c}_n\,] \end{cases} \quad \text{and} \quad \mathbb{S} = \begin{bmatrix} s_1^n & s_2^n & \cdots & s_{m-1}^n & s_m^n \\ s_1^{n-1} & s_2^{n-1} & \cdots & s_{m-1}^{n-1} & s_m^{n-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ s_1 & s_2 & \cdots & s_{m-1} & s_m \\ 1 & 1 & \cdots & 1 & 1 \end{bmatrix} \tag{19}$$

and the elements of matrix $\mathbb{M}$ are of the form $\mathbb{M}(k,j) = \binom{n}{k}\binom{n-k}{j}(-1)^{n-(k+j)}$, $j = 1, \ldots, n-1$. It follows from this formula that $\mathbb{M}(k,j) = \mathbb{M}(j,k)$, i.e. $\mathbb{M}$ is a symmetric matrix, and also $\mathbb{M}(k,j) = 0$ whenever $k + j > n + 1$. Thus, for instance for a quadratic Bézier polynomial the expressions of $\mathbb{M}$ and $\mathbb{M}^{-1}$ are

$$\mathbb{M}_2 = \begin{bmatrix} 1 & -2 & 1 \\ -2 & 2 & 0 \\ 1 & 0 & 0 \end{bmatrix} \qquad \text{and} \qquad \mathbb{M}_2^{-1} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1/2 & 1 \\ 1 & 1 & 1 \end{bmatrix}, \tag{20}$$

while for a cubic polynomial

$$\mathbb{M}_3 = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \qquad \text{and} \qquad \mathbb{M}_3^{-1} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1/3 & 1 \\ 0 & 1/3 & 2/3 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \tag{21}$$

From Eq. (17) the control points can be estimated with a Least Squares approach:

$$\mathbb{C} = \mathbb{R}\,\mathbb{S}^{\mathrm{T}}\,(\mathbb{S}\mathbb{S}^{\mathrm{T}})^{-1}\,\mathbb{M}^{-1} \tag{22}$$

Distribution of the parameter

To solve Eq. (22), a distribution of the parameter s corresponding to the measurements has to be assumed. If the data is provided at constant time step, uniform distribution can be used:

$$s_k = \frac{k-1}{n-1} \qquad \text{where} \qquad k \in [1, n]. \tag{23}$$

Another valid approximation relates the distribution with the relative distance of subsequent data points, as per the following:

$$\begin{cases} s_1 = 0 \\ s_m = 1 \end{cases} \qquad \text{and} \qquad s_k = \frac{\displaystyle\sum_{j=2}^{k} |\boldsymbol{r}_j - \boldsymbol{r}_{j-1}|}{\displaystyle\sum_{j=2}^{m} |\boldsymbol{r}_j - \boldsymbol{r}_{j-1}|} \quad \text{where} \quad k \in [2, m-1]. \tag{24}$$

In general, however, solving Eq. (22) with either distribution will lead to points displaced from the actual measurements, by a value of:

$$d_k = |\mathbb{C}\mathbb{M}\boldsymbol{s}_k - \boldsymbol{r}_k| \quad k \in [1, m]. \tag{25}$$

where $\boldsymbol{s}_k$ is a vector of all the powers of s_k. Obviously, changing a certain s_k will only affect the value of the corresponding d_k. Thus, it is possible to find a new set of parameters by minimizing an aptly chosen cost function L_k, point by point:

$$L_k = (\mathbb{C}\mathbb{M}\boldsymbol{s}_k - \boldsymbol{r}_k)^{\mathrm{T}}(\mathbb{C}\mathbb{M}\boldsymbol{s}_k - \boldsymbol{r}_k) \tag{26}$$

$$L_k = \boldsymbol{s}_k^{\mathrm{T}}\mathbb{M}\mathbb{C}^{\mathrm{T}}\mathbb{C}\mathbb{M}\boldsymbol{s}_k - 2\boldsymbol{r}_k^{\mathrm{T}}\mathbb{C}\,\mathbb{M}\boldsymbol{s}_k + \boldsymbol{r}_k^{\mathrm{T}}\boldsymbol{r}_k \tag{27}$$

