JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 31, No. 4, July–August 2008

Adaptive Gravitational Force Representation for Fast Trajectory Propagation Near Small Bodies

Andrew Colombi* and Anil N. Hirani†
University of Illinois at Urbana–Champaign, Urbana, Illinois 61801
and
Benjamin F. Villac‡
University of California, Irvine, Irvine, California 92697-3975

DOI: 10.2514/1.32559

We present an approximation scheme for gravitational forces near arbitrarily shaped small bodies. The approximation uses polynomial interpolation with an adaptive spatial data structure near the asteroid and spherical harmonics far from it. These data structures allow us to drive the approximation errors of the model to within user-defined thresholds, while significantly reducing the run time of trajectory integrations about small bodies. This alleviates the computational burden of Monte Carlo simulation for small-body proximity operation and mission design. We conclude with performance tests and models for the asteroid 1998 ML 14.

I. Introduction

IN RECENT years our solar system's small bodies (e.g., asteroids and comets) have received increased attention [1–3]. Whereas early asteroid missions represented gravitation with spherical harmonics [3], the goals of current and future small-body missions focus on close-proximity operations in which this representation is no longer valid [4]. To address this problem we may turn to the approach introduced by Werner [5] and Werner and Scheeres [4]; however, this method performs a summation over every face and edge of a polyhedral model representing the surface of the body. Such calculation is tractable over a few evaluations, but becomes computationally expensive for numerical integration. For example, consider the large sets of trajectories necessary for Monte Carlo simulations: each simulation contains hundreds of trajectories, each of which consists of thousands of elementary time steps, each of which relies on multiple force evaluations. This computational bottleneck in design and analysis of small-body missions has led researchers to investigate ways to reduce the cost of force calculation [6,7].

In this research, we propose to alleviate this burden by precomputing the gravitational forces throughout the domain, reducing future queries to a sublinear look up and constant time interpolation operation. This results in 2 orders of magnitude speed improvement when compared with current methods. Indeed, given the availability of cheap, fast memory storage, trading memory against online computation seems advantageous.

The idea of locally representing gravitational fields using interpolation has already been investigated by Junkins [8] and Engels and Junkins [9] in the context of inertial navigation around Earth. These methods, however, rely on the nearly spherical shape of the Earth, and do not transfer directly to the case of arbitrarily shaped bodies. To address this, we use an adaptive local representation that is applicable to general cases and is well suited for fast trajectory integration.

Before presenting these results, we review two computational models relevant to the formulation of our method: the polyhedral method of Werner and Scheeres [4] and the interpolation method of Engels and Junkins [9]. In Sec. III, we analyze the difficulties of interpolating gravitational forces near small bodies, and Sec. IV presents a solution to these difficulties. Secs. V and VI give a discussion of the performance and limitations of the proposed approach.

II. Previous Work

This section briefly reviews key features of two gravitational force representations that are relevant to this research: the polyhedral method developed by Werner [5] and Werner and Scheeres [4] and the interpolation method proposed by Junkins [8] and Engels and Junkins [9].

A. Polyhedral Methods

Polyhedral methods are widely used in mission analysis and planning when one needs to estimate the dynamical environment near the surface of a small body. These methods calculate gravitational force by performing an exact integration over a constant density approximation to the body, which is represented by a polyhedron [4]. Although the assumption of constant density is certainly a limitation, it is not required by the method that we shall develop; thus, it can be assumed in the remainder of the paper. This allows us to define the true gravitational force as that computed via the polyhedral method.

The gist of this method is to transform the volume integral defining gravitational force (or potential) into more tractable quantities. The volume integral for force is

$$\mathbf{F}(\mathbf{x}) = -G\rho_0 \int_{\text{asteroid}} \frac{\mathbf{x}-\mathbf{y}}{\|\mathbf{x}-\mathbf{y}\|^3}\, \mathrm{d}\mathbf{y}$$

where G is the universal gravitational constant, and ρ_0 is the constant density of the asteroid. Specifically, through application of Green's theorem, the above three-dimensional volume integral is transformed into a two-dimensional surface integral. By approximating the surface with a polyhedron, the integral can be further reduced to a summation over faces and edges, where each term requires calculating a transcendental function.

The time complexity per force evaluation of this algorithm is $\mathcal{O}(E+F)$, where E and F are the number of edges and faces,

Presented as Paper AAS 07-223 at the 17th AAS/AIAA Space Flight Mechanics Meeting, Sedona, Arizona, 28 January–1 February 2007; received 13 June 2007; revision received 27 November 2007; accepted for publication 29 November 2007.

*Graduate Student, Department of Computer Science, 201 North Goodwin Avenue; colombi@uiuc.edu.

†Assistant Professor, Department of Computer Science, 201 North Goodwin Avenue; hirani@cs.uiuc.edu.

‡Assistant Professor, Department of Mechanical and Aerospace Engineering, 4200 Engineering Gateway; bvillac@uci.edu.

respectively. As a result, simulations with this method, especially with a large model, can be very costly. Efforts to reduce this burden have been researched by Cangahuala [6], who investigated three techniques to reduce computational cost of the polyhedral method: 1) shape coarsening, 2) calculation caching, and 3) Taylor approximations. Cangahuala found that all three methods have a significant impact on the running time of a simulation. Specifically, in ideal circumstances, a calculation could be sped up 100 fold without compromising the model's accuracy; more general orbits still benefited from a 10 times speedup [6].

B. Interpolation-Based Method

Interpolation-based methods, as developed in Junkins [8] and Engels and Junkins [9], compute gravitational forces by differentiating a polynomial fit of the geopotential in a (small) region of space. With their approach, accurate gravity models around spherical bodies can be represented with relatively low-degree polynomials, and force evaluations are reduced to efficient polynomial evaluations. Hence, in effect they trade computational complexity against memory availability. Though such methods are usually applied to onboard navigation, where computing resources are limited, we use the same philosophy to address the bottlenecks mentioned in the introduction.

Given the nearly ellipsoidal shape of the Earth, [8,9] divide space with a regular grid in ellipsoidal coordinates. Orthogonal polynomials are used as an interpolation basis for the interpolation of the geopotential. In that setting, each component $F_j(\mathbf{x})$ of the gravitational force, $\mathbf{F}(\mathbf{x}) = -\nabla U(\mathbf{x})$, is approximated as a linear combination of the derivative of the polynomial interpolation basis, $p_i(\mathbf{x})$, that is,

$$F_j(\mathbf{x}) \approx \sum_{i=1}^{n} a_i \frac{\partial}{\partial x_j} p_i(\mathbf{x})$$

where the coefficients a_i depend on the interpolation nodes within an interpolation domain.

Although the virtues of the interpolation approach (efficient force evaluation and ease of local gravity model update) are attractive for small body missions, it is not clear that a uniform grid is practical for reaching the accuracy required for precise orbit determination around such bodies. For example, in Sec. III, we give a rough calculation showing that between 6 and 74 petabytes of storage would be required for a uniform grid, depending on the order of interpolation. A nonuniform grid, on the other hand, raises doubts about the existence of an interpolant for arbitrary data [10].

III. Polynomial Force Interpolation Near Irregular Bodies

Before developing a complete solution, we describe how gravitational force can be interpolated locally. Following this description are numerical experiments exploring the errors of our local interpolation.

A. Interpolation Scheme

Given our initial goals of accelerating force evaluations for numerical integration applications, we opted to directly interpolate the force rather than the potential. Although this makes our force not globally exact (i.e., not the gradient of a potential), this choice proved adequate for the purpose of this research.

Several shapes for interpolation domains, henceforth referred to as cells, have been considered; a cubic region was finally selected for its simplicity and sufficiency. Below we give a more detailed account of the interpolation used.

To interpolate gravitational force, we choose Gauss–Lobatto–Legendre (GLL) interpolation points with the barycentric form of Lagrange polynomials. GLL points have a low Lebesgue constant, which translates to being close to the best uniform approximation of the interpolated function [11]; and the barycentric form of Lagrange polynomials are known for their superior numerical conditioning and computational efficiency [12].

Constructing GLL points in a cubic domain is achieved by Cartesian product of the one-dimensional case. The one-dimensional GLL points of order n are the $n+1$ zeros of

$$\ell(x) := (x-1)(x+1)P_n'(x) \tag{1}$$

where $P_n(x)$ is the Legendre polynomial of degree n. The multidimensional case is then built by computing these zeros for each axis separately and forming their Cartesian product, as illustrated in Fig. 1. We shall refer to this collection of points as N and use $|N|$ to denote the number of points in N, which is $(n+1)^3$.

