Содержание

1 Введение

В теории управления движением космических, авиационных, наземных и других движущихся аппаратов важное место занимает задача определения ориентации аппарата в инерциальной системе координат по его известной (измеренной) абсолютной угловой скорости. Эта задача реализуется в современных системах ориентации космических и аэрокосмических летательных аппаратов, различных наземных и морских автоматических управляемых аппаратов (таких, как роботы, подводные лодки и другие).

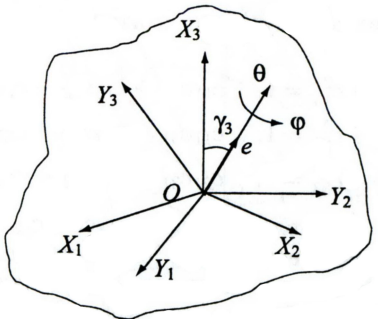
Эти системы ориентации относятся к так называемым системам аналитического типа, поскольку в них определяющую роль играют вычисления, проводимые на бортовом компьютере в реальном масштабе времени. В этих системах ориентации вместо физического моделирования различных координатных систем, осуществляемого с помощью сложных электромеханических устройств, используется математическое моделирование систем координат на бортовом компьютере.

В данной работе рассматривается постановка задачи определения ориентации движущегося объекта в инерциальной системе координат, аналитический аппарат, применяемый для решения этой задачи, а также набор алгоритмов решения этой задачи на ЭВМ. Все рассматриваемые алгоритмы были реализованы и протестированы на ЭВМ с помощью методики математического моделирования. Приводится анализ погрешностей и сравнение методов с точки зрения точности и объёма необходимых вычислений.

2 Кинематические уравнения в параметрах Эйлера

2.1 Параметры Эйлера (Родрига-Гамильтона)

Рассмотрим твёрдое тело, имеющее одну неподвижную точку O. Поместим в эту точку начала двух систем координат: X и Y. Неподвижная система координат X принимается нами в качестве системы отсчёта, а система координат Y жёстко связывается с телом. Считается, что в начальный момент времени системы координат X и Y совпадали.



В соответствии с теоремой Эйлера-Даламбера, всякое перемещение твёрдого тела, имеющего одну неподвижную точку, из начального положения X в конечное Y может быть заменено одним конечным поворотом вокруг оси, проходящей через неподвижную точку.

Обозначим через $\boldsymbol{e}$ единичный вектор, направленный вдоль оси конечного поворота, а через φ — угол конечного поворота тела вокруг этой оси. Введём вектор *конечного поворота*:

$$\boldsymbol{\theta} = 2\boldsymbol{e}\,\mathrm{tg}\,\frac{\varphi}{2}. \tag{2.1}$$

С помощью введённого вектора $\boldsymbol{\theta}$ может быть задано угловое положение твёрдого тела в выбранной системе отсчёта X. Тангенс и коэффициент 2 взяты для удобства: векторные уравнения кинематики твёрдого тела имеют в этом случае более простой вид, чем в любом другом варианте.

Через θ_i обозначим проекции вектора $\boldsymbol{\theta}$ на оси системы координат X.

Тогда *параметрами Эйлера (Родрига-Гамильтона)* называются четыре следующих скалярных величины, определяемых условиями:

$$\begin{cases} \lambda_i = \frac{1}{2}\lambda_0\theta_i, \quad i = 1,2,3, \\ \lambda_0^2 + \lambda_1^2 + \lambda_2^2 + \lambda_3^2 = 1. \end{cases} \tag{2.2}$$

Таким образом, три параметра $\lambda_j, (j = 1,\ldots,3)$ пропорциональны проекциям вектора конечного поворота, а коэффициент пропорциональности λ_0 выбирается из условия нормировки.

Следует отметить, что любому положению тела будет соответствовать ровно два набора параметров λ_j, отличающихся друг от друга только знаками.

Из четырёх параметров Эйлера (Родрига-Гамильтона) можно составить так называемый *кватернион* — гиперкомплексное число, имеющее вид:

$$\boldsymbol{\lambda} = \lambda_0 \mathbf{1} + \lambda_1 \boldsymbol{i}_1 + \lambda_2 \boldsymbol{i}_2 + \lambda_3 \boldsymbol{i}_3. \tag{2.3}$$

Кватернионы были введены ирландским математиком Гамильтоном в 1843 г.

