Aerosp. Sci. Technol. 5 (2001) 209–220

S1270-9638(01)01096-3/FLA

State interpolation for on-board navigation systems

Oliver Montenbruck*, Eberhard Gill

German Space Operations Center, Deutsches Zentrum für Luft- und Raumfahrt, 82230 Weßling, Germany

Received 24 July 2000; revised and accepted 25 January 2001

Abstract

Common concepts for autonomous on-board navigation systems rely on the numerical integration of a spacecraft trajectory between subsequent measurements of a navigation sensor such as GPS. In combination with a Kalman filter, a predicted state vector becomes available at discrete, but not necessarily equidistant time steps. When used for real-time attitude control or geo-coding of image data, the on-board navigation system has to provide continuous dense output at equidistant time steps, which usually conflicts with the natural stepsize of the relevant integration methods and the non-equidistant measurement times. To cope with this problem, the integrator has to be supplemented by an interpolation scheme of compatible order and accuracy.

After presenting a representative formulation of an on-board navigation system and deriving related timing and accuracy requirements, suitable Runge–Kutta methods and associated interpolants are selected and evaluated. Promising results are obtained for the classical RK4 method in combination with Richardson extrapolation and 5th-order Hermite interpolation. The 5th-order Fehlberg method with interpolation due to Enright and, for drag-free scenarios, the 5th-order Runge–Kutta–Nystrom method with 5th-order Hermite interpolation provide a good performance in terms of position interpolation. However, as both methods exhibit significant errors for the velocity interpolation, they are not recommended for use with the outlined navigation filter. © 2001 Éditions scientifiques et médicales Elsevier SAS

on-board navigation / interpolation / Runge–Kutta method / Kalman filter / GPS

Zusammenfassung

Zustandsinterpolation für bordgestütze Navigationssysteme. Autonome bordgestützte Navigationssysteme verwenden in der Regel numerische Integrationsverfahren, um die Satellitenbahn zwischen aufeinanderfolgenden Messungen eines Navigationssensors, wie z.B. GPS, zu berechnen. Werden diese Verfahren mit einem Kalman-Filter kombiniert, können Zustandsvektoren, bestehend aus Position und Geschwindigkeit des Satelliten, vorhergesagt werden, die an diskreten, aber nicht notwendig äquidistanten, Zeitpunkten vorliegen. Wird das Navigationssystem jedoch für die Echtzeit-Lageregelung oder die Geokodierung von Bilddaten eingesetzt, muss eine kontinuierliche und äquidistante Ausgabe der Zustandsvektoren erfolgen. Diese Anforderung widerspricht in der Regel der natürlichen Schrittweite des Integrators und den nichtäquidistanten Messzeitpunkten. Zur Lösung dieses Problems kann der Integrator durch ein geeignetes Interpolationsverfahren ergänzt werden, welches eine dem Integrator entsprechende Ordnung und Genauigkeit aufweist.

Zunächst wird ein typisches Konzept eines bordgestützten Navigationssystems vorgestellt, aus dem sich die Anforderungen hinsichtlich des zeitlichen Ablaufs und der erforderlichen Genauigkeit ableiten lassen. Im Anschluss werden geeignete Runge–Kutta-Methoden mit den jeweiligen Interpolanten ausgewählt, dargestellt und ausgewertet. Dabei ergeben sich gute Ergebnisse für das klassische RK4-Verfahren mit Richardson-Extrapolation und einer Hermit-Interpolation 5. Ordnung. Das Fehlberg-Verfahren 5. Ordnung mit einer Interpolation von Enright sowie, für Szenarien ohne atmosphärische Reibung, das Runge–Kutta Nystrom-Verfahren 5. Ordnung mit einer Hermit-Interpolation 5. Ordnung ergeben gute Ergebnisse für die Interpolation der Position. Aufgrund der beträchtlichen Interpolationsfehler für die Geschwindigkeit sind jedoch beide

* Correspondence and reprints.
E-mail address: oliver.montenbruck@dlr.de (O. Montenbruck).

Verfahren für das hier zugrundegelegte Filterkonzept nicht geeignet.

Bordgestütze Navigation / Interpolation / Runge–Kutta Verfahren / Kalman Filter / GPS

1. Introduction

The basic purpose of an on-board navigation system consists in the real-time provision of orbit related information. Its typical output comprises position and velocity vectors at discrete time steps which may be used for a variety of purposes. Representative applications include the autonomous computation of the nadir and flight direction for attitude control [8], the prediction of contact times, the pointing of cameras [15] and antennas, the geocoding of image data [8] as well as autonomous orbital control [27]. Each on-board navigation system is necessarily based on some space-borne sensor that is sensitive to the position or velocity of the spacecraft. Among the available systems, GPS receivers have gained wide acceptance, but other tracking devices like DORIS [22], PRARE [20] or TDRSS [9] may likewise be applied. Aside from moderate hardware requirements, GPS is particularly attractive for the design of simple navigation systems through the provision of pre-computed position and velocity data in addition to the raw pseudo-range and phase measurements.

To derive the desired position and velocity information, the measurements are processed in a Kalman filter, which serves a three-fold purpose: it allows the processing of measurements that contain only partial information on the spacecraft state vector (such as range or pseudo-range measurements), it reduces the inherent measurement noise by averaging over multiple measurements, and it constrains the estimated trajectory to a dynamical model. Representative algorithms are described in the published literature (see [19] and references therein) and will not be repeated here. Ignoring the special case of epoch state filters, all common Kalman filter implementations comprise a state update, in which the latest state estimate (as well as its covariance) is propagated to the time of the new measurement and a measurement update, in which a new measurement is merged with the existing information to create an improved estimate of the state vector. Traditionally, numerical integration methods and dynamical models with varying levels of sophistication are used to propagate the state vector between subsequent measurement times.

The above formulation of the Kalman filter process implies that new information is only available after discrete time steps that coincide with the times of new measurements. Evidently, some form of interpolation is desirable to fully decouple the navigation system input and output, which may be required at completely independent times. In particular, interpolation is required if the overall computation time of a single Kalman filter step exceeds the envisaged time between subsequent state vector outputs. Furthermore, the execution and synchronization of multiple real-time tasks inside an on-board processor requires a careful allocation of process start times and durations. It is, therefore, apparent that a Kalman filter process driven by non-equidistant measurements may be in conflict with the overall scheduling of processing resources unless an interpolant is applied.

In order to establish requirements and constraints for the selection of suitable integrators and interpolants, a representative real-time navigation process is considered in *figure 1*. It is assumed that the process is invoked at discrete times t_i and requires a maximum computation time Δt_{comp}. Upon initiating the ith step, a continuous representation of the trajectory T_{i-1} between t_{i-1} and t_i is available from the previous step, if we neglect the startup phase during which this information has to be generated from an a priori state vector. Using T_{i-1}, a predicted state vector (and covariance) at the time $t_{\mathrm{upd},i}$ of the latest measurement prior to t_i is obtained. Here, a filter update is performed that yields a best estimate of the instantaneous state vector based on the predicted state and the tracking data. The updated state is then integrated to the time $t_{i+1} + \Delta t_{\mathrm{comp}}$, i.e. the guaranteed end time of the following navigation step. Furthermore, an interpolant is constructed along with the numerical integration, which provides a continuous description of the trajectory throughout the time interval $[t_{\mathrm{upd},i}, t_{i+1} + \Delta t_{\mathrm{comp}}]$.