By applying the necessary condition for a stationary point:

$$\frac{dL_k}{ds_k} = \frac{dL_k}{d\boldsymbol{s}_k} \cdot \frac{d\boldsymbol{s}_k}{ds_k} = 2\,\boldsymbol{s}_k^{\mathrm{T}}\mathbb{M}\mathbb{C}^{\mathrm{T}}\mathbb{C}\mathbb{M}\frac{d\boldsymbol{s}_k}{ds_k} - 2\boldsymbol{r}_k^{\mathrm{T}}\mathbb{C}\,\mathbb{M}\frac{d\boldsymbol{s}_k}{ds_k} = 0 \tag{28}$$

which implies

$$F(s_k) = \boldsymbol{s}_k^{\mathrm{T}}\mathbb{M}\mathbb{C}^{\mathrm{T}}\mathbb{C}\mathbb{M}\frac{d\boldsymbol{s}_k}{ds_k} - \boldsymbol{r}_k^{\mathrm{T}}\mathbb{C}\,\mathbb{M}\frac{d\boldsymbol{s}_k}{ds_k} = 0 \tag{29}$$

Eq. (29) is a polynomial in s which can be solved for instance via Newton-Raphson method, given an initial guess $\bar{s}_k$. Indeed:

$$s_k^* = \bar{s}_k - \frac{F(\bar{s}_k)}{F'(\bar{s}_k)} \tag{30}$$

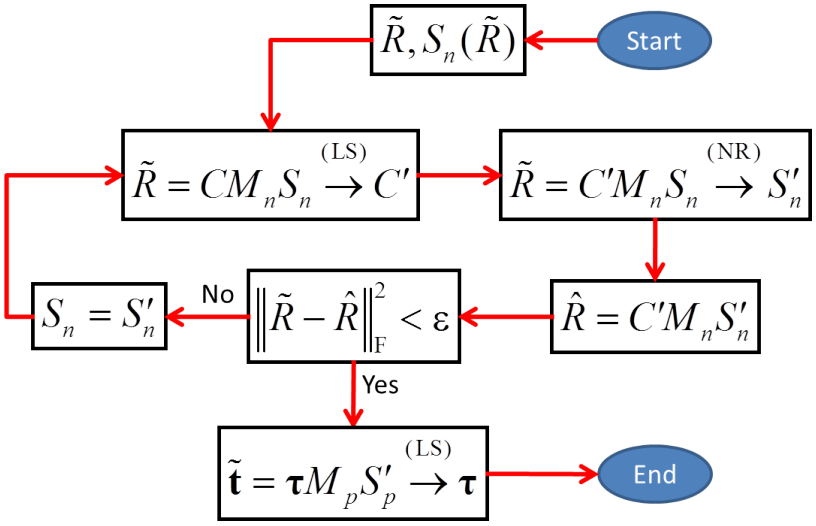
where

$$F'(s_k) = \frac{dF(s_k)}{ds_k} = \frac{d\boldsymbol{s}_k^{\mathrm{T}}}{ds_k}\mathbb{M}\mathbb{C}^{\mathrm{T}}\mathbb{C}\mathbb{M}\frac{d\boldsymbol{s}_k}{ds_k} + \boldsymbol{s}_k^{\mathrm{T}}\mathbb{M}\mathbb{C}^{\mathrm{T}}\mathbb{C}\mathbb{M}\frac{d^2\boldsymbol{s}_k}{ds_k^2} - \boldsymbol{r}_k^{\mathrm{T}}\mathbb{C}\,\mathbb{M}\frac{d^2\boldsymbol{s}_k}{ds_k^2} \tag{31}$$

In Eq. (25 - 31), the vector $\boldsymbol{s}_k$ and its derivatives have the following expressions:

$$\boldsymbol{s}_k = \begin{Bmatrix} s_k^n \\ s_k^{n-1} \\ \vdots \\ s_k \\ 1 \end{Bmatrix} \qquad \frac{d\boldsymbol{s}_k}{ds_k} = \begin{Bmatrix} n s_k^{n-1} \\ (n-1)s_k^{n-2} \\ \vdots \\ 1 \\ 0 \end{Bmatrix} \qquad \frac{d^2\boldsymbol{s}_k}{ds_k^2} = \begin{Bmatrix} n(n-1)s_k^{n-2} \\ (n-1)(n-2)s_k^{n-3} \\ \vdots \\ 0 \\ 0 \end{Bmatrix} \tag{32}$$