Над кватернионами можно определить операции сложения, умножения на число, а также умножения и деления кватернионов; подробное рассмотрение этих операций выходит за рамки данной работы.

Преимущество параметров Эйлера (Родрига-Гамильтона) перед другими способами задания ориентации твёрдого тела состоит в том, что кинематические уравнения, записанные в этих параметрах, не имеют особых точек (т.е. не вырождаются ни при каких положениях тела) и не содержат тригонометрических функций. Вид кинематических уравнений в параметрах Эйлера рассматривается в следующих пунктах.

2.2 Кватернионное кинематическое уравнение вращательного движения

Уравнение, связывающее параметры Родрига-Гамильтона и их производные по времени с вектором ω угловой скорости тела, называется кватернионным кинематическим уравнением вращательного движения. В кватернионной записи оно имеет вид:

$$2\frac{d\boldsymbol{\lambda}}{dt} = \boldsymbol{\lambda} \circ \boldsymbol{\omega}(t). \tag{2.4}$$

Здесь $\boldsymbol{\lambda} = \lambda_0 + \lambda_1 \boldsymbol{i}_1 + \lambda_2 \boldsymbol{i}_2 + \lambda_3 \boldsymbol{i}_3$ — кватернион, составленный из параметров Эйлера (Родрига-Гамильтона), задающий ориентацию твёрдого тела в пространстве в каждый момент времени. Вектор-функция $\boldsymbol{\omega}(t)$ — вектор угловой скорости тела.

В матричной форме уравнение (2.4) принимает вид:

$$2\begin{pmatrix} d\lambda_0/dt \\ d\lambda_1/dt \\ d\lambda_2/dt \\ d\lambda_3/dt \end{pmatrix} = \begin{pmatrix} 0 & -\omega_1(t) & -\omega_2(t) & -\omega_3(t) \\ \omega_1(t) & 0 & \omega_3(t) & -\omega_2(t) \\ \omega_2(t) & -\omega_3(t) & 0 & \omega_1(t) \\ \omega_3(t) & \omega_2(t) & -\omega_1(t) & 0 \end{pmatrix} \begin{pmatrix} \lambda_0 \\ \lambda_1 \\ \lambda_2 \\ \lambda_3 \end{pmatrix}. \tag{2.5}$$

В скалярном виде эти уравнения соответствуют системе уравнений:

$$\begin{cases} 2d\lambda_0/dt = -\omega_1(t)\lambda_1 - \omega_2(t)\lambda_2 - \omega_3(t)\lambda_3, \\ 2d\lambda_1/dt = \omega_1(t)\lambda_0 + \omega_3(t)\lambda_2 - \omega_2(t)\lambda_3, \\ 2d\lambda_2/dt = \omega_2(t)\lambda_0 - \omega_3(t)\lambda_1 + \omega_1(t)\lambda_3, \\ 2d\lambda_3/dt = \omega_3(t)\lambda_0 + \omega_2(t)\lambda_1 - \omega_1(t)\lambda_2. \end{cases} \tag{2.6}$$

Таким образом, это система линейных дифференциальных нестационарных уравнений четвёртого порядка относительно переменных $\lambda_j(t)$, $(j = 0, \ldots, 3)$. Преимущество такого подхода перед другими способами задания положения тела в том, что у него отстутствуют особые точки для любых положений тела в пространстве, и кроме того, что получаемая в итоге система дифференциальных уравнений линейна.

2.3 Вывод кватернионного кинематического уравнения вращательного движения

Рассмотрим произвольную точку M твёрдого тела. Обозначим через $\boldsymbol{r}$ радиус-вектор точки, проведённый из неподвижной точки O тела, а через x_k и y_k, $(k = 1, \ldots, 3)$ — проекции радиуса-вектора на оси опорной X и связанной Y систем координат (т.е. координаты точки M в этих системах координат).