Assuming that the navigation process is activated at equidistant times, separated by an interval Δt, it is evident that the size of each integration step as well as the time covered by the interpolant can vary between a minimum of $\Delta t + \Delta t_{\mathrm{comp}}$ and a maximum of $2\Delta t + \Delta t_{\mathrm{comp}}$. Clearly, the order and accuracy of the applied numerical methods must be suitably adjusted to the applied duty cycle. Since at most one measurement is processed in each step (depending on the actual availability of data between t_{i-1} and t_i), the interval Δt also determines the total amount of data used for the trajectory adjustment. Subject to an adequate dynamical model, a moderate number of 50–200 measurements per orbit is fully sufficient for a reliable state estimation, yielding representative stepsizes of 30–120 seconds for a low Earth orbit with a hundred minutes orbital period. For high altitude missions like geostationary satellites or navigation satellites like GPS, these values may be

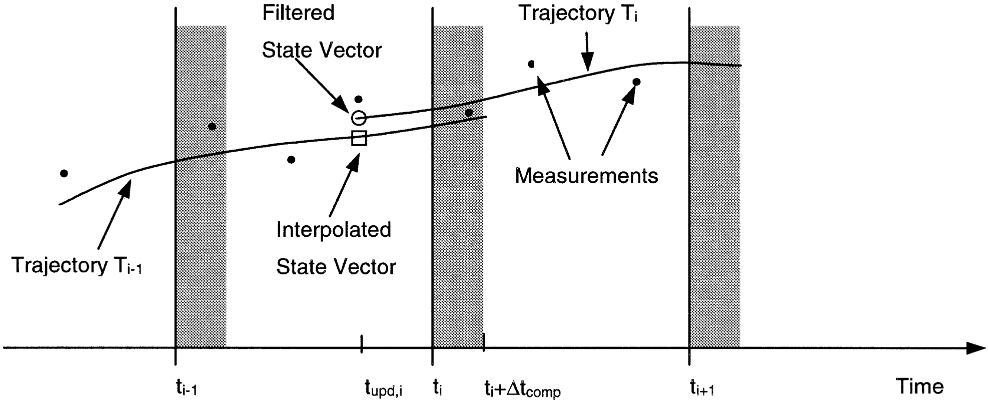

Figure 1. Timeline of real-time navigation system. Shaded bars indicate computational activity.

scaled accordingly. It is emphasized that any increase in the number of processed measurements over the 'optimal' value will result in a corresponding increase of the total workload of the navigation process, since each measurement update necessitates an integrator restart.

Each measurement processed by the Kalman filter implies the use of new initial conditions in the trajectory integration. Therefore, classical high-order and multi-step methods that are otherwise preferred in astrodynamic applications [18] are hardly useful for real-time navigation systems. Instead, low-order single-step methods should be selected for which the 'natural' stepsize is in the order of the selected filter duty cycle. Considering again a near-circular low-Earth orbit, a one minute stepsize h corresponds to 0.01 revolutions or 0.063 radians. Assuming that the leading error term of a pth-order method is approximately $h^{(p+1)}/(p+1)!$, one finds that a 4th-order method yields a relative error of 10^{-8} per step, which corresponds to 7 cm at the given orbital altitude. Based on these preliminary considerations, we focus on single-step methods with orders 3 to 5 in the subsequent discussion. While lower order methods have difficulties to meet the required accuracy, the performance of higher order methods cannot be utilized in view of stepsize restrictions implied by the measurement updates.

2. Integration methods

In the sequel a selection of appropriate single-step methods is presented, all of which belong to the family of Runge–Kutta methods. Aside from the methods themselves, the concept of Richardson extrapolation is addressed, which is useful both to increase the order of a method and to provide solutions suitable for higher order interpolation.

The spacecraft motion is represented by a first-order differential equation

$$\dot{\boldsymbol{y}} = \boldsymbol{f}\big(t, \boldsymbol{y}(t)\big) \tag{1}$$

for the position-velocity vector or state vector

$$\boldsymbol{y}(t) = \begin{pmatrix} \boldsymbol{r}(t) \\ \boldsymbol{v}(t) \end{pmatrix} \tag{2}$$

as a function of time t or the equivalent second-order formulation

$$\ddot{\boldsymbol{r}}(t) = \boldsymbol{a}\big(t, \boldsymbol{r}(t), \boldsymbol{v}(t)\big) \tag{3}$$

in case of RK–Nystrom methods.

2.1. Low-order Runge–Kutta methods

The well-known four stage Runge–Kutta method (RK4) provides a 4th-order approximation

$$\boldsymbol{\eta}(t+h) = \boldsymbol{y}(t) + h\sum_{i=1}^{4} b_i \boldsymbol{k}_i \tag{4}$$

of $\boldsymbol{y}(t+h)$ with derivatives

$$\boldsymbol{k}_i = \boldsymbol{f}\left(t + c_i h,\ \boldsymbol{y}(t) + h\sum_{j=1}^{i-1} a_{ij}\boldsymbol{k}_j\right) \tag{5}$$

and coefficients as given in *table I.* The method is primarily considered here because of its widespread use and the simple coefficient set. Alternative 4th-order methods (see [10]) include the method of Gill (RKG4), which requires less storage registers but uses a non-rational coefficient set and the '3/8-rule' method. Both four-stage methods revealed a slightly lower performance when applied to the test problem described in section 3 and can therefore be ruled out as alternatives to the classical RK4 method. Finally, we mention the 4th-order method of Kutta–Merson (KM4) which requires a total of 5 stages and offers an embedded method for error control.

Table I. RK4 coefficient set.

c_i	a_{ij}			
0	0			
$\frac{1}{2}$	$\frac{1}{2}$			
$\frac{1}{2}$	0	$\frac{1}{2}$		
1	0	0	1	
b_i	$\frac{1}{6}$	$\frac{2}{6}$	$\frac{2}{6}$	$\frac{1}{6}$

It achieves a higher accuracy per step and a slightly better efficiency (i.e. accuracy per stage) than RK4.

Among the various fifth-order Runge–Kutta methods, we have considered an embedded six-stage method (RKF5), which is due to Fehlberg [7] and employs the coefficients from *table II*. The method provides both a fourth-order solution (described by b_i) and a fifth-order solution (described by $\hat{b}_i$). The difference of both values is useful to assess the accuracy of the lower order method and to control the stepsize of the integration in accord with given accuracy requirements. For on-board applications, stepsize control is generally out of scope and the

Table II. RKF5 coefficient set.

c_i	a_{ij}						
0	0						
$\frac{1}{4}$	$\frac{1}{4}$						
$\frac{3}{8}$	$\frac{3}{32}$	$\frac{9}{32}$					
$\frac{12}{13}$	$\frac{1932}{2197}$	$\frac{-7200}{2197}$	$\frac{7296}{2197}$				
1	$\frac{439}{216}$	-8	$\frac{3680}{513}$	$\frac{-845}{4104}$			
$\frac{1}{2}$	$\frac{-8}{27}$	2	$\frac{-3544}{2565}$	$\frac{1859}{4104}$	$\frac{-11}{40}$		
b_i	$\frac{25}{216}$	0	$\frac{1408}{2565}$	$\frac{2197}{4104}$	$\frac{-1}{5}$	(4th-order)	
$\hat{b}_i$	$\frac{16}{135}$	0	$\frac{6656}{12825}$	$\frac{28561}{56430}$	$\frac{-9}{50}$	$\frac{2}{55}$	(5th-order)

Table III. DP5 coefficient set.