Minimizing Eq. (26) independently for each $k \in [2, m-1]$ (the first and last point are left unchanged, to ensure the parameter always spans the range $[0, 1]$), leads to a new parameter distribution, that can in turn be used in Eq. (22) to find a new set of control points. This process can be iterated multiple times until a certain convergence criteria has been met, for instance a measure of the distance between measured and estimated trajectory points. Schematically:



Choosing the optimal degree

Depending on the shape of the actual trajectory, Bézier curves of different degree provide different levels of approximation. Although an explicit rule has not been found, and could be subject of further research, in general trajectories of higher curvature require polynomials of higher degree. On the other hand, the maximum degree usable for a given segment is limited by the number of data points available, namely:

$$n = \left\lceil \frac{m}{3} - 1 \right\rceil . \tag{33}$$

If n doesn't satisfy this relation, Eq. (22) cannot be properly defined. Moreover, as n increases the matrix $\mathbb{S}\mathbb{S}^{\mathrm{T}}$ becomes ill-conditioned and numerical issues arise. An exploratory analysis has been conducted to find the optimal degree for each of the segments, using the mean error along the trajectory as a performance index. Results are illustrated in Fig. (10), (11) and (12).

The first segment(Fig. (10)), having the greatest curvature, is better estimated with a higher degree. This result persists when noisy measurements are taken into account. Segments 2 and 3 (Figg. (11) and (12)) have a much less pronounced curvature and therefore are best approximated by polynomials of 5^{th} and 6^{th} degree respectively. However, when considering noise, the results are dramatically different; this is due to the curvature being so small that it is completely masked by the noise.

Non-linear Time Best Fitting

In the previous discussion, while the spacecraft trajectory was assumed to be a Bézier curve of any degree, the relationship between time t and parameter s has been considered linear, in the form $t = t_i\,(1 - s) + t_f\,s$, which can be seen as a Bézier polynomial of degree 1 . This is a reasonable choice when measurements have been taken evenly spaced in time. However, non-linear time models better accommodate velocity variations along the trajectory. Therefore, higher degree curves can be used to describe time, and not necessarily of the same degree used for the trajectory. In principle, the two curves exist in different dimensional spaces, with the "time curve" being effectively one dimensional. However, they still are related via the parameter s, because each measurement is associated with a time instant. Therefore, upon reaching convergence in the

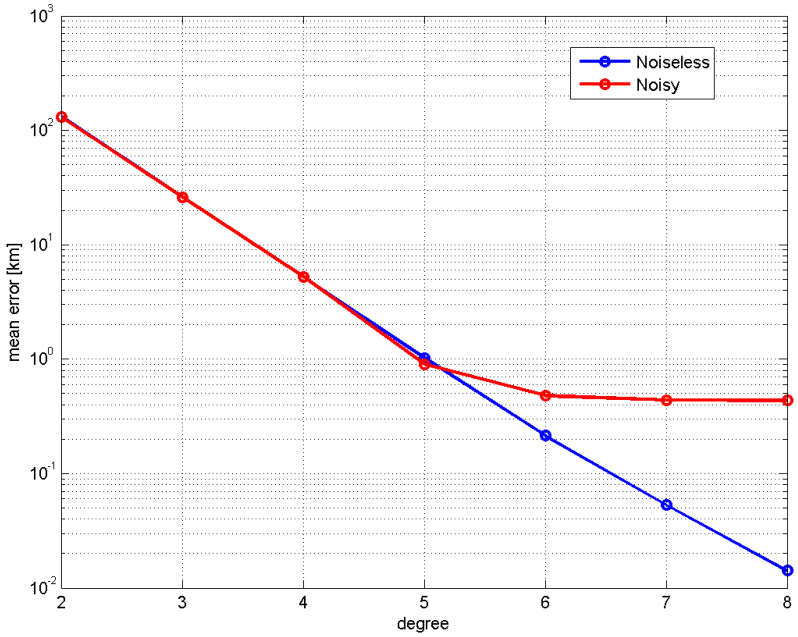


Figure 10. Approximation errors of Bézier curves of different degrees for the first segment of the trajectory