Lagrange polynomials satisfy

$$p_{\mathbf{x}_i}^N(\mathbf{x}_j) = \delta_{ij} \quad \text{for } \mathbf{x}_i, \mathbf{x}_j \in N \tag{2}$$

and can be computed as the products of the univariate Lagrange polynomials centered at the coordinates of $\mathbf{x}_i$. In three dimensions, this yields

$$p_{\mathbf{x}_i}^N(\mathbf{x}) = \frac{\ell(x)}{\ell'(x_i)(x-x_i)} \frac{\ell(y)}{\ell'(y_i)(y-y_i)} \frac{\ell(z)}{\ell'(z_i)(z-z_i)} \tag{3}$$

which is of degree $3n$. Here $\ell(x)$, $\ell(y)$, and $\ell(z)$ are as defined in Eq. (1), and each factor in the product above is the barycentric form of the univariate Lagrange polynomial [12]. The factors corresponding to each dimension are of the same degree n. For this reason, we will call the interpolation scheme based on such polynomials "order n interpolation" and refer to interpolating polynomials as "order n polynomials." Note that the quantities $\ell'(x_i)$, $\ell'(y_i)$, and $\ell'(z_i)$ only depend on the locations of the interpolation points. Thus, these can be precomputed and stored in a table, which costs $\mathcal{O}(|N|^2)$ [each ℓ' costs $\mathcal{O}(|N|)$, and we must compute one for each $\mathbf{x}_i \in N$]. The evaluation of the interpolation polynomial, however, is an $\mathcal{O}(|N|)$ operation: all $p_{\mathbf{x}_i}^N$ share the term $\ell(x)\ell(y)\ell(z)$, so this can be computed once per evaluation at cost $\mathcal{O}(|N|)$. Finally, each $p_{\mathbf{x}_i}^N$ also uses $1/[(x-x_i)(y-y_i)(z-z_i)]$ for a total across all $p_{\mathbf{x}_i}^N$ of an additional $\mathcal{O}(|N|)$. Thus, interpolation using barycentric form of Lagrange polynomial basis requires $\mathcal{O}(|N|^2)$ setup but only $\mathcal{O}(|N|)$ work per evaluation.

Now we can construct a polynomial approximation of function f as a linear combination of the function's value at the interpolation nodes times the Lagrange polynomial centered at that point:

$$f(\mathbf{x}) \approx \sum_{\mathbf{x}_j \in N} f(\mathbf{x}_j) p_{\mathbf{x}_j}^N(\mathbf{x}) \tag{4}$$

In our case, the function f corresponds to a component of the gravitational force around an asteroid. The forces per unit mass are computed via the polyhedral method, but can be obtained from other sources as well, such as measured gravimetric data or by numerically solving Poisson's equation at the interpolation points.

The interpolation error of the above scheme can be quantified via Ciarlet's formula [10], which in our case depends on an integral of derivatives of the components of the force. This suggests that the error will increase as we approach the asteroid (because force depends inversely on the second power of distance) and decrease as the interpolation domain becomes smaller.

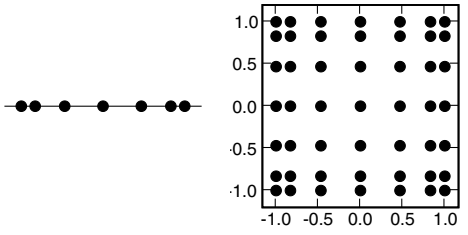

Fig. 1 Left: one-dimensional Gauss–Lobatto–Legendre nodes. Right: two-dimensional Cartesian product of the one-dimensional nodes.

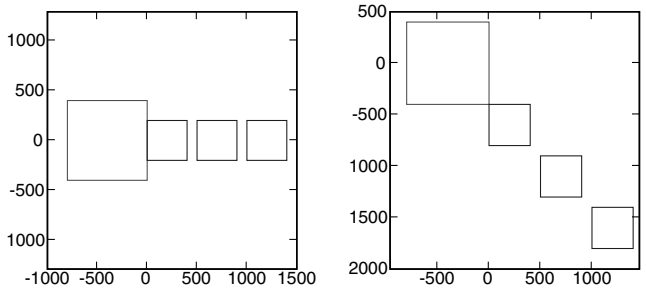

Fig. 2 Context for computing error as a distance from body. The body is the larger square and the smaller squares are test regions. Left figure is the context for face approach in which the distances are 0, 500, and 1000 m. Right figure is context for edge approach in which distances are 0, 707, and 1414 m. These are labeled adjacent, medium, and far in Table 1, which shows the errors.

B. Numerical Experiments

Now we study interpolation errors using numerical experiments. To find the error in a cell, we took random samples of the approximate force $\mathbf{F}_a$ and exact force $\mathbf{F}_e$ (computed via the polyhedral method); we computed the relative error as $\|\mathbf{F}_a - \mathbf{F}_e\|/\|\mathbf{F}_e\|$. The maximum such error was taken as the error bound of a cell. Note that this measure is dimensionless. In the following, we explore the change in interpolation error as parameters of our local model are varied: distance from the asteroid, effect of surface irregularities, size of the cell, and order of the polynomial interpolation. Cangahuala [6] suggests that the relative error in acceleration that is acceptable for mission design is 10^{-5}; thus, for the remainder of the paper, we will target 10^{-5} error or less. All tests were done with three-dimensional bodies and cells. For simplicity, many of the figures depict a two dimensional slice. The banding that is visible in the subsequent error graphs is due to the location of the interpolation points and the shape of the asteroid. In each of these plots, the logarithm (base 10) of the error is shown.

1. *Distance from an Asteroid*

Our first experiment was designed to test the hypothesis that gravity in cells closer to the asteroid would be more difficult to approximate than in cells farther away. We set up an interpolation cell with 400 m sides and order 6 polynomials at progressively closer locations. To keep our test simple, we used a cuboid with the approximate dimensions of Castalia [13] as our asteroid. The cells used were in two types of locations. One set was at varying distances from a flat face of the asteroid at 0, 500, and 1000 m between the closest part of the interpolation region and the surface of the asteroid. A second set was placed similarly at 0, 707, and 1414 m from an edge of the asteroid. Figure 2 shows the test cases, and Table 1 summarizes the results.

In both cases, we can see that bringing the cell closer to the asteroid increases the error in approximation. For the edge-on case, we see a dramatic difference, though this is probably due to the presence of the edge, which is a high-curvature feature. To investigate the shape effect further, we repeated this experiment with a point-mass approximation instead of the polynomial interpolant while keeping the test regions identical. The results of this experiment are also summarized in Table 1. The point-mass approximation is very ill suited to approximating our cuboid asteroid. Thus, the errors made by the polynomial interpolant are not merely a result of misrepresenting a $1/r^2$ term because if the $1/r^2$ term were dominant, a point-mass approximation would suffice. From this, we conclude that shape effects significantly impact the errors present at this range.

Table 1 Effect of distance on error of approximation

	Face approach max. error		Edge approach max. error	
Distance	Interpolation	Point mass	Interpolation	Point mass
Far	2.29×10^{-8}	0.168	3.89×10^{-9}	0.0819
Medium	2.95×10^{-7}	0.400	7.76×10^{-8}	0.192
Adjacent	9.77×10^{-7}	1.85	5.37×10^{-3}	0.634

2. *High-Curvature Features*

To further investigate the effects of high-curvature features, we performed another test. In this experiment, we used the same Castalia-like cube but added a tetrahedron to one face to act like a small hill on the surface. We started with a tetrahedron with 200 m edge lengths and scaled down from there. For each configuration, we used a cell with 25 m sides and order 6 interpolation, placed such that the center of the closest face was aligned with the tip of the tetrahedron. Figure 3 shows some test cases, and Table 2 summarizes the results.

As we want errors beneath 10^{-5}, it seems that even a slight bump can give polynomial interpolation significant problems. Given that the 1998 ML14 model has edge lengths as small as 9 m, this is especially troubling. The solution, as we shall see, is to use smaller cells that wrap around the feature instead of one large cell.

3. *Varying the Size of a Cell*

Given the poor performance of polynomial interpolation for cubes at the 25 m scale, we wanted to know how small our cubes would need to be to achieve the desired error. In this experiment, we studied the effect of varying the size of a cell on the accuracy of our interpolation. To test this, we placed a cell with order 6 interpolation centered near a tip of a 1998 ML14 model. Starting with an edge length of 250 m, we halved each dimension of the cube for every subsequent experiment. Figure 4 shows some test cases, and Table 3 summarizes the results.