c_i	a_{ij}							
0	0							
$\frac{1}{5}$	$\frac{1}{5}$							
$\frac{3}{10}$	$\frac{3}{40}$	$\frac{9}{40}$						
$\frac{4}{5}$	$\frac{44}{45}$	$\frac{-56}{15}$	$\frac{32}{9}$					
$\frac{8}{9}$	$\frac{19372}{6561}$	$\frac{-25630}{2187}$	$\frac{64448}{6561}$	$\frac{-212}{729}$				
1	$\frac{9017}{3168}$	$\frac{-355}{33}$	$\frac{46732}{5247}$	$\frac{49}{176}$	$\frac{-5103}{18656}$			
1	$\frac{35}{384}$	0	$\frac{500}{1113}$	$\frac{125}{192}$	$\frac{-2187}{6784}$	$\frac{11}{84}$		
$\hat{b}_i$	$\frac{35}{384}$	0	$\frac{500}{1113}$	$\frac{125}{192}$	$\frac{-2187}{6784}$	$\frac{11}{84}$	(5th-order)	
b_i	$\frac{5179}{57600}$	0	$\frac{7571}{16695}$	$\frac{393}{640}$	$\frac{-92097}{339200}$	$\frac{187}{2100}$	$\frac{1}{40}$	(4th-order)

higher order solution itself can be used for integrating the equation of motion. In the present study, the Fehlberg method is of prime interest due to the availability of various interpolants.

In addition, we consider the embedded 5th/4th-order method of Dormand and Prince [3,10], which requires a total of seven stages and likewise allows the construction of smooth interpolants. It is described by the coefficients from *table III*. If stepsize control is not required, six stages are sufficient to compute the fifth-order solution (DP5), which then makes the method slightly superior to the RKF5 code.

2.2. Runge–Kutta–Nystrom methods

Runge–Kutta–Nystrom methods provide an efficient means for solving second-order differential equations

$$\ddot{\boldsymbol{r}}(t) = \boldsymbol{f}\big(t, \boldsymbol{r}(t)\big) \tag{6}$$

that do not depend on first-order derivatives. This is, for example, the case, if the equation of motion comprises gravitational and radiation forces only, but no drag perturbations. A method developed by Nystrom (see [10]) will be considered in the sequel, which employs the coefficients from *table IV* to compute the fifth-order solution

$$\boldsymbol{\eta}(t+h) = \boldsymbol{y}(t) + \begin{pmatrix} h\boldsymbol{v}(t) \\ 0 \end{pmatrix} + \sum_{i=1}^{4} \begin{pmatrix} h^2 b_i \boldsymbol{k}_i \\ h\hat{b}_i \boldsymbol{k}_i \end{pmatrix} \tag{7}$$

with

$$\boldsymbol{k}_i = \boldsymbol{f}\left(t + c_i h,\ \boldsymbol{r}(t) + hc_i \boldsymbol{v}(t) + h^2 \sum_{j=1}^{i-1} a_{ij} \boldsymbol{k}_j\right). \tag{8}$$

Table IV. RKN5 coefficient set.

c_i	a_{ij}			
0	0			
$\frac{1}{5}$	$\frac{1}{50}$			
$\frac{2}{3}$	$-\frac{1}{27}$	$\frac{7}{27}$		
1	$\frac{3}{10}$	$\frac{-2}{35}$	$\frac{9}{35}$	
b_i	$\frac{14}{336}$	$\frac{100}{336}$	$\frac{54}{336}$	
$\hat{b}_i$	$\frac{14}{336}$	$\frac{125}{336}$	$\frac{162}{336}$	$\frac{35}{336}$

2.3. Richardson extrapolation

The local truncation error

$$\boldsymbol{e} = \boldsymbol{\eta}(t+h) - \boldsymbol{y}(t+h) \approx \boldsymbol{C} \cdot h^{p+1} \tag{9}$$

of a pth-order method is governed by a leading error term that is proportional to h^{p+1}. By performing two consecutive integration steps of size h (yielding $\boldsymbol{\eta}_1$ and $\boldsymbol{\eta}_2$) by comparing the result $\boldsymbol{\eta}_H$ with a single macro-step of size $H = 2h$, the constant $\boldsymbol{C}$ can be determined and used to obtain improved solutions

$$\hat{\boldsymbol{\eta}}_1 = \boldsymbol{\eta}_1 + \frac{\boldsymbol{\eta}_2 - \boldsymbol{\eta}_H}{2(2^p - 1)}, \qquad \hat{\boldsymbol{\eta}}_2 = \boldsymbol{\eta}_2 + \frac{\boldsymbol{\eta}_2 - \boldsymbol{\eta}_H}{2^p - 1} \tag{10}$$

(see e.g. [10]). The derivative $\boldsymbol{k}_0 = \boldsymbol{f}(t, \boldsymbol{y}(t))$ is jointly used in both the first micro-step h and the macro-step H. Richardson extrapolation thus increases the order of the underlying method by one at the expense of an additional $(s-1)/2$ stages per micro-step. It is particularly useful with 4th-order Runge–Kutta methods, since the resulting 5th-order method requires only 5½ stages as compared to the Butcher barrier of six stages for a 5th-order Runge–Kutta method.

3. Interpolants

3.1. Hermite interpolation

Independent of a particular integration method, a cubic Hermite approximation polynomial

$$\begin{aligned} \boldsymbol{y}(t+\theta h) = {} & d_0(\theta)\boldsymbol{y}_0 + d_1(\theta)h\boldsymbol{f}_0 + d_2(\theta)\boldsymbol{y}_1 \\ & + d_3(\theta)h\boldsymbol{f}_1 \end{aligned} \tag{11}$$

with $0 < \theta < 1$ and coefficients

$$\begin{aligned} d_0 &= (\theta-1)^2(2\theta+1), \qquad d_1 = \theta(\theta-1)^2, \\ d_2 &= \theta^2(3-2\theta), \qquad d_3 = \theta^2(\theta-1), \end{aligned} \tag{12}$$

can always be established that matches the state vectors $\boldsymbol{y}_0 = \boldsymbol{y}(t)$and $\boldsymbol{y}_1 = \boldsymbol{y}(t+h)$ as well as the associated derivatives $\boldsymbol{f}_i$ at the begin and end of the integration step [16]. It requires knowledge of $\boldsymbol{f}_1 = \boldsymbol{f}(t+h, \boldsymbol{y}_1)$ in addition to quantities already available from the integrator. While it is commonly emphasized that $\boldsymbol{f}_1$ can be reused in a subsequent integration step from $t+h$ to $t+2h$, care must be taken, since the navigation scheme does not, necessarily, provide this possibility in case of a measurement update.