estimation of the $\boldsymbol{s}_k$, the control points τ for the "time curve" are easily found by applying the same procedure as above:

$$\tilde{\boldsymbol{t}} = \tau \mathbb{M}_p \mathbb{S}_p \quad \rightarrow \quad \tau = \tilde{\boldsymbol{t}}\, \mathbb{S}_p^{\mathrm{T}} \left(\mathbb{S}_p \mathbb{S}_p^{\mathrm{T}}\right)^{-1} \mathbb{M}_p^{-1} \tag{34}$$

where the matrix $\mathbb{S}_p$ contains all the powers of the parameter values and the subscript p indicates the degree of the polynomial, which can be progressively increased until the distance between the measured and estimated times is minimal:

$$\min_p \; |\tilde{\boldsymbol{t}} - \hat{\boldsymbol{t}}|^2 \tag{35}$$

and therefore find an optimal degree for the "time curve".

Because of the monotonic behavior of time, these control points must satisfy the inequality

$$t_i < \tau_1 < \cdots < \tau_{n-1} < t_f. \tag{36}$$

Estimator model	B_{21}	B_{22}	B_{31}	B_{32}	B_{33}	$\ldots$	Bij
Number of variables	9	10	12	13	14	$\ldots$	$3i + j - 2$

Table 1. Number of optimization variables of Bézier trajectory estimators

It is worth emphasizing that any of the τ_k certainly lie between t_i and t_f and are expressed in seconds, but are not times associated with any specific point of the trajectory. These are not "times" but are *control points* (expressed in seconds) associated with the time description of the trajectory.

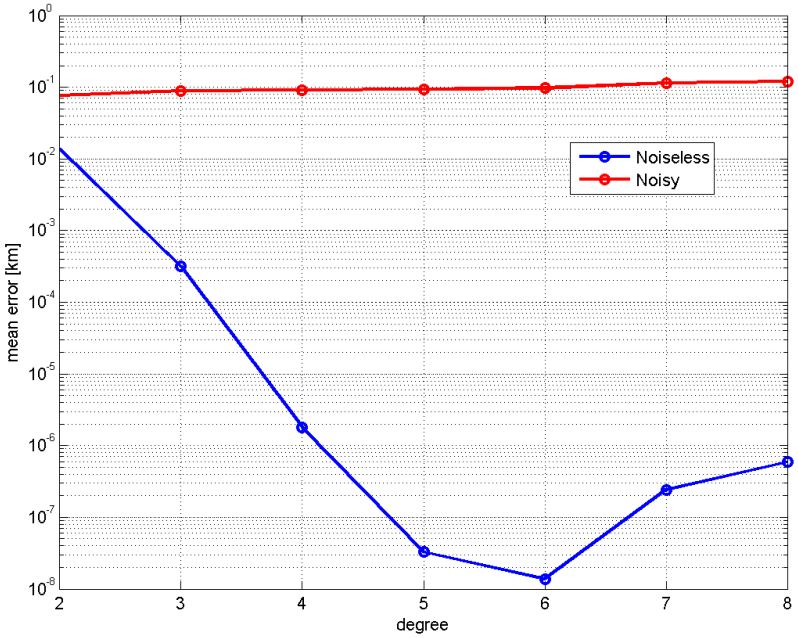


Figure 11. Approximation errors of Bézier curves of different degrees for the second segment of the trajectory