Введём также отображения $\boldsymbol{r}_X$ и $\boldsymbol{r}_Y$ вектора $\boldsymbol{r}$ на базисы X и Y:

$$\boldsymbol{r}_X = x_1\boldsymbol{i}_1 + x_2\boldsymbol{i}_2 + x_3\boldsymbol{i}_3, \quad \boldsymbol{r}_Y = y_1\boldsymbol{i}_1 + y_2\boldsymbol{i}_2 + y_3\boldsymbol{i}_3. \tag{2.7}$$

Они связаны *кватернионными соотношениями*:

$$\boldsymbol{r}_X = \boldsymbol{\lambda} \circ \boldsymbol{r}_Y \circ \overline{\boldsymbol{\lambda}}, \;\; \boldsymbol{r}_Y = \overline{\boldsymbol{\lambda}} \circ \boldsymbol{r}_X \circ \boldsymbol{\lambda}, \tag{2.8}$$

где $\boldsymbol{\lambda} = \lambda_0 + \lambda_1\boldsymbol{i}_1 + \lambda_2\boldsymbol{i}_2 + \lambda_3\boldsymbol{i}_3$ — кватернион поворота, характеризующий ориентацию твёрдого тела в опорной системе координат; $\lambda_j (j = 0, \ldots, 3)$ — параметры Эйлера (Родрига-Гамильтона); $\overline{\boldsymbol{\lambda}}$ — сопряжённый кватернион (который имеет ту же скалярную часть, а векторная часть взята со знаком минус).

Дифференцируя первое соотношение (2.7) по времени и учитывая, что отображение $\boldsymbol{r}_Y = \text{const}$ (так как вектор r принадлежит телу), получим:

$$\boldsymbol{r}'_X = \boldsymbol{\lambda}' \circ \boldsymbol{r}_Y \circ \overline{\boldsymbol{\lambda}} + \boldsymbol{\lambda} \circ \boldsymbol{r}_Y \circ \overline{\boldsymbol{\lambda}'}. \tag{2.9}$$

Подставляя сюда второе соотношение (2.8), и учитывая, что $\boldsymbol{\lambda} \circ \overline{\boldsymbol{\lambda}} = 1$, имеем:

$$\boldsymbol{r}'_X = \boldsymbol{\lambda}' \circ \overline{\boldsymbol{\lambda}} \circ \boldsymbol{r}_X + \boldsymbol{r}_X \circ \boldsymbol{\lambda} \circ \overline{\boldsymbol{\lambda}'}. \tag{2.10}$$

С другой стороны, по *формуле Эйлера* имеем:

$$\boldsymbol{r}' = \boldsymbol{\omega} \times \boldsymbol{r}, \tag{2.11}$$

где $\boldsymbol{\omega}$ — вектор мгновенной угловой скорости вращения твёрдого тела относительно системы координат X.

Представим эту формулу в кватернионном виде, используя связь операций векторного и кватернионного произведений (кватернионное произведение равно разности векторного и скалярного произведений) и выражение скалярного произведения через отрицание полусуммы двух кватернионных произведений:

$$\boldsymbol{r}'_X = \boldsymbol{\omega} \cdot \boldsymbol{r} + \boldsymbol{\omega}_X \circ \boldsymbol{r}_X = \frac{1}{2}\left(\boldsymbol{\omega}_X \circ \boldsymbol{r}_X - \boldsymbol{r}_X \circ \boldsymbol{\omega}_X\right), \tag{2.12}$$

где $\boldsymbol{\omega}_X = \omega_1^*\boldsymbol{i}_1 + \omega_2^*\boldsymbol{i}_2 + \omega_3^*\boldsymbol{i}_3$ — отображение вектора $\boldsymbol{\omega}$ на опорный базис X.

Приравнивая правые части соотношений (2.10) и (2.12), находим

$$\boldsymbol{\omega}_X \circ \boldsymbol{r}_X - \boldsymbol{r}_X \circ \boldsymbol{\omega}_X = 2\left(\boldsymbol{\lambda}' \circ \overline{\boldsymbol{\lambda}} \circ \boldsymbol{r}_X + \boldsymbol{r}_X \circ \boldsymbol{\lambda} \circ \overline{\boldsymbol{\lambda}'}\right). \tag{2.13}$$