Using the same information as above, a higher-order interpolant can be constructed by taking into account the particular nature of the equation of motion. Considering the fact that the six-dimensional state vector and its derivative comprise information on the position, velocity, and acceleration one may establish a fifth-order Hermite polynomial

$$\begin{aligned} \boldsymbol{r}(t+\theta h) &= d_0(\theta)\boldsymbol{r}_0 + d_1(\theta)h\boldsymbol{v}_0 + d_2(\theta)h^2\boldsymbol{a}_0 \\ &\quad + d_3(\theta)\boldsymbol{r}_1 + d_4(\theta)h\boldsymbol{v}_1 + d_5(\theta)h^2\boldsymbol{a}_1 \end{aligned} \tag{13}$$

with coefficients

$$\begin{aligned} d_0 &= 1 - 10\theta^3 + 15\theta^4 - 6\theta^5, \\ d_1 &= \theta - 6\theta^3 + 8\theta^4 - 3\theta^5, \\ d_2 &= \frac{1}{2}\big(\theta^2 - 3\theta^3 + 3\theta^4 - \theta^5\big), \\ d_3 &= 10\theta^3 - 15\theta^4 + 6\theta^5 = 1 - d_0, \\ d_4 &= -4\theta^3 + 7\theta^4 - 3\theta^5, \\ d_5 &= \frac{1}{2}\big(\theta^3 - 2\theta^4 + \theta^5\big), \end{aligned} \tag{14}$$

for the spacecraft position that matches these values at the begin and end of the integration interval. Upon differentiation a corresponding fourth-order interpolant

$$\begin{aligned} \boldsymbol{v}(t+\theta h) &= d_0(\theta)\boldsymbol{r}_0/h + d_1(\theta)\boldsymbol{v}_0 + d_2(\theta)h\boldsymbol{a}_0 \\ &\quad + d_3(\theta)\boldsymbol{r}_1/h + d_4(\theta)\boldsymbol{v}_1 + d_5(\theta)h\boldsymbol{a}_1 \end{aligned} \tag{15}$$

for the velocity vector is obtained which employs the coefficients

$$\begin{aligned} d_0 &= -30\theta^2 + 60\theta^3 - 30\theta^4, \\ d_1 &= 1 - 18\theta^2 + 32\theta^3 - 15\theta^4, \\ d_2 &= \frac{1}{2}\big(2\theta - 9\theta^2 + 12\theta^3 - 5\theta^4\big), \\ d_3 &= +30\theta^2 - 60\theta^3 + 30\theta^4 = -d_0, \\ d_4 &= -12\theta^2 + 28\theta^3 - 15\theta^4, \\ d_5 &= \frac{1}{2}\big(3\theta^2 - 8\theta^3 + 5\theta^4\big). \end{aligned} \tag{16}$$

Finally, when combining data from two consecutive integration steps, a quintic Hermite polynomial

$$\begin{aligned} \boldsymbol{y}(t+\theta h) &= d_0(\theta)\boldsymbol{y}_0 + d_1(\theta)h\boldsymbol{f}_0 + d_2(\theta)\boldsymbol{y}_1 \\ &\quad + d_3(\theta)h\boldsymbol{f}_1 + d_4(\theta)\boldsymbol{y}_2 + d_5(\theta)h\boldsymbol{f}_2 \end{aligned} \tag{17}$$

with coefficients

$$\begin{aligned} d_0 &= \frac{1}{4}(\theta-1)^2(\theta-2)^2(1+3\theta), \\ d_1 &= \frac{1}{4}\theta(\theta-1)^2(\theta-2)^2, \\ d_2 &= \theta^2(\theta-2)^2, \\ d_3 &= (\theta-1)\theta^2(\theta-2)^2, \\ d_4 &= \frac{1}{4}\theta^2(\theta-1)^2(7-3\theta), \\ d_5 &= \frac{1}{4}(\theta-2)\theta^2(\theta-1)^2, \end{aligned} \tag{18}$$

(cf. [23]) becomes available that provides a 5th-order approximation over the interval $[t, t+2h]$. This is particularly attractive for use with Richardson extrapolation and the RK4 method, since it results in a consistent 5th-order integrator and interpolant.

3.2. Scaled Runge–Kutta methods

Following Horn [13,14] various Runge–Kutta methods can be supplemented by an embedded interpolant

$$\boldsymbol{\eta}(t+\theta h) = \boldsymbol{y}(t) + \theta h \sum_{i=1}^{s^*} b_j^* \boldsymbol{k}_i \tag{19}$$

that utilizes the derivatives $\boldsymbol{k}_1, \ldots, \boldsymbol{k}_s$ of the basic method together with a limited set of additional derivatives $\boldsymbol{k}_{s+1}, \ldots, \boldsymbol{k}_{s^*}$. Building up on the RKF5 method described above, the introduction of

$$\boldsymbol{k}_7 = \boldsymbol{f}\left(t+h, \boldsymbol{y}(t) + h\left(\frac{1}{6}\boldsymbol{k}_1 + \frac{1}{6}\boldsymbol{k}_5 + \frac{2}{3}\boldsymbol{k}_6\right)\right) \tag{20}$$

allows the construction of a continuous 4th-order method with coefficients

$$\begin{aligned} b_1^* &= 1 - \theta\left(\frac{301}{120} + \theta\left(-\frac{269}{108} + \theta\frac{311}{360}\right)\right), \\ b_2^* &= 0, \\ b_3^* &= \theta\left(\frac{7168}{1425} + \theta\left(-\frac{4096}{513} + \theta\frac{14848}{4275}\right)\right), \\ b_4^* &= \theta\left(-\frac{28561}{8360} + \theta\left(+\frac{199927}{22572} - \theta\frac{371293}{75240}\right)\right), \\ b_5^* &= \theta\left(\frac{57}{50} + \theta\left(-3 + \theta\frac{42}{25}\right)\right), \\ b_6^* &= \theta\left(-\frac{96}{55} + \theta\left(\frac{40}{11} - \theta\frac{102}{55}\right)\right), \\ b_7^* &= \theta\left(\frac{3}{2} + \theta\left(-4 + \theta\frac{5}{2}\right)\right). \end{aligned} \tag{21}$$

The order of the interpolant is thus one order less than that of the integrator, but still one order higher than that of a cubic Hermite polynomial.

For various 5th-order method of Dormand and Prince, embedded 4th-order interpolants are given in [4,10,21].

3.3. Bootstrap methods

A generic method for constructing higher order interpolants for Runge–Kutta methods is given by Enright et al. [5]. It extends the concept of Hermite interpolation by using additional function values within the integration steps to obtain approximations of higher order. In case of the RKF5 method, a 5th-order approximation

$$\begin{aligned}\boldsymbol{y}_{0.6} = \boldsymbol{y}(t) + h\biggl(&\frac{1559}{12500}\boldsymbol{k}_1 + \frac{153856}{296875}\boldsymbol{k}_3\\ &+ \frac{68107}{2612500}\boldsymbol{k}_4 - \frac{243}{31250}\boldsymbol{k}_5 - \frac{2106}{34375}\boldsymbol{k}_6\biggr) \qquad (22)\end{aligned}$$

of the state vector at $t + 0.6h$ is first computed, which is then used to establish a quartic Hermite polynomial

$$\begin{aligned}\boldsymbol{y}(t+\theta h) = {}& d_0(\theta)\boldsymbol{y}_0 + d_1(\theta)h\boldsymbol{f}_0 + d_2(\theta)\boldsymbol{y}_1\\ &+ d_3(\theta)h\boldsymbol{f}_1 + d_4(\theta)\boldsymbol{y}_{0.6} \qquad (23)\end{aligned}$$

with

$$\begin{aligned}d_0 &= (\theta-1)^2\left(1-\frac{5}{3}\theta\right)\left(\frac{11}{3}\theta+1\right),\\ d_1 &= \theta(\theta-1)^2\left(1-\frac{5}{3}\theta\right),\\ d_2 &= \theta^2\left(\frac{3}{4}-\frac{5}{4}\theta\right)(9\theta-11), \qquad (24)\\ d_3 &= \theta^2(\theta-1)\left(\frac{5}{2}\theta-\frac{3}{2}\right),\\ d_4 &= \frac{625}{36}\theta^2(\theta-1)^2.\end{aligned}$$

The resulting interpolant has the highest order attainable without an additional function evaluation and is continuous at the end points of the integration interval. In case of higher-order Runge–Kutta methods, a consistent interpolant can be constructed in a bootstrap fashion, by successively adding additional intermediate points and deriving associated higher-order Hermite polynomials [5].