Time-varying Bézier Best Fitting

Another way of dealing with non-linear time, is to extend nonrational Bézier function to include times, as follows

$$\underset{4\times m}{\mathbb{X}} = \begin{bmatrix} \boldsymbol{r}_1 & \boldsymbol{r}_2 & \cdots & \boldsymbol{r}_{m-1} & \boldsymbol{r}_m \\ t_1 & t_2 & \cdots & t_{m-1} & t_m \end{bmatrix} = \underset{4\times(n+1)}{\mathbb{C}} \; \underset{(n+1)\times(n+1)}{\mathbb{M}} \; \underset{(n+1)\times m}{\mathbb{S}} \tag{37}$$

where

$$\underset{(n+1)\times m}{\mathbb{S}} = \begin{bmatrix} 0 & s_2^n & s_3^n & \cdots & s_{m-2}^n & s_{m-1}^n & 1 \\ 0 & s_2^{n-1} & s_3^{n-1} & \cdots & s_{m-2}^{n-1} & s_{m-1}^{n-1} & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & s_2 & s_3 & \cdots & s_{m-2} & s_{m-1} & 1 \\ 1 & 1 & 1 & \cdots & 1 & 1 & 1 \end{bmatrix} \tag{38}$$

and (note that here $0^0 = 1$) [†]

$$\underset{4\times(n+1)}{\mathbb{C}} = \begin{bmatrix} \boldsymbol{c}_0 & \boldsymbol{c}_1 & \cdots & \boldsymbol{c}_{n-1} & \boldsymbol{c}_n \\ \tau_0 & \tau_1 & \cdots & \tau_{n-1} & \tau_n \end{bmatrix} \qquad \text{where} \qquad \begin{cases} \tau_0 = t_1 \\ \tau_n = t_m \end{cases} \tag{39}$$

Equation (37) consists of $(4m-2)$ scalar equations, while the unknowns are: $3(n+1)$ for the position control points, $(n-1)$ for the time control points, $(m-2)$ for the Bézier parameters. Therefore, the Bézier best fitting problem admits solution if $4m - 2 \geq 4n + m$, that is, if

$$m \geq \left\lceil \frac{4n+2}{3} \right\rceil \tag{40}$$

[†]Note that, since $s_1 = 0$ and $s_m = 1$, the two identities, $\tau_0 = t_1$ and $\tau_n = t_m$, come out automatically.

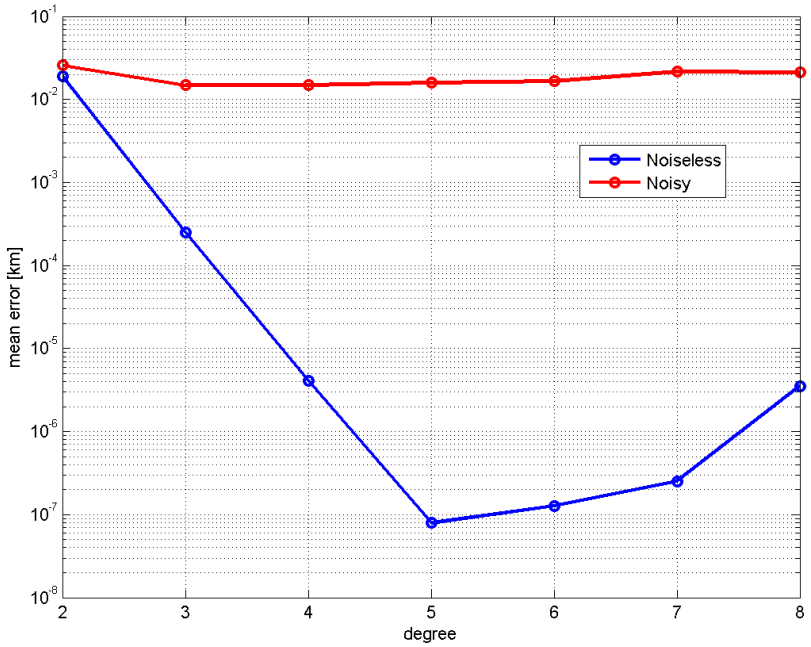


Figure 12. Approximation errors of Bézier curves of different degrees for the third segment of the trajectory

This means that for a quadratic ($n = 2$), $m \geq 4$ points are needed, for a cubic $m \geq 5$, for a quartic $m \geq 6$, for a quintic $m \geq 8$, etc. With this approach, however, the "time curve" is forced to be the same degree as the trajectory curve, which might be suboptimal, depending on the specific problem.