We can use these results to motivate the case for an irregular grid. First, assume the domain of interest is a cube surrounding 1998 ML14 with edge length 2500 m; this is a fair assumption because

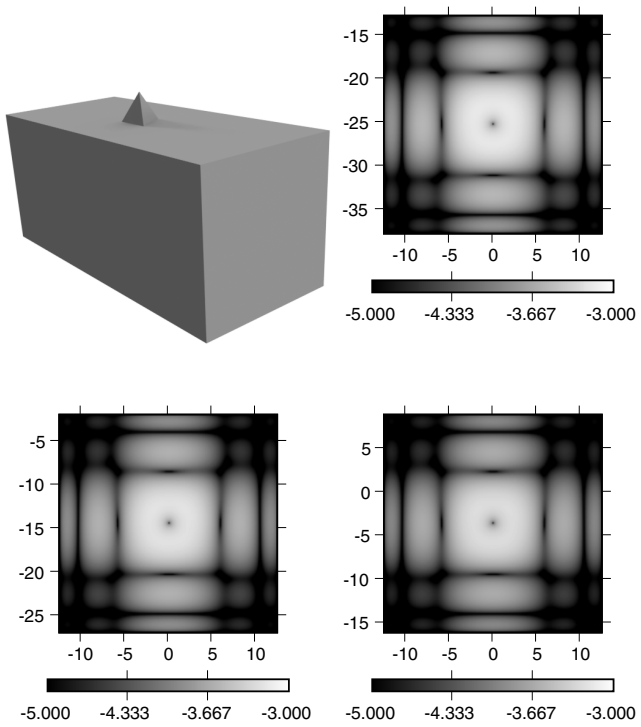

Fig. 3 Effect of small features on error. The shaded plots show the logarithm of relative error of gravitational force along the face of the cell touching the tetrahedron feature. The goal is to avoid errors larger than 10^{-5}, so saturated white regions indicate a poor approximation. Top left shows an image of our test body. In this image, the tetrahedron had 200 m edge lengths. Error plots shown are for tetrahedra with edge lengths of 175 m (top right), 100 m (bottom left), and 25 m (bottom right). Each cell used order 6 interpolation, and the true values were taken to be the ones computed by the polyhedral method. See also Table 2.

Table 2 Effect of small features on error

Feature size, m	Max. error	Feature size, m	Max. error
200	6.84×10^{-4}	100	6.00×10^{-4}
175	6.62×10^{-4}	75	5.79×10^{-4}
150	6.41×10^{-4}	50	5.62×10^{-4}
125	6.19×10^{-4}	25	5.37×10^{-4}
——	——	0	3.98×10^{-13}

anywhere outside that range we can use low-order spherical harmonics as a fast approximation. From these results, we conclude that we will need resolution down to 12 cm to capture the fine details near the surface. With a regular grid, this would require dividing the domain into $(2500/0.12)^3 \approx 9 \times 10^{12}$ cells. If each cell contains order 6 interpolation, we need $7^3 \times 3 = 1029$ double-precision coefficients, or 8232 bytes per cell (the power of 3 comes from three dimensions, and the factor of 3 from each component of force). The total memory cost for such a model is about 74 petabytes. Even at order 2, we would need around 6 petabytes to store a regular grid.

4. Varying the Polynomial Order

Our next experiment focused on varying the order of approximation; specifically, we explored the interaction between distance from the asteroid and order of approximation. Two cells centered at (441, 231, 0 m) (near surface) and (750, 231, 0 m) (about 1.5 radii away) with 250-m edge length and polynomial interpolants between orders 1 and 7 were tested. Figures 5 and 6 show some test cases, and Table 4 summarizes our results.

Far away, there is a clear benefit to using high-order polynomials for approximating the gravitation. Closer in, however, these results show diminishing returns for higher-order polynomials. We conclude that the appropriate strategy for polynomial approximation

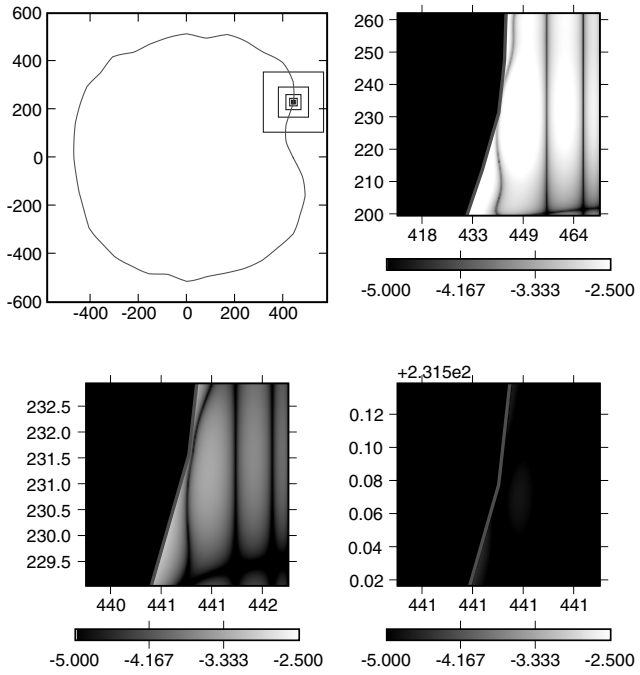

Fig. 4 Error as a function of cell size. The shaded plots show a cross section of the logarithm of relative error of gravitational force. The goal is to avoid errors larger than 10^{-5}, so saturated white regions indicate a poor approximation. Top left shows the context for the test. A silhouette outline shows the boundary of asteroid 1998 ML14; the squares are slices of cells we tried. Errors are shown for cells with edge lengths 250 m (top right), 62.5 m (bottom left), and 3.91 m (bottom right). The saturated white regions in the first two error plots show that large errors are present. The errors in the last plot are within tolerance. Each cell used order 6 interpolation, and the true values were taken to be the ones computed by the polyhedral method. See also Table 3.

Table 3 Effect of cell size on error

Size, m	Max. error	Size, m	Max. error
250	5.75×10^{-2}	3.91	9.03×10^{-4}
125	2.85×10^{-2}	1.95	4.63×10^{-4}
62.5	1.46×10^{-2}	0.977	2.28×10^{-4}
31.3	7.24×10^{-3}	0.488	1.14×10^{-4}
15.6	3.63×10^{-3}	0.244	5.66×10^{-5}
7.81	1.86×10^{-3}	0.122	2.82×10^{-5}

uses few high-order cubes far away and many low-order cubes closer to the asteroid. This is somewhat counterintuitive, as one may expect high-order approximations to yield the most significant benefits where the field changes most rapidly.

With all these results in mind, we conclude that an efficient representation of gravitational force near a small body must be adaptive in both size and interpolation order of each cell.

IV. Adaptive Spatial Partitioning

Our approach to modeling the gravitational force near an asteroid separates the problem into two tasks: dividing the domain into cells and approximating the force in a cell. Having already settled on a solution for the latter, we now discuss the former.

Subdividing the domain into manageable cells is accomplished with an adaptive *octree* data structure [14]. This data structure is constructed by recursively splitting the domain into a hierarchy of different-sized cuboids so that each contains a local model of gravitational force. We initialized the process with a local model for the highest-level cuboid. From here we estimate the error relative to the polyhedral method; if the error is too large, the cuboid is divided, and the process is repeated recursively on each of the pieces. Otherwise the cuboid is retained and becomes a *leaf* cell in the octree. The interpolation information is retained only for leaf cells, and it is discarded for the rest. In building the octree data structure, we conservatively choose a relative error threshold of 5×10^{-7}. This is almost 2 orders of magnitude less than the error in acceleration

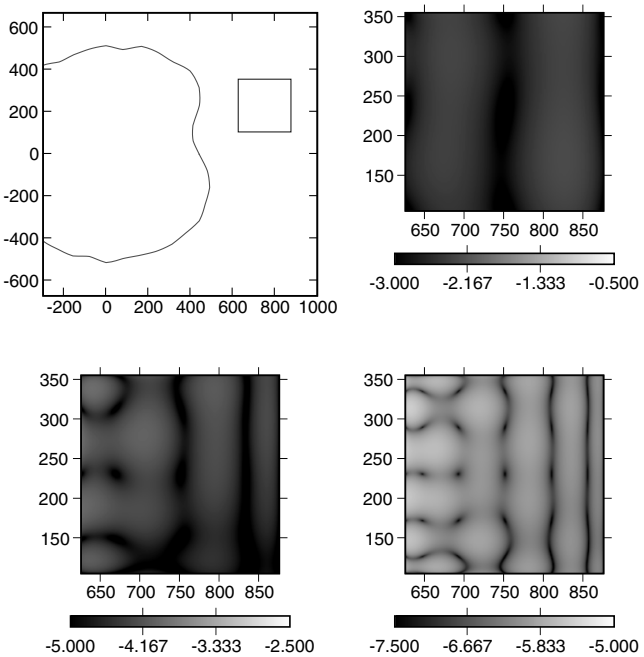

Fig. 5 Error as a function of order of interpolation in a distant cell. The shaded plots show a cross section of the logarithm of relative error of gravitational force. Top left shows the context for the test. The silhouette outline shows the boundary of 1998 ML14; the square is a projection of the cell. Error plots shown are for cells with interpolation orders 2 (top right), 4 (bottom left), and 6 (bottom right). The true values were taken to be the ones computed by the polyhedral method. See also Table 4.