The same approach has been applied by [24] to derive a 4th-order interpolant for the 5th-order Runge–Kutta method of Dormand and Prince. It makes use of a fifth-order result

$$\begin{aligned}\boldsymbol{y}_{0.5} = \boldsymbol{y}(t) + 0.5h\biggl(&\frac{7157}{37888}\boldsymbol{k}_1 + \frac{70925}{82362}\boldsymbol{k}_3 + \frac{10825}{56832}\boldsymbol{k}_4\\ &- \frac{220887}{2008064}\boldsymbol{k}_5 + \frac{80069}{1765344}\boldsymbol{k}_6\\ &- \frac{107}{2627}\boldsymbol{k}_7 - \frac{5}{37}\boldsymbol{k}_8\biggr) \qquad (25)\end{aligned}$$

with

$$\begin{aligned}\boldsymbol{k}_8 = \boldsymbol{f}\biggl(t+\frac{h}{2}, \boldsymbol{y}(t) + h\biggl(&-\frac{33728713}{104693760}\boldsymbol{k}_1 + 2\boldsymbol{k}_2\\ &- \frac{30167461}{21674880}\boldsymbol{k}_3 + \frac{7739027}{17448960}\boldsymbol{k}_4\\ &- \frac{19162737}{123305984}\boldsymbol{k}_5 - \frac{26949}{363520}\boldsymbol{k}_7\biggr)\biggr) \qquad (26)\end{aligned}$$

at the midpoint $t + h/2$, which is then used together with the end-points to construct a quartic polynomial across the interval.

3.4. Keplerian interpolation

Given the special nature of the equation of motion, an entirely different interpolant is given by the osculating Keplerian orbit that matches the state vector at the beginning of the integration step. Relevant equations for the mutual conversion between state vectors and orbital elements are given in the literature (see [6,19]) and will not be reproduced here. Compared to the polynomial approximations discussed before, the Keplerian orbit approximation requires a notably higher computational effort, but is not restricted to a single integration step.

4. Analysis and results

The various integration methods and associated interpolants presented above were applied to a representative low-Earth orbit prediction problem to assess their accuracy over time scales of up to one orbital revolution. Initial conditions, as given in *table V*, were employed to integrate a reference trajectory with the variable-order predictor-corrector method of Shampine and Gordon [25]. A fidelity force model comprising a 20 × 20 Earth gravity field as well as luni-solar gravity, atmospheric drag (except for use with Nystrom methods), and solar radiation pressure was employed to obtain a realistic scenario. The selected model provides meter level accuracy in orbit predictions over at least one revolution at the given altitude of 750 km orbit (see e.g. [19]) and is representative of advanced on-board navigation systems. In view of the fundamental time scales involved, it may safely be assumed, however, that the relative merits of individual integrators and interpolants are independent of the particular choice of the force model.

The same equation of motion as used for the reference trajectory was subsequently integrated with each of the aforementioned low-order methods using different stepsizes and approximated by the applicable interpolants. Representative examples showing the increase of the global integration error over time are collated in *figure 2*, where all computations have been performed in IEEE 8-byte floating point arithmetic. Using a stepsize of 10 seconds per stage, the global integration error can be controlled within 0.1–10 m over one orbit revolution.

Table V. Reference orbit used for the comparison of low-order integration methods and interpolants.

Epoch	1996/10/01 00:00 UTC
x [m]	−2616512.77
y [m]	+5992529.01
z [m]	−2846280.49
vx [m/s]	−1449.266428
vy [m/s]	−3648.375664
$\boldsymbol{vz}$ [m/s]	−6356.361255

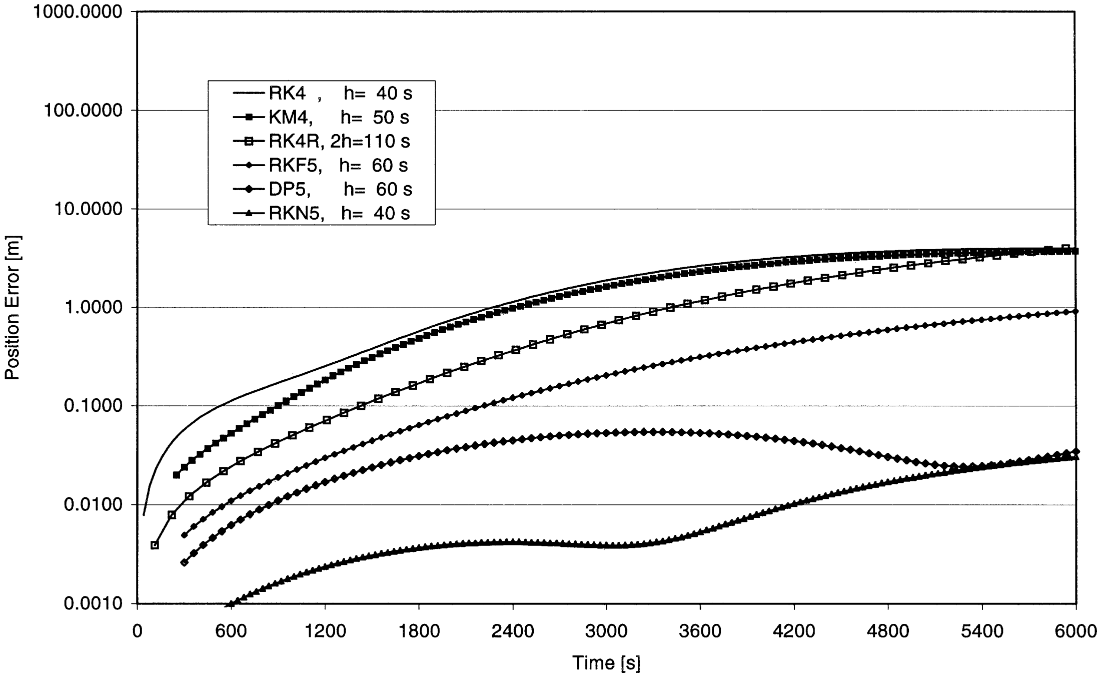

Figure 2. Global integration error for the integration of a low-Earth satellite orbit over $t = 100$ min using various integration methods. The stepsizes for each method have been selected such as to require an average of one stage per 10 s. The considered set of methods comprises the 4th-order methods of Runge–Kutta (RK4) and Kutta–Merson (KM4), the 4th-order Runge–Kutta method with Richardson extrapolation (RK4R), the 5th-order methods of Fehlberg (RKF5) and Dormand–Prince (DP5) as well as the 5th-order Runge–Kutta–Nystrom method, which are further described in the text.

To further assess the relative performance of the various methods during short prediction intervals, the total number of derivative computations required for an integration over $t = 720$ s is displayed in *figure 3* as a function of the achieved global accuracy. Evidently, all fifth-order methods considered are more efficient than any of the fourth-order Runge–Kutta schemes. Furthermore, the use of Nystrom methods provides a clear advantage over general purpose methods for first-order differential equations and is therefore recommended as long as drag forces need not to be accounted for. Extrapolation of the RK4 method yields a performance slightly inferior to that of the 5th-order Runge–Kutta–Fehlberg method, while the Dormand–Prince method is slightly more efficient, assuming that only the 5th-order solution is computed. The Kutta–Merson method turns out to be slightly more efficient than the classical RK4 method despite an equal order and a higher number of stages.