Iterative Batch Least-Squares

The standard batch least-squares orbit determination technique [5] is used as a mean to assess the performance of the proposed method. Rather then processing the Moon direction and Radius separately, a single derived measurement is obtained. From the apparent Moon radius and the focal length, the "measured" distance from the Moon ρ_k is deduced. Then a measurement $\tilde{\boldsymbol{y}}_k$ of the vehicle position from the center of the Moon is calculated

$$\tilde{\boldsymbol{y}}_k = f \, \frac{R_{\leftmoon}}{\tilde{R}_k} \, C_k^{\mathrm{T}} \, \tilde{\boldsymbol{b}}_k \tag{41}$$

where $R_{\leftmoon}$ is the Moon radius, f is the camera focal length, and C_k is the J2000-to-camera attitude.

To obtain the batch least-squares solution we start from an initial estimate $\hat{\boldsymbol{x}}_0^{\mathrm{T}} = \{ \hat{\boldsymbol{r}}_0^{\mathrm{T}}, \hat{\boldsymbol{v}}_0^{\mathrm{T}} \}$ and propagate it forward to each measurement time using Earth and Moon's central gravities. Together with the state we integrate the state transition matrix $\Phi(t_k, t_0)$ that takes linearized state deviations from the time of the initial estimate to the time of the k-th measurement. At each measurement time we accumulate the current observation:

$$\tilde{H}_k = \begin{bmatrix} I_{3\times3} & 0_{3\times3} \end{bmatrix} \tag{42}$$

$$H_k = \Phi(t_k, t_0) \tag{43}$$

$$\Lambda = \Lambda + H_k^{\mathrm{T}} H_k \tag{44}$$

$$N = N + H_k^{\mathrm{T}} \left[\tilde{\boldsymbol{y}}_k - (\boldsymbol{r}_{mk} - \hat{\boldsymbol{r}}_k) \right] \tag{45}$$

where Λ and N are initialized at zero, $\tilde{H}_k$ is the partial of the measurement with respect to the current state, H_k is the partial of measurement with respect to the initial state, $\hat{\boldsymbol{r}}_k$ is the estimated position of the vehicle with respect to the center of Earth and $\boldsymbol{r}_{mk}$ is the position of the Moon also with respect to Earth.

After all measurements are accumulated the initial estimate is updated as:

$$\hat{\boldsymbol{x}}_0 = \hat{\boldsymbol{x}}_0 + \Lambda^{-1} N \tag{46}$$

This procedure can be iterated multiple times.

Least-squares Trajectory Estimate using Bézier Function

The direction to the Moon center, $\tilde{\boldsymbol{b}}_k$ (in camera reference frame), and the apparent Moon radius, $\tilde{R}_k$, evaluated at time t_k, are measured from image processing. These, together with Eq. (13), can be combined to obtain the satellite-to-Moon vector as in Eq. (41). This vector is the difference between the the Earth-to-Moon vector, $\boldsymbol{r}_{Mk}$, and the satellite position estimated by the Bézier function, $\boldsymbol{r}_k(\boldsymbol{c}, s_k)$. Therefore, the trajectory estimation using Bézier functions consists of finding the set of $n+1$ control points that minimizes the cost function involving all m measurements

$$L = \frac{1}{2} \sum_k^m \sigma_k^{-2} (\tilde{\boldsymbol{y}}_k - \hat{\boldsymbol{y}}_k)^2 = \frac{1}{2} \sum_k^m \sigma_k^{-2} \left[\tilde{\boldsymbol{y}}_k - \boldsymbol{r}_{Mk} + \boldsymbol{r}_k(\boldsymbol{c}, s_k)\right]^2 . \tag{47}$$

where σ_k^{-2} is the inverse variance of the measurement vector, $\tilde{\boldsymbol{y}}_k$, which is estimated by translation and rotation as explained in the previous section. Since $\boldsymbol{r}_k(\boldsymbol{c}, s_k)$ is linear in the control points, the minimization problem consists of $n+1$ equations, $\partial L / \partial \boldsymbol{c}_i = 0$, where $i = 0, \cdots, n$

$$\sum_k^m \sigma_k^{-2} \left[\tilde{\boldsymbol{y}}_k - \boldsymbol{r}_{Mk} + \boldsymbol{r}_k(\boldsymbol{c}, s_k)\right] \frac{\partial \boldsymbol{r}_k}{\partial \boldsymbol{c}_i} = 0, \tag{48}$$

where

$$\frac{\partial \boldsymbol{r}_k}{\partial \boldsymbol{c}_i} = \binom{n}{i} s_k^i \left(1 - s_k\right)^{n-i}. \tag{49}$$