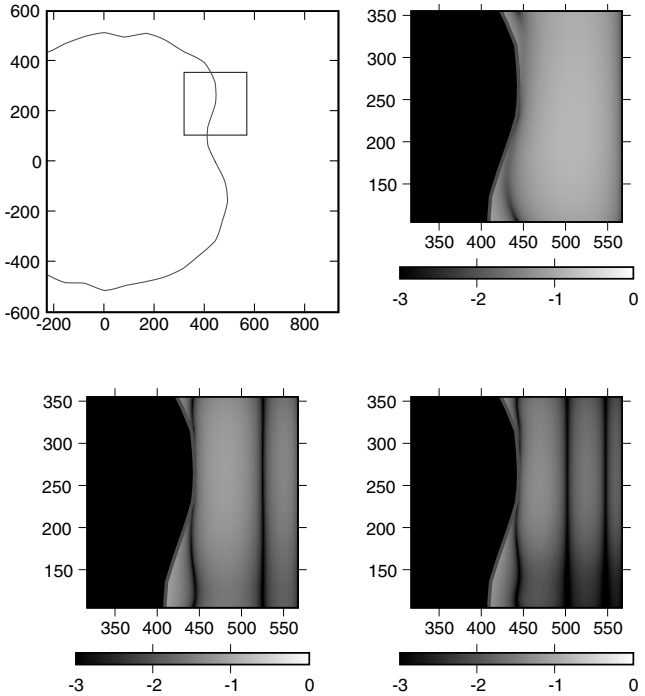

Fig. 6 Error as a function of order of interpolation in a cell close to the asteroid. The shaded plots show a cross section of the logarithm of relative error of gravitational force. Top left shows the context for the test. The silhouette outline shows the boundary of 1998 ML14, the square is a projection of the cell. Error plots shown are for cells with interpolation orders 2 (top right), 4 (bottom left) and 6 (bottom right). As we can see all errors reported are outside the acceptable range. The true values were taken to be the ones computed by the polyhedral method. See also Table 4.

advised by Cangahuala [6]. We choose the more conservative threshold during octree construction because error estimation during that stage is based on random sampling as explained in Sec. III.B.

The use of such an adaptive structure, in combination with the interpolation scheme we used, precludes continuity of the interpolated force across cell boundaries. However, in our experiments with trajectories, this was not an issue, as shown by results of Sec. VI.

A. Octrees

We now provide practical details regarding the octree data structure. Octrees are more easily illustrated by their 2-D analog; hence, the following description is given for "quadtrees." Quadtrees are adaptive tree data structures for organizing localized data in a rectangular domain. Two operations characterize the function of a quadtree: `Subdivide` and `Find`. `Subdivide` is a constant time operation that splits a rectangle into four quadrants by splitting each dimension in half. A quadtree is built by beginning with a single rectangle and subdividing recursively until the desired tree structure is created [14]. Figure 7 shows a quadtree as a tree and as a collection of rectangles.

Table 4 Effect of interpolation order on error

Order of interpolation	Distant case max. error	Close case max. error
1	7.94×10^{-2}	3.48×10^{-1}
2	5.37×10^{-3}	1.40×10^{-1}
3	8.91×10^{-4}	1.13×10^{-1}
4	1.20×10^{-4}	7.59×10^{-2}
5	2.34×10^{-5}	7.16×10^{-2}
6	3.23×10^{-6}	5.75×10^{-2}
7	6.03×10^{-7}	5.37×10^{-2}

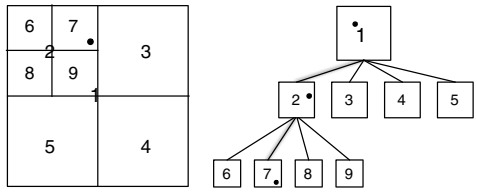

Fig. 7 Left: a quadtree viewed geometrically. Right: a quadtree viewed as a tree. Labels show the mapping between geometric and tree views. The point represents a query; the tree view shows the query being resolved. Cells 3 to 9 are leaf cells.

`Find` recalls localized data associated with query points by recursively traversing the tree. At each level, `Find` picks the child quadrant containing our query; this is a constant time operation, as each cell has at most four children. After every level is traversed, a leaf is reached, and the data it contains is returned. As balanced trees have at most $\log N$ levels, the run-time complexity of `Find` is $\mathcal{O}(\log N)$, where N is the total number of cells [15]. Thus, subdividing our model to improve accuracy incurs only a sublinear run-time penalty. This compares favorably to the polyhedral method, which requires a linear cost increase to improve accuracy.

To illustrate the practical benefit of a sublinear run time, consider the following example. For the sake of argument, we shall examine moving from polyhedral models with 1000 elements to 10,000, and compare that to moving from 10,000 octree cells to 100,000. Following this scenario, the polyhedral method would cost 10 times more computation when moving to the new model, whereas an octree method only costs 1.2 times the former computation. Taking this out another factor of 10, we find costs rising 100 times and 1.4 times, respectively. In other words, methods with asymptotically better performance have dramatically superior run times as problems scale up to take advantage of newest computational power available.

Note that the computational complexities of polyhedral method versus our method depend on different things (asteroid mesh complexity versus number of cells, respectively). The same gain in accuracy by the two methods may require different refinements. Thus, comparing complexities as we do above is simplistic. However, our experiments described in Sec. VI show that trajectory integration using our method is much faster than the polyhedral method while producing extremely accurate trajectories. This is made more precise in Sec. VI.

Octrees follow the same design, but use eight cuboids in 3-D instead of four rectangles in 2-D.

B. Spherical Harmonics Far Away

The octree structure described herein must exist in a bounded cuboid region; this precludes it from producing approximations to acceleration everywhere outside the body. To provide such approximations, we employ spherical harmonics. We place a sphere centered at the origin and just large enough to enclose the body. Then coefficients to spherical harmonics that fit the gravitational potential are computed. In theory, one could take sufficient coefficients and produce an accurate field for all space outside the sphere; in practice, computing to such a degree of accuracy is too costly. Instead we satisfy ourselves with accurate results outside the octree domain. This is easier to achieve as the higher order components of spherical harmonics fall off quickly as distance from the body increases.

V. Example Octree Construction and Performance

To investigate the errors in our approximate gravity field, we constructed an octree model in a region around asteroid 1998 ML14 and numerically analyzed the interpolation error. The experiments were done using a triangle mesh surface model of the asteroid 1998 ML14 with 8162 vertices, 24,480 edges, and 16,320 triangles [16].[§]

[§]Data available online at http://www.psi.edu/pds/resource/rshape.html, EAR-A-5-DDR-RADARSHAPE-MODELS-V2.0 [retrieved 10 June 2007].

A. Octree Model Construction and Error Analysis

The domain of the octree began at $(-1250\ \text{m}, -1250\ \text{m}, -1250\ \text{m})$ and extended 2500 m in each direction, where the origin is the center of mass of the asteroid model. (As a point of reference, the radius of 1998 ML14 is ≈ 500 m.) The octree was limited in depth to 10 levels; this implies a smallest cell size of 4.88 m. The first three levels of cells used order 6 polynomials, the last two levels order 2, and the rest used order 4. Each cell was tested with 10,000 sample points, and subdivision continued until the maximum of these errors was beneath 5×10^{-7}. The error was measured as described in Sec. III.B. This model was created in 1150 CPU hours on a parallel computer using the Message Passing Interface (MPI) [17] (64 processors for approximately 18 h) and occupies 653 MB of memory. To cover the region outside the octree, spherical harmonics of degree and order 12 were employed.

Part of an octree structure for 1998 ML14 is shown in Fig. 8, and Fig. 9 shows cross sections of the error in the x–y plane. The first error plot starts at (400 m, −250 m) and extends 500 m; subsequent plots magnify the region near (400 m, 100 m) by 2 and 4 times. As we can see, errors are very well behaved for the majority of the plotted regions. In fact, only when we get very close (within a few meters) to the surface are we in danger of violating our goal of 10^{-5}. This is due to the bound on the octree depth imposed on this particular experiment and can be improved by building a tree with smaller cells close to surface. Note that even though the error bound is violated at the surface, the results from our experiments on ejecta trajectories in Sec. VI.E show that this is not a problem in practice.

B. Single Evaluation Speed Tests

In this experiment, we measured the comparative performances of the competing models by measuring the time required for a single force evaluation. Table 5 summarizes the relative speeds (1.0 being polyhedral method) of several methods. We can see that the polynomial interpolation scheme compares favorably with other methods. Compared to Cangahuala [6], we have similar performance but better understanding of the errors. Specifically, whereas Cangahuala [6] only reports errors for orbits at 3 radii from the body, our error estimates go all the way to the body.