It is emphasized that all methods are able to meet a given accuracy requirement within the range of interest by proper adaptation of the stepsize. The associated computational workload, however, differs by up to a factor of three among the various integrators compared in *figure 3*. Even though all methods are generally applicable for on-board orbit predictions, the choice of a proper method can thus provide notable savings in on-board resources. Since the higher order integration methods achieve a given accuracy with larger stepsizes than their lower order counterparts, their use may, however, be constrained by independent requirements on the permissible filter update or output data rate.

Based on the above results, only fifth-order integration methods are considered in the subsequent comparison of available interpolants. In order to obtain a consistent combination and to preserve the efficiency of the underlying integration method, the interpolation error should preferably be smaller than the local integration error. As will be shown, compatible pairs may be obtained by combining a 5th-order integrator with either a 4th- or a 5th-order interpolant.

For the RK4 method with Richardson extrapolation, a 5th-order Hermite interpolation across the macro-step of size 2h provides a smooth and continuous interpolation of the trajectory (see *figure 4*, method RK4/Herm5). It makes use of the state vector and its derivative at the start, mid-point and end of the macro-step. In accord with [10] the state vector at the mid-point is corrected by the Richardson extrapolation, but no update of the derivative obtained after the first RK4 step is performed. Compared to the secular growth of the integration error itself, the resulting errors of the interpolated position exhibit an oscillation of roughly 2 mm for a macro stepsize of 2h = 120 s. This amplitude may further be reduced to about half its value, by recomputing the derivative

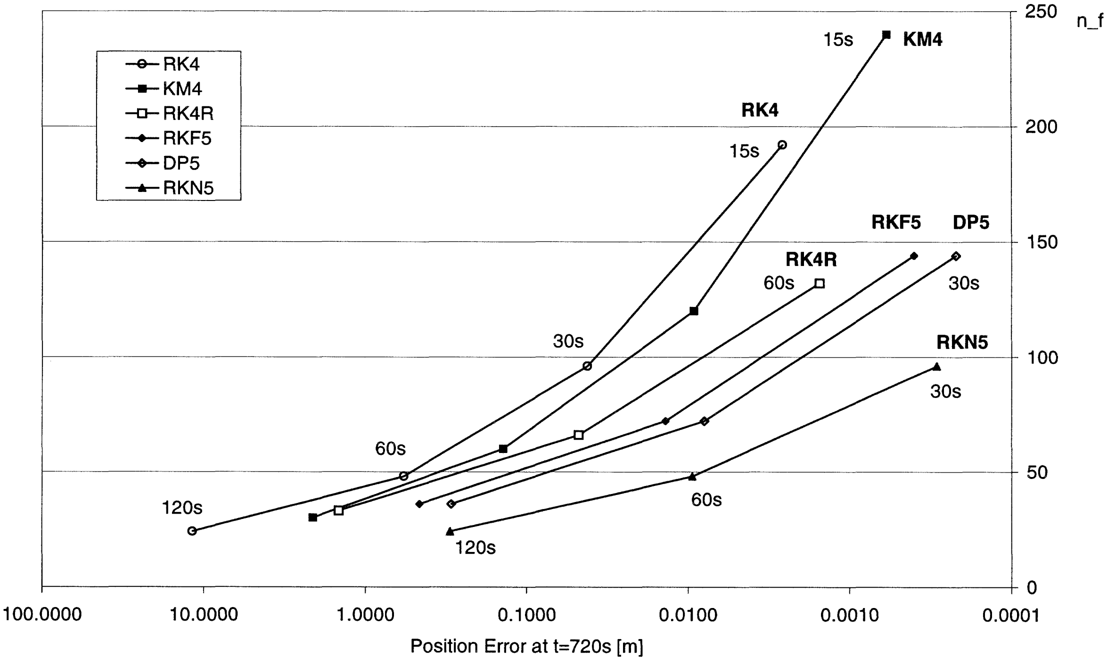

Figure 3. Work diagram of different integration methods. The total number of derivative computations is shown as a function of the achieved global integration error for the integration of a low-Earth satellite orbit over 720 s. See *figure 2* for a description of applied abbreviations.

$f_1(t+h, y_1)$ at the mid-point of the integration interval with the refined value of y_1 obtained from the extrapolation. Depending on the treatment of the mid-point derivative, a total of 12 or 13 stages per integration step are required for the generation of dense output.

For the fifth-order Runge–Kutta–Fehlberg method, the 4th-order interpolant suggested by Enright [5] is illustrated in *figure 4* (label RKF5/Enr4). At a stepsize of 60 s, it adds an error of about 1.5−2.5 mm to the global integration error. Thus it is clearly superior to the scaled 4th-order method of [13,14], which was found to exhibit a 2 times higher variation across an integration step. The interpolant requires the derivative at the end of the integration interval in addition to the six stages of the basic RKF5 method.

Among the various interpolants proposed for the 5th-order Dormand–Prince method preference is given to a quartic Hermite interpolation making use of Shampine's 5th-order solution at the mid-point of the integration interval [24] (see *figure 4*, label DP5/Sh4). Compared to any of the other 4th-order interpolants [4,10,24], this is the only method yielding an interpolation competetive to those of the RK4R and RKF5 integrators mentioned above. The construction of the interpolant is rather costly, however, adding three more stages to the basic six stage DP5 method.

Finally, an excellent interpolation of the spacecraft trajectory is provided by the 5th-order Hermite polynomial that matches position, velocity and acceleration at the begin and end of a 5th-order Runge–Kutta-Nystrom step (cf. *figure 4*, method RKN5/Herm5). It is particularly attractive, since it requires only one derivative computation at the end of the integration step in excess of the four stage RKN5 method itself.

Taking into account the total number of stages required for dense output in each of the aforementioned methods, the RKN5/Herm5 scheme clearly outperforms all integrators for first-order differential equations. Among the latter, a near equal performance is obtained with a slight preference for RKF5/Enr4, followed by RK4R/Herm5. A notably different picture is obtained, however, if velocity interpolation is considered in addition to dense output of position data (*figure 5*). Here, the quartic Hermite polynomial for the RKF5 method as well as the derivative of the quintic Hermite polynomial for the RKN5 method (which is of fourth order, only) yield interpolation errors of up to 0.04 mm/s in excess of the global integration error at a 60 s stepsize. While this appears to be a small quantity at first sight, it may result in unacceptably large errors when restarting the integration from an interpolated state vector. Therefore, neither of these two methods can be recommended for use with the onboard navigation filter outlined in *figure 1*. Among the remaining methods, the RK4R/Herm5 integrator yields a slightly higher efficiency and uses a more simple coefficient set than the DP5/Sh4 scheme.