Numerical results

The following results have been obtained by taking segments on a trajectory between Earth and Moon generated with GMAT. The performance of the method described in this paper is compared with an Iterative Batch Least Square which propagates the equations of motion of a three-body problem.

CONCLUSIONS

Bézier curves appear to be a valid model to approximate cislunar trajectories. As seen in Fig. (13) to Fig. (15), the average error along the trajectory is lower or comparable with the Iterative Least Square method. However, there are three major advantages Bézier functions have over the Iterative LS method:

- While the Iterative LS requires an "initial guess", which can affect the convergence, Bézier curves only need a sufficiently large number of measurements to generate a consistent solution.

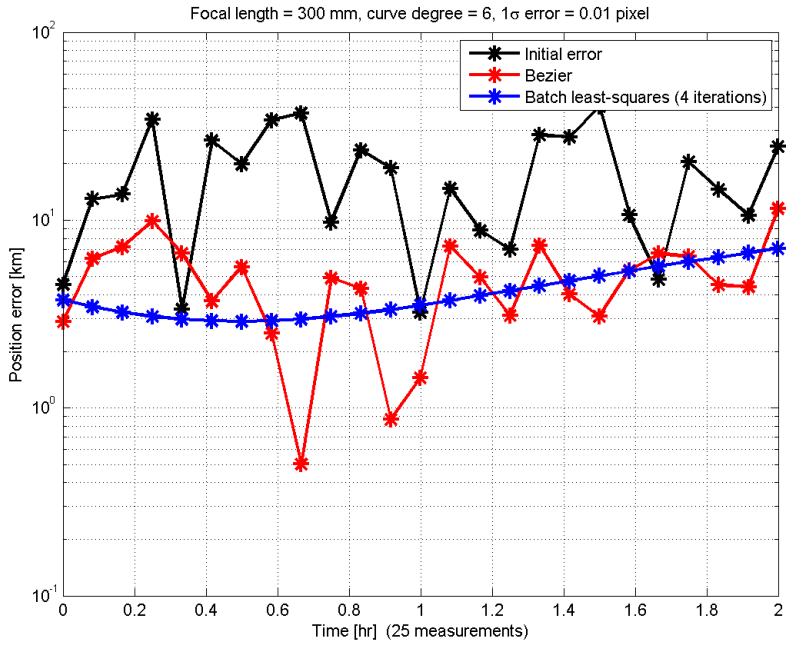


Figure 13. Comparison between Batch Least Square and Bézier curves. First trajectory segment, n = 6