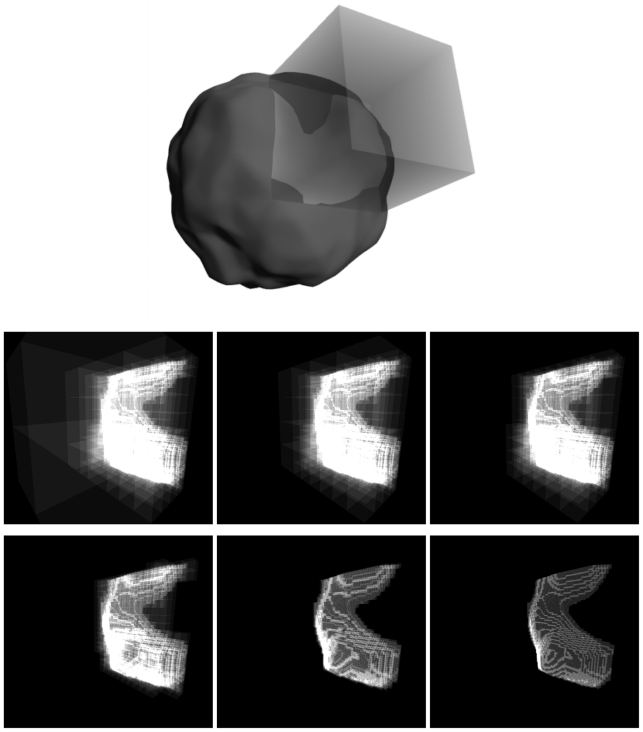

Fig. 8 An octree model for part of space around asteroid 1998 ML14. The cube in the top figure shows the region for which this octree was constructed. The cubes in the bottom two rows are visualized as translucent to reveal the hidden structure. In the sequence shown in bottom two rows, larger cubes are incrementally removed to reveal the finer structure of the octree.

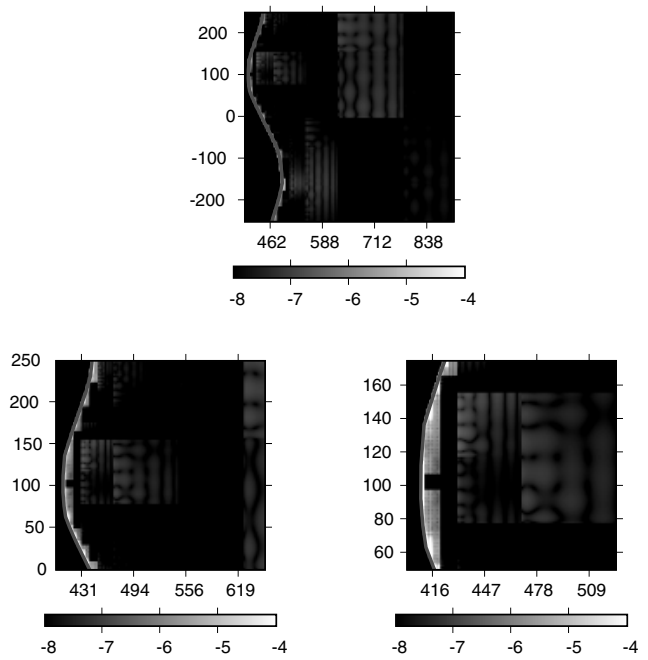

Fig. 9 Error of an octree model for asteroid 1998 ML14. The plots show the logarithm of relative error in gravitational force. Top plot shows the error along the x–y plane at $z = 0$ m cutting across many cells. Lower left and right show a zoom of 2 and 4 times. The error in most of the cells is less than 10^{-5} as desired. The only exceptions are in the cells very close to the asteroid. These are cells of size about 5 m. Cells used a variable order interpolation depending on their size; orders ranged from 2 to 6. The true values of force were taken to be the ones computed by the polyhedral method.

VI. Performance Analysis Using Trajectory Integrations

Whereas the measurement of a single force evaluation described in previous section gives some idea of the order of speedup obtained with the octree method, this factor is actually a function of space. For example, cells closer to the asteroid are usually deeper in the octree, and so they receive lower-order interpolants. To get a notion for actual speed improvements, we have to integrate trajectories. This was done for four different classes of trajectory: close retrograde orbits, midrange orbits, random trajectories, and ejecta.

A. Experiment Design

In each of the experiments below, we generate several trajectories with different force models and parameters.

Table 5 Relative speeds of available methods for single-force evaluation

Model	Speed factor	Comments
Polyhedral	1.0	——
Order 6 polynomial	0.0104	——
Order 4 polynomial	0.0038	Estimated from order 6
Order 2 polynomial	0.0008	Estimated from order 6
Degree and order 12 spherical harmonics	0.0145	——
Coarse shape, Taylor series	0.091	See [6]
Coarse shape, Taylor series, histories	0.01	See [6]

Cubetree trajectories are generated using the force model described herein. Integration is done with the embedded Runge–Kutta Prince–Dormand (order 8, 9) method using relative error tolerance 10^{-13} and absolute error tolerance 10^{-6}. Going beneath 10^{-6} can cause problems with our method, as the discontinuities in the force will cause the adaptive time-stepping routine to overrefine the step size.

A reference trajectory refers to a simulation done with the polyhedral method as described by Werner and Scheeres [4]. Integration is done with the same method as above; however, the absolute error threshold is set to 10^{-10}. These trajectories are used as a baseline to measure the accuracy of cubetree trajectories.

Finally, trajectories generated with the augmented polyhedral model use a mix of polyhedral and spherical harmonics: spherical harmonics are used wherever they would be used in a cubetree trajectory. Furthermore, augmented polyhedral trajectories use the same tolerances as cubetree trajectories. The augmented polyhedral method is used to measure timing performance of the cubetree model.

The trajectory integrations were done in rotating coordinates. The period of rotation for 1998 ML14 was assumed to be 14.93 h, and the moment of inertia tensor was computed from the triangle mesh surface of the asteroid. This was used to compute the principal axis. The local coordinate system for the asteroid was used as the rotating coordinate system; the z axis turned out to be close to but not exactly the same as the principal axis. The computed normalized principal axis, in the coordinate system of the asteroid mesh, was (0.0636, 0.0008, 0.9356). Thus, the x–y plane was close to but not the same as the equatorial plane.

For the semimajor axis calculation in Sec VI.F, the mass value was calculated from an assumed density of 2.5×10^3 kg/m^3 and computed volume of approximately $511,320,552$ m^3.

B. Close Retrograde Orbits

For our first experiment, we chose a known family of stable orbits. Initial conditions were chosen randomly within a band of retrograde orbits close to the asteroid. Specifically, we placed initial conditions near the equatorial plane with randomly chosen radii between 600 and 1000 m from the center of the asteroid. Velocities were always chosen to place the orbiter in a retrograde orbit. Initial speeds were chosen between 0.45 and 0.75 of escape speed. [The escape speed is evaluated in inertial space from $v = \sqrt{2U(\mathbf{x})}$, where U is the gravitational potential (at the selected location, $\mathbf{x}$) computed from the polyhedral method.] The only force simulated was gravitation in rotating coordinates. Simulations ran for 30 days of ballistic motion with each model (cubetree, augmented polyhedral, and reference), and impacting trajectories were thrown out. Impacting trajectories are addressed in Sec. VI.E. The position and velocity of the orbiter was recorded every 5 minutes of simulated time.

This experiment was repeated for 1111 trajectories. For each trajectory, we measured the maximum difference in position and velocity between the cubetree trajectory and reference trajectory. Figure 10 is a histogram of the errors in position and velocity. Clearly, the vast majority of trajectories fall within 2 m of the reference trajectory; in fact, only four trajectories were outside a 2 m range. The maximum position error was 3.56 m and the minimum was 9.76 mm. On average, integrations with the cubetree method were 112 times faster than the augmented polyhedral method.

C. Midrange Orbits

Another region of interest in our model is the jump between octree and spherical harmonics. To investigate this domain, we performed a similar experiment. Initial conditions were placed between 1250 and 1500 m from the center; velocities were randomly picked between 0.67 and 0.8 of the escape speed with both prograde and retrograde orbits. Inclination of the orbits was limited by choosing initial positions near the equatorial plane and initial velocities with a small component outside the equatorial plane. Otherwise, this experiment was identical to the previous one.

We repeated the simulation for 1487 trajectories. Figure 11 is a histogram of the errors in position and velocity. Again, the majority of trajectories differ by less than 2 m; only 15 have position error greater than 2 m. The minimum error in position is 3.2 mm, median 12.22 cm, and maximum 10.24 m. These three orbits are shown in Fig. 12. Note that the maximum error is significantly larger than in our previous experiment. We expect these worst-case trajectories to have inherently sensitive dynamics, a hypothesis we shall revisit and show evidence for in Sec. VI.F. On average, integrations using our cubetree method were 90 times faster than using the augmented polyhedral method. Recall that both methods used spherical harmonics outside a certain range.

D. Random Trajectories

Next, we explored more of the phase space. Positions were taken between 600 and 1500 m from the center of the asteroid, and velocities were chosen from the plane passing through the initial position and tangent to the sphere centered at the origin. Magnitude of the velocity was clamped to within 0.45 and 0.75 of escape speed. Simulation was performed identically to the previous two and was repeated 911 times. The histograms in Fig. 13 summarize the results; the last column in each histogram represents all differences greater than or equal to the 20 m or mm/s. In this experiment, even more cubetree trajectories diverge from the reference: 38 trajectories have error greater than 2 m and of those, 8 have greater than 100 m error. On average, the integration time of these trajectories was accelerated by 111 times.