Supplementary to the polynomial approximation of the Cartesian state vector, an interpolation based on Keplerian elements has been examined. To this end, the osculating Keplerian elements were computed at the begin-

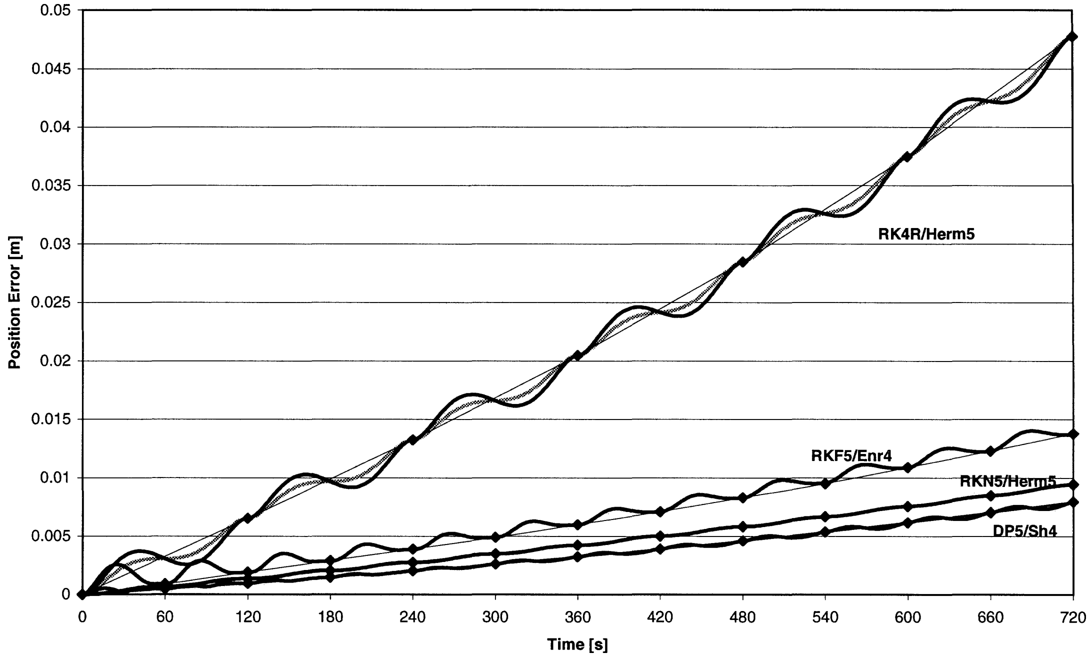

Figure 4. Position error of the interpolated trajectories for various integrators and interpolants. Diamonds connected by thin lines indicate the solution at the grid points. For comparison, all integrations have been performed with the same (micro-)step size of $h = 60$ s. The examined pairs of integrators/interpolants comprise the 4th-order Runge–Kutta method with Richardson extrapolation and 5th-order Hermite interpolation (RK4R/Herm5), the 5th-order Fehlberg method with a 4th-order interpolant of Enright (RKF5/Enr4), the 5th-order Runge–Kutta–Nystrom method with 5th-order Hermite interpolation (RKN5/Herm5) as well as the 5th-order Dormand–Prince method with 4th-order interpolant by Shampine (DP5/Sh4). The grey line refers to a modification of the 4th-order RK method with Richardson extrapolation and 5th-order Hermite interpolation, in which the mid-point derivative is recomputed after the extrapolation.

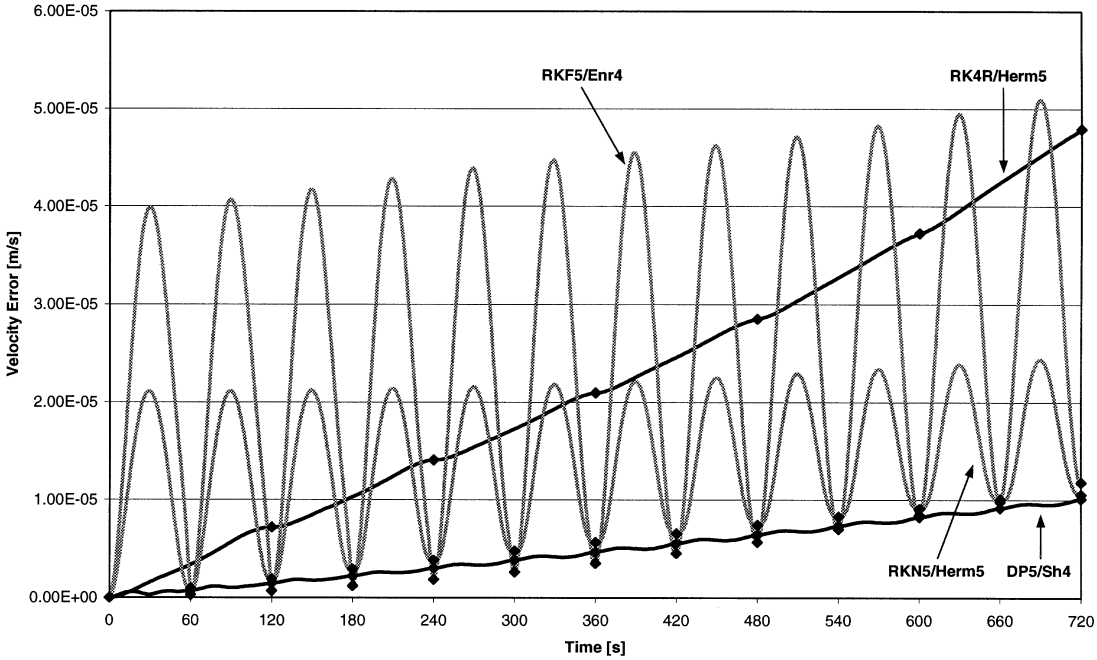

Figure 5. Velocity error of interpolated trajectories for various integrators and interpolants. See *figure 4* for a description of applied abbreviations.

Table VI. Integration methods applied in present on-board navigation systems.

System	Method	Step size	Authors/references
TONS	4th-order Runge–Kutta; dense output generation using RK4 and simplified force model	Filter update 16.384 s (Terra), 10 s (EUVE); output step 1 s	Lockheed Martin 1997 [17] Gramling et al. 2000 [9]
DIODE	4th-order Runge–Kutta–Gill with 3rd-order Hermite interpolation	Filter update 10 s	Pradines et al. 1994 [22] Tournier et al. 1999 [26]
DIOGENE	6th-order Runge–Kutta (variation of parameters formulation) with 3rd-order Hermite interpolation	Filter update 60 s (LEO), 5 min (GEO)	Berthias et al. 1997 [1]
GEODE	4th-order Runge–Kutta; dense output generation using RK4 and simplified force model	Filter update 30 s, output step 1 s	Hart et al. 1996 [11] Hart et al. 1996 [12]
CHAMP	4th-order Runge–Kutta	10 s step and filter update	CHAMP 1998 [2]
PROBA	4th-order Kutta–Merson		Lafontaine et al. 1999 [15]
BIRD	4th-order Runge–Kutta with Richardson extrapolation and 5th-order Hermite interpolation	35–65 s step size, average filter update 30 s	Gill et al. 2000 [8]

ning of the integration step and propagated, assuming an unperturbed motion. The propagated elements were converted to state vectors and compared with the reference trajectory, yielding position differences of 4 m, 16 m, and 65 m for intervals of 30 s, 60 s, and 120 s. Even worse results are obtained by a simple linear interpolation of the osculating elements at the begin and end of an integration step, yielding position errors of 10 m, 40 m, and 150 m for stepsizes of 30 s, 60 s, and 120 s. Consequently, Keplerian approximation is ruled out for the envisaged application, both in terms of accuracy and the imposed computational burden.

When comparing our finding to existing implementations of on-board navigation systems (cf. *table VI*), we note that considerable savings in processor load might be achieved in most cases by choosing a 5th-order integrator and combining it with a compatible interpolant. Aside from increasing the efficiency of the basic integration method, this approach circumvents the need for stepsize reduction due to dense output requirements. Even when using an interpolant, the measurement and filter update rate should not, however, exceed a value of typically $50-100$ per orbit to achieve an optimum working point for the employed integration method.