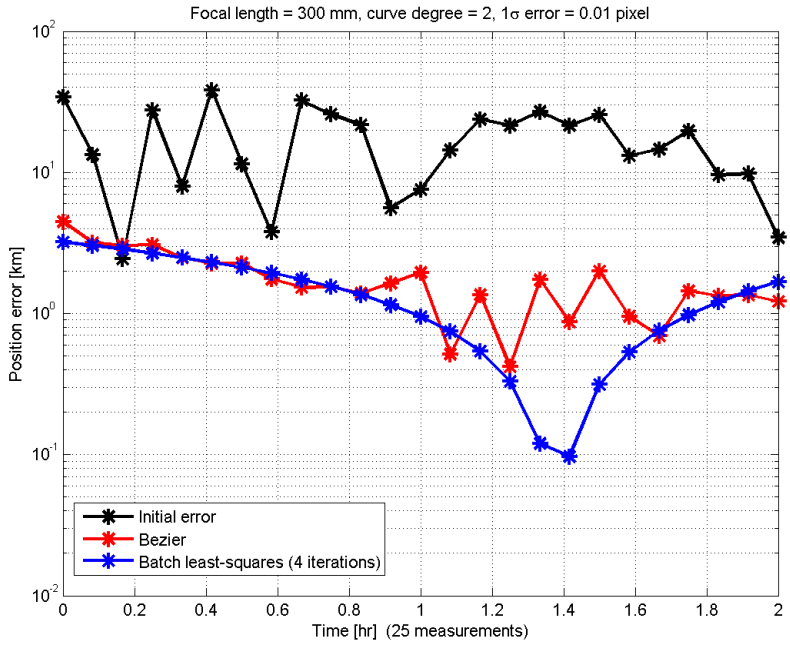


Figure 14. Comparison between Batch Least Square and Bézier curves. Second trajectory segment, n = 2

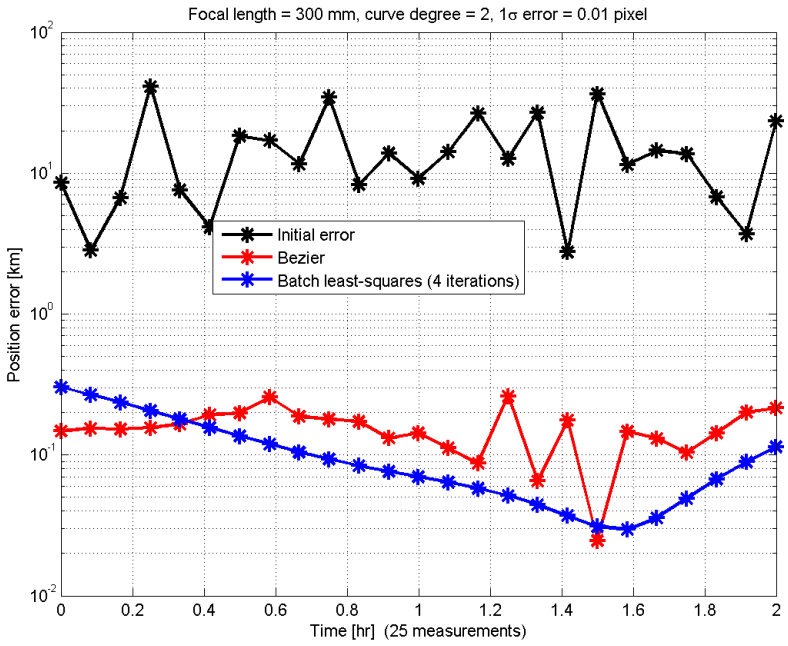


Figure 15. Comparison between Batch Least Square and Bézier curves. Third trajectory segment, n = 2

- The method proposed in the paper has a much lower computational cost than an Iterative Least Square. This is because the only computationally intensive operation is the inversion of a matrix of integers, namely $\mathbb{M}$, which needs to be done only once even if the whole procedure is iterative. Also, the convergence of the Newton-Raphson method to find the optimal parameter distribution has been extremely fast in all the simulations.

- Lastly, and most important, Bézier curves don't need any knowledge of the physic laws affecting the system, which not only means at no point in the computation process it is necessary to introduce equations of motion of any kind, but also this method is completely generic and can be potentially used to estimate trajectories in different class of problems, for instance interplanetary missions or re-entry trajectories.

REFERENCES

[1] Mortari, D., de Dilectis, F., and D'Souza, C. "Image Processing of Illuminated Ellipsoid," 2013 AAS/AIAA Astrodynamics Specialist Conference, Hilton Head, SC, August 11–15, 2013.

[2] Park, K.J. and Mortari, D. "Planet or Moon Image Processing for Spacecraft Attitude Estimation," *Journal of Electronic Imaging*, Vol. 17, No. 2, April-June 2008, pp. 1-11.

[3] Gooding, R. H. "A New Procedure for the Solution for the Classical Problem of Minimal Orbit Determination from Three Lines of Sight," *Celestial Mechanics and Dynamical Astronomy*. Vol. 66, No. 1, 1997, pp. 387-423.

[4] Pastva, T.A. "Bézier Curve Fitting," *Naval Postgraduate School Thesis*. September 1998, Monterey, CA.

[5] Byron D. Tapley et al., *Statistical Orbit Determination*, Elsevier Academic Press, 2004.