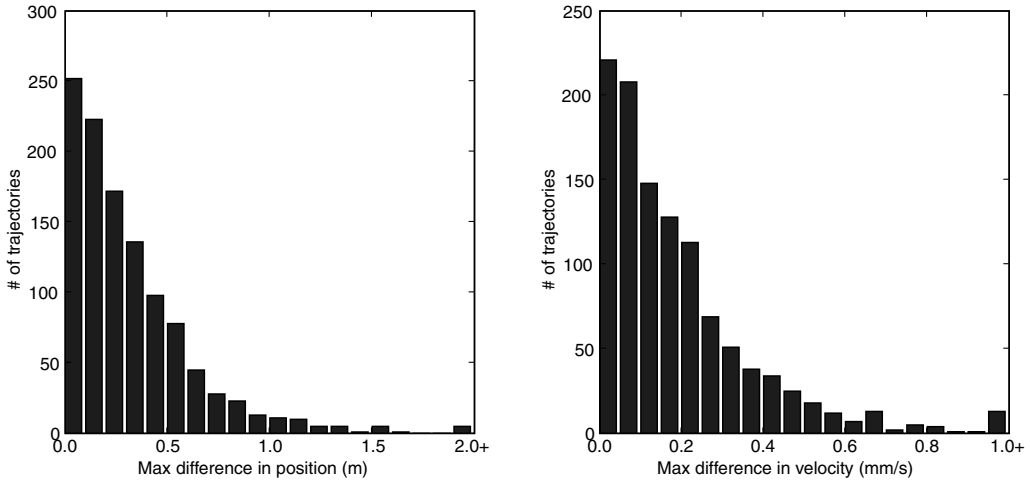

Fig. 10 Histograms of errors in position and velocity for 1111 close retrograde cubetree trajectories integrated for 30 days and observed at 5 min intervals.

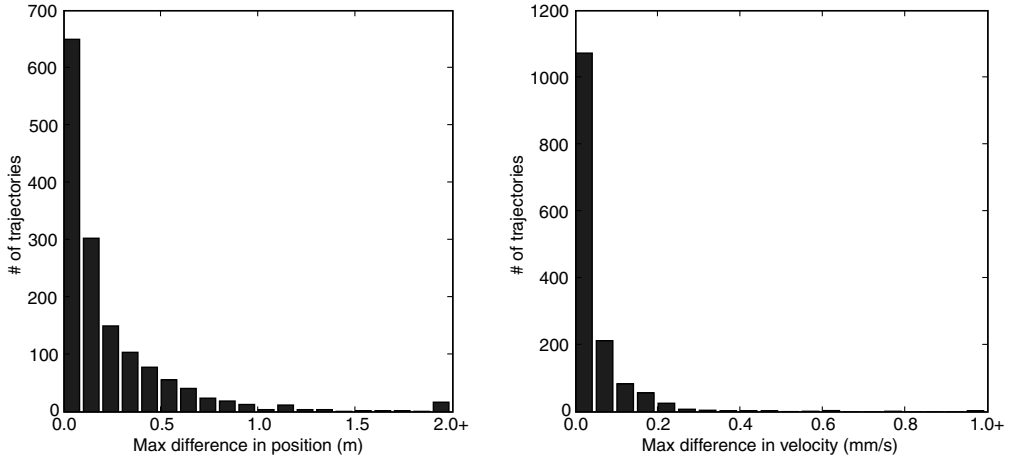

Fig. 11 Histogram of errors in position and velocity for 1487 midrange cubetree trajectories integrated for 30 days and observed at 5 min intervals.

E. Ejecta Trajectory

The final experiment focused on ejecta and impacts. Initial positions were chosen with a uniform random distribution on the polyhedron's surface; initial velocities were chosen from the hemisphere above the surface and with magnitude uniformly distributed between 0.1 and 0.9 of escape speed. Otherwise, the experiment is the same as the previous one; 2440 trajectories were generated this way. Because most ejecta trajectories are short lived, the differences are small: only one trajectory had more than 0.5 m difference in position (it had 4 m difference). As such, histograms have been omitted. Ejecta trajectory were calculated 169 times faster using the cubetree approximation. This improvement reflects the lower-order interpolation used near the surface of the asteroid.

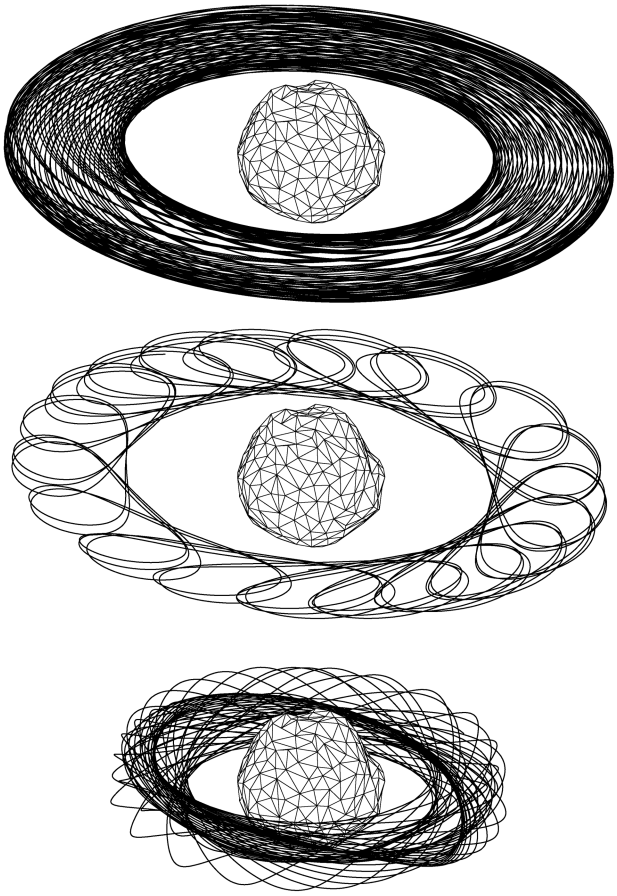

Fig. 12 Three midrange 30 day cubetree trajectories sampled at 5 min intervals. Each plot corresponds to a different initial condition in the experiment described in Sec. VI.C. The top, middle, and bottom have the smallest (3.2 mm), median (12.22 cm), and maximum (10.24 m) errors, respectively. Frequency analysis for these is shown in Fig. 14. The asteroid model shown is a simplified version. The one used for actual trajectory propagation in all experiments had 16,320 triangles.

F. Dynamical Error Analysis

To better understand the errors found in the previous experiments, the underlying dynamical properties of a few sample trajectories have been considered. This analysis indicates that the large errors obtained for particular trajectories are not an intrinsic limitation of the approximation scheme used but rather of the sensitive nature of the trajectories in chaotic regions. This result is achieved via frequency analysis and Monte Carlo sensitivity analysis.

1. *Frequency Analysis*

Given that the physical system we are modeling is conservative (Hamiltonian), Fourier analysis of trajectories is a powerful tool to discriminate between regular and chaotic motion [18,19]. In particular, regular (quasi-periodic) trajectories in Hamiltonian systems, which correspond to stable trajectories and present only linear sensitivity with respect to the initial conditions, exhibit a discrete spectrum of frequencies corresponding to the natural frequencies of the torus on which they lie. On the other hand, chaotic trajectories, which present exponential divergence between neighboring trajectories, present a continuous spectrum and appear as noise on the power spectrum of some coordinates.