Summary and conclusions

A selection of low-order Runge–Kutta type integration methods and associated interpolants has been assessed with regard to use in on-board navigation systems. It is concluded that fifth-order integration methods provide an optimum choice both in terms of efficiency and implementation effort. While lower order methods require a larger amount of derivative computations to meet a given accuracy, the potential of higher order methods cannot be exploited due to the required restart of the trajectory integration after each measurement update. Where applicable (i.e. in the absence of velocity dependent forces), the use of Runge–Kutta–Nystrom methods for the direct integration of the second-order equation of motion provides a 30% efficiency gain over general RK methods for first-order differential equations.

Among the candidate methods considered for interpolation, both fourth- and fifth-order interpolants provide adequate performance as concerns the dense output of position data. It may, however, be observed that only a quintic Hermite polynomial matching both the state vector and its derivative at two consecutive mesh points is able to provide an accurate interpolation of the spacecraft velocity. The latter capability is mandatory if the times of measurements and Kalman filter updates do not coincide with the end-point of the integration interval. Since a double integration step must be used to establish the aforementioned Hermite interpolant, it is particularly attractive for use with Richardson extrapolation and a simple fourth-order Runge–Kutta integration method.

Compared to present implementations of on-board navigation systems, a notable saving in processing time is possible by adding an adequate interpolant to the traditional integration schemes and adjusting the Kalman filter update interval to a natural value of about 1/50th of the orbital period. This allows the use of less powerful processors or, vice versa, the implementation of improved force and measurement models inside the navigation system.

References

[1] Berthias J.P., Broca P., Fourcade J., Jayles C., Pradines D., Laurichesse D., General characteristics of real-time on-board orbit determination, in: Proc. of the 12th International Symposium on Space Flight Dynamics, ESOC, Darmstadt, Germany, SP-403, 1997, pp. 267–274.

[2] CHAMP Design & Interface Document, Iss. 2D, Daimler-Chrysler Aerospace; CH-IT-DID-0001, 1998/10/26, 1998.

[3] Dormand J.R., Prince P.J., A family of embedded Runge–Kutta formulae, J. Comput. Appl. Math. 6 (1) (1980) 19–26.

[4] Dormand J.R., Prince P.J., Runge–Kutta triples, Comput. Math. Appl. 12A (9) (1986) 1007–1017.

[5] Enright W.H., Jackson K.R., Norsett S.P., Thomsen P.G., Interpolants for Runge–Kutta formulas, ACM T. Math. Software 12 (3) (1986) 193–218.

[6] Escobal P.R., Methods of Orbit Determination, John Wiley & Sons, New York, 1965 and Krieger Publishing Company, Malabar, Florida, 1976.

[7] Fehlberg E., Low-order classical Runge–Kutta formulas with step size control and their application to some heat transfer problems, NASA TR R-315, 1969.

[8] Gill E., Montenbruck O., Brieß K., GPS-Based Autonomous Navigation for the BIRD Satellite, 15th International Symposium on Spaceflight Dynamics, June 2000, Biarritz, 2000.

[9] Gramling C., Lorah J., Santoro E., Work K., Chambers R., Preliminary Operational Results of the TDRSS Onboard Navigation System (TONS) for the Terra Mission, 15th International Symposium on Spaceflight Dynamics, June 2000, Biarritz, 2000.

[10] Hairer E., Norsett S.P., Wanner G., Solving Ordinary Differential Equations I, Springer-Verlag, Berlin-Heidelberg-New York, 1987.

[11] Hart R.C., Gramling C.J., Deutschmann J.K., Long A.C., Oza D.H., Steger W.L., Autonomous navigation initiatives at the NASA GSFC flight dynamics division, 96-c-23, in: Proceedings of the 11th IAS (International Astrodynamics Symposium), 1996 Gifu, Japan, 1996, pp. 125–130.

[12] Hart R.C., Hartman K., Long A.C., Lee T., Oza D.H., Global positioning system enhanced orbit determination experiment (GEODE) on the small satellite technology initiative (SSTI) lewis spacecraft, in: Proceedings of ION GPS-96, 17–20, 1996, Kansas City M, 1996, pp. 1303–1312.

[13] Horn M.K., Scaled Runge–Kutta Algorithms for Handling Dense Output, DFVLR-FB 81-13, Deutsche Forschungs- und Versuchsanstalt für Luft- und Raumfahrt, Oberpfaffenhofen, 1981.

[14] Horn M.K., Fourth- and fifth-order scaled Runge–Kutta algorithms for treating dense output, SIAM J. Numer. Anal. 20 (3) (1983) 558–568.

[15] Lafontaine J. de, Buijs J., Vuilleumier P., Brambussche P.v.d., Mellab K., Development of the PROBA attitude control and navigation software, in: Proc. 4th ESA International Conference on Spacecraft Guidance, Navigation and Control Systems, October 1999 Noordwijk NL, ESA SP, 1999, p. 110.

[16] Ledermann W. (Ed.), Handbook of Applicable Mathematics – Vol III: Numerical Methods, John Wiley & Sons, 1984.

[17] Lockheed Martin Corporation, Missiles and Space Division, 20045510C, Software Requirements Specification, Flight Software – Navigation CSCI for EOS-AM Spacecraft (SD-110a), 1997.

[18] Montenbruck O., Numerical integration methods for orbital motion, Celestial Mechanics and Dynamical Astronomy 53 (1992) 59–69.

[19] Montenbruck O., Gill E., Satellite Orbits – Models, Methods, and Applications, Springer-Verlag, Heidelberg, 2000.

[20] Neumayer K.H., König R., An efficient on-board propagator/estimator for LEO missions, in: Proc. AAS/GSFC Internat. Symposium on Space Flight Dynamics, May 1998, Greenbelt, MD, AAS 98-328, 1998, pp. 339–348.

[21] Papageorgiou G., Simos Th., Tstouras Ch., Some new Runge–Kutta methods with interpolation properties and their application to the magnetic-binary problem, Celestial Mechanics 44 (1988) 167–177.

[22] Pradines D., Berthias J.P., Jayles C., Real-Time on-Board Orbit Computation with DORIS, 9th International Symposium on Spaceflight Dynamics, St. Petersburg, Moscow, 1994.

[23] Sauer R., Szabo I. (Eds.), Mathematische Hilfsmittel des Ingenieurs Vol. III, Springer-Verlag, 1968.

[24] Shampine L.F., Some practical Runge–Kutta formulas, Math. Comput. 46 (173) (1986) 135–150.

[25] Shampine L.F., Gordon M.K., Computer Solution of Ordinary Differential Equations, Freeman, San Francisco, 1975.

[26] Tournier T., Berthias J.P., Jayles C., Mercier F., Laurichesse D., Cauquil P., Orbit and Time On-Board Computation: From the Current SPOT4 Solution to GNSS2 Needs, INPE/ABCM 14th International Symposium on Space Flight Dynamics ISSFD XIV Iguassu, Brazil, February 1999.

[27] Unwin M.J., Oldfield M.K., Purivigraipong S., Orbital Demonstration of a New Space GPS Receiver for Orbit and Attitude Determination, Int. Workshop on Aerospace Apps. of GPS, 31 January–2 February 2000 Breckenridge, Colorado, 2000.