To test the hypothesis that trajectories presenting large discrepancies between the numerical integration in the two models are sensitive, we applied a fast Fourier transform (FFT) on sample trajectories in the midrange test case of Sec. VI.C. We chose three sample trajectories corresponding to the minimum, median, and maximum error cases and applied the FFT on the semimajor axis, a, for an integration time span of 30 days with a sampling period of 5 min. The results are shown in Fig. 14 and the actual position paths corresponding to those trajectories are shown in Fig. 12. Figure 14 shows the power spectral density $\mathcal{P}$ of the trajectories for both the cubetree approximation (continuous line) and reference method (dots), as well as the normalized difference in power spectral density, $\mathcal{D}$:

$$\mathcal{D}(f) = \frac{|\mathcal{P}_{\text{cubetree}}(f) - \mathcal{P}_{\text{polyhedral}}(f)|}{\mathcal{P}_{\text{polyhedral}}(f)}$$

As can be observed, the minimum difference trajectory (top plot of Fig. 14) shows a "discrete" spectrum in which a few main frequency peaks are apparent. Thus, this trajectory is likely to be a regular one. On the other hand, as the discrepancy between the integration results based on the two different models increases, the number of frequencies in the power spectrum tends to increase. In the median case, the main frequencies are still dominant but show a small

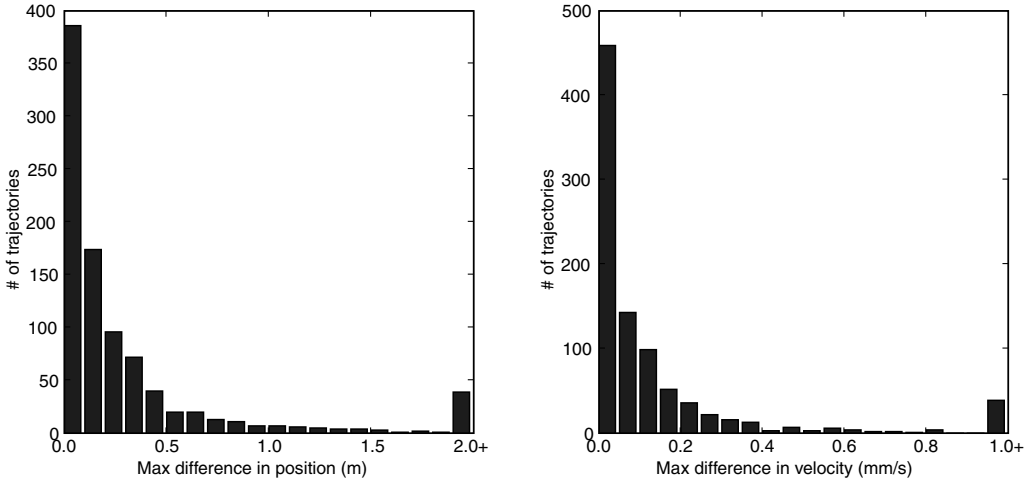

Fig. 13 Histogram of errors in position and velocity for 911 random cubetree trajectories integrated for 30 days and observed at 5 min intervals.

instability. For the largest error case, the noise is significant, suggesting a chaotic trajectory. Similar results have been obtained for the other action elements (eccentricity, inclination). This indicates that, in the cases in which large errors between the polyhedral and cubetree model have been obtained, the discrepancy is likely to be a result of the intrinsic sensitive nature of the trajectories rather than due to the approximation method.

2. Sensitivity Analysis

To better test the extent of the sensitive nature of the trajectories corresponding to the largest errors in the midrange experiment, we performed a Monte Carlo simulation over perturbations around initial conditions. For seed initial conditions, we used the 15 worst-case trajectories from the midrange experiment. Using only reference trajectories, trajectories from each perturbation were compared with the trajectory of the nominal initial condition by taking the Hausdorff distance (the maximum of the minimum distance between two curves; the curves here are taken to be the path in position space of two trajectories). Perturbations were chosen from a normal distribution with the mean centered at the nominal initial condition and standard deviations 1 mm and 1 μm/s for position and velocity. Each initial condition was run with 10 perturbations, and the maximum Hausdorff distance was recorded. Table 6 shows the results; trajectory 1 was the worst-case trajectory, trajectory 2 the second worst, etc.

As can be clearly observed from this Table, small perturbations in the initial conditions lead to significant variations in the resulting trajectories, which shows that the dynamics is highly sensitive to initial conditions. These results quantify the previous observation obtained via frequency analysis: because both models can be considered small perturbations of each other, the discrepancies between two integrated trajectories in chaotic regions may present large variations independently of the approximation method used. The cubetree method captures this high sensitivity as seen in the frequency analysis, which shows its overall consistency.

VII. Discussion

We have presented an efficient method for representing gravitational force of small, irregular bodies that is accurate enough for planning missions near them. Our technique combines an adaptive spatial data structure with polynomial interpolation to cover the entire domain with an approximation of gravitational force. The decomposition of the domain can be quite coarse in some regions, and it also varies in order of interpolation. This flexible spatial hierarchy enables us to refine our model in regions that are difficult to approximate (e.g., near high-curvature regions) and enables error guarantees on the model. We believe this model is competitive with its counterparts for Monte Carlo simulations of spacecraft trajectories passing near small bodies. A summary and comparison of the computational characteristics for each available method follows.

A. Cubetree (Our Method)

Memory: This method (our method described herein) has moderate memory requirements: every octree leaf cell requires $(\text{order}+1)^3 \times 3$ coefficients. The octree used in our experiments described in Secs. V and VI had 386,880 leaf cells and occupied 653 MB. *Speed*: Using interpolating polynomials permits a constant time reconstruction of the force within a cell, and finding the correct octree cell is an $\mathcal{O}(\log(\text{number of cell}))$ operation. In practice, this gives about 100 times speedup over the polyhedral method. *Error*: Errors can be controlled to within user tolerances. In our experiments, we reached our 10^{-5} goal quite near the asteroid (farther than 4 m from the surface) and surpassed it at points 2 and 3 radii away (from the center).

B. Spherical Harmonics

Memory: Spherical harmonics have a very small memory footprint of only $(\text{order and degree}+1)^2$ coefficients to store the whole model. *Speed*: Depending on the order and degree used, spherical harmonics can be up to 100 times faster than the polyhedral method. At best, spherical harmonics approaches the speed of our method and compares favorably with other methods. *Error*: No matter what order is chosen, errors near nonconvex regions of the asteroid render this method useless for missions close to a small irregular body [4]. This puts spherical harmonics at a significant disadvantage to both the polyhedral method and the method presented in this paper.

C. Mascons

Memory: Mascons use memory linear to the number of point masses used. In practice, this number is in the thousands; thus, mascons memory footprint is very small. *Speed*: Mascons are faster than polyhedral methods; however, they are still subject to a linear run-time complexity. In practice, their performance is irrelevant, as they have nontrivial error. *Error*: Errors for this method have not been theoretically bounded, and experiments show that large errors do exist [4].

D. Polyhedral Method

Memory: The primary memory cost of this method is storing the polyhedral model and its associated data. As most models have only tens of thousands of elements, polyhedral methods only require a small amount of memory. *Speed*: Calculation requires iterating over every edge and face; furthermore, each edge and face calculation includes a transcendental function. Hence, polyhedral methods are slow to compute gravitational force; our technique is 2 orders of magnitude faster. *Error*: As long as the polyhedral model and density assumptions are not far from the truth, this method produces exact results. Of course, any errors in this method would spoil models using it as a base line (e.g., the examples considered in this paper).

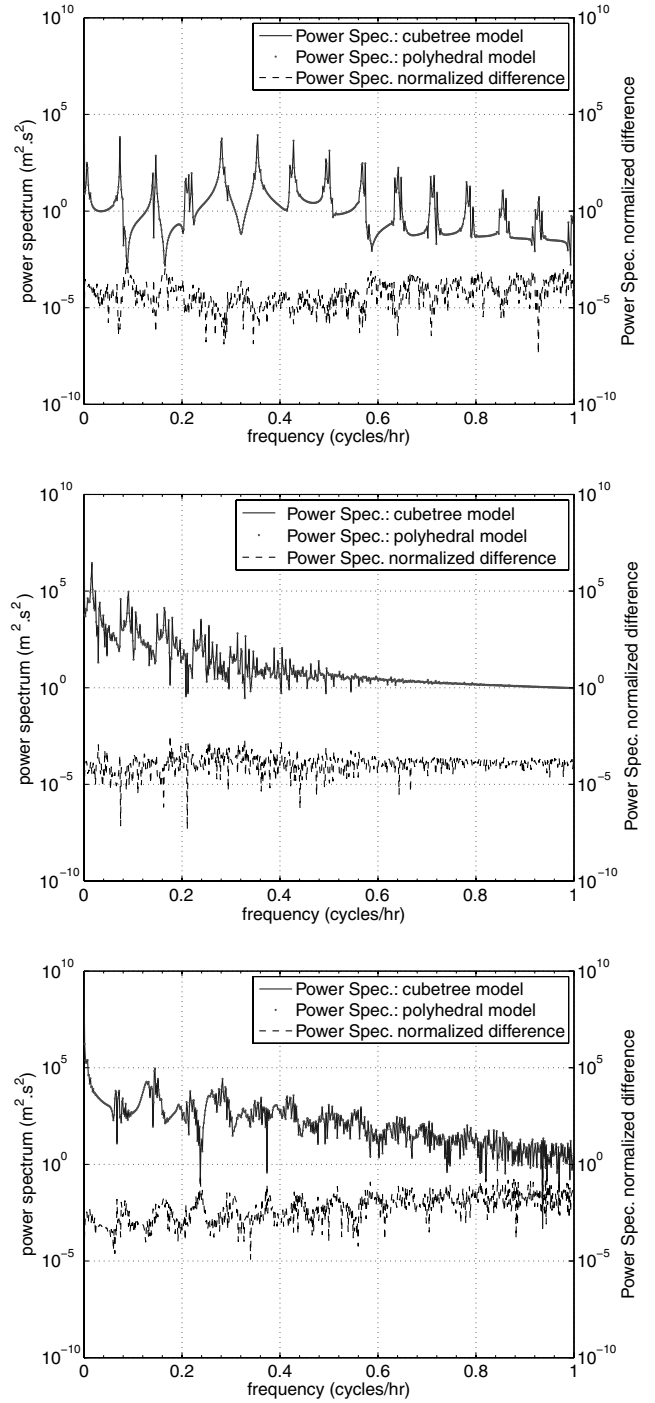

Fig. 14 Frequency analysis of semimajor axis of midrange 30 day cubetree trajectories shown in Fig. 12. The plots, from top to bottom, are for the trajectory with the smallest, median, and maximum errors in the set of midrange trajectories.

E. Modified Polyhedral Method

Memory: In addition to the memory costs of the polyhedral method, modified polyhedral methods [6] cache many prior calculations; this, however, can only take a constant factor beyond the nominal. Hence, modified polyhedral methods also have small memory requirements. *Speed*: Depending on the orbit, speed varies between 10 and 100 times faster. In ideal circumstances, this reaches our performance, but in general, 100 times speedups are not achieved. *Error*: Errors at 3-radii are close to or below 10^{-3}; closer trajectories may experience more or less error. Our method has an advantage here because we can adaptively drive error down as needed and thereby guarantee certain error bounds.

Table 6 Sensitivity with respect to initial condition: maximum Hausdorff distance between reference trajectories and their perturbations for midrange orbits

Trajectory	Difference in position, m	Trajectory	Difference in position, m
1	40.03	9	16.53
2	35.36	10	19.00
3	256.0	11	30.04
4	22.34	12	346.9
5	19.95	13	12.28
6	13.00	14	20.86
7	31.65	15	16.99
8	60.99	——	——

VIII. Open Questions

There are several directions in which this research can proceed. First, alternative local representations should be considered. For example, the Cartesian product construction of interpolation points, as discussed in Sec. III.A, uses many more points than are required for the accuracy provided [20]. Also, the chosen representation is not globally exact, that is, it is not the gradient of a potential. This could be solved by using Hermite interpolation or by other techniques, but the ramifications of these choices must be more carefully considered.

Another avenue of research explores subdivision schemes that provide continuity across cell boundaries. The discontinuities across cells do not seem to have a major impact on trajectory propagation. However, it remains to be seen if this discontinuity affects optimization of, say, low-thrust trajectories. In this case, regularly subdivided tetrahedra or special variants of octrees may be able to resolve discontinuities in the interpolated force across cell boundaries.

Finally, different techniques for capturing the interpolated values should be considered. This data could come from physical experiments, or one might, for example, wish to use Poisson's equation as a starting point. One of the drawbacks of the polyhedral method is that incorporating direct measurements of the force field requires updating the physical description of the small body. On the other hand, interpolatory techniques should be more apt at making corrections because they are based on data to begin with. Aside from the speed of polynomial approximations, this is potentially their greatest strength.

IX. Conclusions

Polynomial interpolation paired with an adaptive spatial data structure provides an efficient framework for gravitational force-field approximation near small, irregular bodies. Even lacking continuity and exactness, this technique produces sufficiently accurate results for mission planning in the vast majority of the trajectories tested. The modest memory requirements and vastly decreased computation time enables our method to bring Monte Carlo simulation off of cluster computers and onto workstations.

Acknowledgments

The authors would like to thank William N. Bell for useful discussions, Keenan M. Crane for assistance with our octree visualization, and the Langage Objet pour la Relativite Numerique (LORENE) project for software that computes the zeros of Gauss–Lobatto–Legendre polynomials. The 1998 ML14 shape model used is from the NASA Planetary Data System, 2004.

References

[1] Abe, S., Mukai, T., Hirata, N., Barnouin-Jha, O. S., Cheng, A. F., Demura, H., Gaskell, R. W., Hashimoto, T., Hiraoka, K., Honda, T., Kubota, T., Matsuoka, M., Mizuno, T., Nakamura, R., Scheeres, D. J., and Yoshikawa, M., "Mass and Local Topography Measurements of Itokawa by Hayabusa," *Science*, Vol. 312, No. 5778, 2006, pp. 1344–1347.
doi:10.1126/science.1126272

[2] Rayman, M. D., Fraschetti, T. C., Raymond, C. A., and Russell, C. T., "Dawn: A Mission in Development for Exploration of Main Belt Asteroids Vesta and Ceres," *Acta Astronautica*, Vol. 58, No. 11, 2006, pp. 605–616. doi:10.1016/j.actaastro.2006.01.014

[3] Miller, J. K., Williams, B. G., Bollman, W. E., Davis, R. P., Helfrich, C. E., Scheeres, D. J., Synnot, S. P., Wang, T. C., and Yeomans, D. K., "Navigation Analysis for Eros Rendezvous and Orbital Phases," *Journal of the Astronautical Sciences*, Vol. 43, No. 4, 1995, pp. 453–476.

[4] Werner, R. A., and Scheeres, D. J., "Exterior Gravitation of a Polyhedron Dervied and Compared with Harmonic and Mascon Gravitation Representations of Asteroid 4769 Castalia," *Celestial Mechanics and Dynamical Astronomy*, Vol. 65, No. 3, 1996, pp. 313–344, doi:10.1007/BF00053511.

[5] Werner, R. A., "The Gravitational Potential of a Homogeneous Polyhedron or Don't Cut Corners," *Celestial Mechanics and Dynamical Astronomy*, Vol. 59, No. 3, 1994, pp. 253–278. doi:10.1007/BF00692875

[6] Cangahuala, L. A., "Augmentations to the Polyhedral Gravity Model to Facilitate Small Body Navigation," *AAS/AIAA Space Flight Mechanics Meeting, American Astronautical Society and American Institute of Aeronautics and Astronautics*, AAS/AIAA, San Diego, CA, 2005, pp. 685–698; also AAS Paper 05-146.

[7] Weeks, C., and Miller, J. K., "A Gravity Model for Navigation Close to Asteroids and Comets," *Journal of the Astronautical Sciences*, Vol. 52, No. 3, 2004, pp. 381–389.

[8] Junkins, J. L., "Investigation of Finite-Element Representations of the Geopotential," *AIAA Journal*, Vol. 14, No. 6, 1976, pp. 803–808.

[9] Engels, R. C., and Junkins, J. L., "Local Representation of the Geopotential by Weighted Orthonormal Polynomials," *Journal of Guidance and Control*, Vol. 3, No. 1, 1980, pp. 55–61.

[10] Gasca, M., and Sauer, T., "Polynomial Interpolation in Several Variables," *Advances in Computational Mathematics*, Vol. 12, No. 4, 2000, pp. 377–410. doi:10.1023/A:1018981505752

[11] Karniadakis, G. E., and Sherwin, S., *Spectral/hp Element Methods for Computational Fluid Dynamics*, Oxford Science Publications, New York, 2005, pp. 15–95, Chaps. 2, 3.

[12] Berrut, J.-P., and Trefethen, L. N., "Barycentric Lagrange Interpolation," *SIAM Review*, Vol. 46, No. 3, 2004, pp. 501–517. doi:10.1137/S0036144502417715

[13] Hudson, R. S., and Ostro, S. J., "Shape of Asteroid 4769 Castalia (1989 PB) from Inversion of Radar Images," *Science*, Vol. 263, No. 5149, 1994, pp. 940–943. doi:10.1126/science.263.5149.940

[14] Samet, H., *The Design and Analysis of Spatial Data Structures*, Addison Wesley Longman, Reading, MA, 1990.

[15] Frisken, S. F., and Perry, R. N., "Simple and Efficient Traversal Methods for Quadtrees and Octrees," *Journal of Graphics Tools*, Vol. 7, No. 3, 2002, pp. 1–11.

[16] Ostro, S. J., Hudson, R. S., Benner, L. A. M., Nolan, M. C., Giorgini, J. D., Scheeres, D. J., Jurgens, R. F., and Rose, R., "Radar Observations of Asteroid 1998 ML14," *Meteoritics and Planetary Science*, Vol. 36, No. 9, 2001, pp. 1225–1236.

[17] Snir, M., Otto, S., Walker, D., Dongarra, J., and Huss-Lederman, S., *MPI: The Complete Reference*, MIT Press, Cambridge, MA, 1995.

[18] Laskar, J., "Frequency Analysis for Multi-Dimensional Systems. Global Dynamics and Diffusion," *Physica D*, Vol. 67, Nos. 1–3, 1993, pp. 257–281. doi:10.1016/0167-2789(93)90210-R

[19] Lara, M., Russell, R., and Villac, B., "Fast Estimation of Stable Regions in Real Models," *Meccanica*, Vol. 42, No. 5, 2007, pp. 511–550. doi:10.1007/s11012-007-9060-z

[20] Smoljak, S. A., "Quadrature and Interpolation Formulas for Tensor Products of Certain Classes of Functions," *Soviet Mathematics*, Vol. 4, No. 1, 1963, pp. 240–243.