Поскольку кватернионные произведения $\boldsymbol{\lambda} \circ \overline{\boldsymbol{\lambda}'}$ и $\overline{\boldsymbol{\lambda}'} \circ \boldsymbol{\lambda}$ являются векторами, отличающимися знаками (поскольку $\mathrm{scal}(\boldsymbol{\lambda}' \circ \overline{\boldsymbol{\lambda}}) = \mathrm{scal}(\boldsymbol{\lambda} \circ \overline{\boldsymbol{\lambda}'}) = 0$), то получаем:

$$\boldsymbol{\omega}_X = \omega_1^* \boldsymbol{i}_1 + \omega_2^* \boldsymbol{i}_2 + \omega_3^* \boldsymbol{i}_3 = 2\boldsymbol{\lambda}' \circ \overline{\boldsymbol{\lambda}}. \tag{2.14}$$

Для вектора $\boldsymbol{\omega}$ выполняются соотношения, аналогичные (2.8):

$$\boldsymbol{\omega}_X = \lambda \circ \boldsymbol{\omega}_Y \circ \overline{\boldsymbol{\lambda}}, \quad \boldsymbol{\omega}_Y = \overline{\boldsymbol{\lambda}} \circ \boldsymbol{\omega}_X \circ \boldsymbol{\lambda}. \tag{2.15}$$

Из (2.14) и первого соотношения (2.15) находим:

$$\boldsymbol{\omega}_Y = \omega_1 \boldsymbol{i}_1 + \omega_2 \boldsymbol{i}_2 + \omega_3 \boldsymbol{i}_3 = 2\overline{\boldsymbol{\lambda}} \circ \lambda'. \tag{2.16}$$

Из (2.14) и (2.16) получаем две формы *кватернионных кинематических уравнений движения твёрдого тела в параметрах Эйлера*:

$$2\boldsymbol{\lambda}' = \boldsymbol{\omega}_X \circ \boldsymbol{\lambda}, \tag{2.17}$$

$$2\boldsymbol{\lambda}' = \boldsymbol{\lambda} \circ \boldsymbol{\omega}_Y, \tag{2.18}$$

Уравнение (2.18) как раз и является уравнением (2.4), которое мы доказывали (кинематическое уравнение в параметрах Эйлера (Родрига-Гамильтона) в кватернионной форме):

$$2\frac{d\boldsymbol{\lambda}}{dt} = \boldsymbol{\lambda} \circ \boldsymbol{\omega}(t).$$

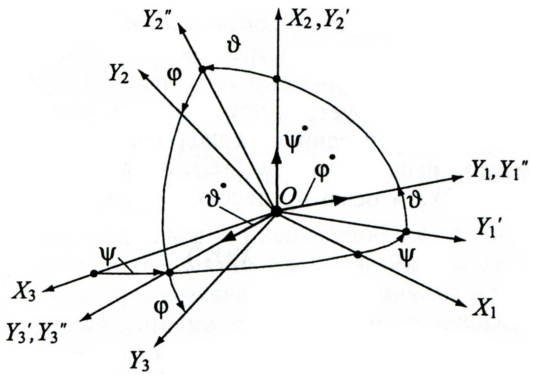
3 Углы Крылова

Углы Крылова являются альтернативным способом задания ориентации твёрдого тела, в отличие от параметров Эйлера (Родрига-Гамильтона). Кроме того, эта система углов более удобна для восприятия человеком, и потому в реальных приложениях текущая ориентация тела, выраженная в кватернионе, конвертируется в углы Крылова и выводится в таком виде. С другой стороны, при проведении математического моделирования удобно использовать углы Крылова для описания простых колебательных движений. Таким образом, следует рассмотреть не только задание углов Крылова, но и преобразование их в кватернион и наоборот.

3.1 Определение

Углы Крылова также называются самолётными. Пусть система координат X по-прежнему является системой отсчёта, относительно которой рассматривается угловое положение твёрдого тела, а с летательным аппаратом жёстко связана система координат Y. Считается, что ось OX_1 направлена горизонтально на географический север, ось OX_2 направляется по вертикали вверх, а ось OX_3 дополняет систему координат X до правой. Ось OY_1 направлена вдоль продольной оси летательного аппарата, ось OY_2 — вдоль поперечной, а OY_3 — вдоль поперечной.

Тогда углами Крылова называется тройка углов ψ, ϑ, φ, определяемых следующим образом. Будем считать, что система координат Y должна быть получена из X тремя поворотами. Первый поворот производится вокруг оси OX_2 на угол ψ (отсчитывается по ходу часовой стрелки), в результате получается некоторая система координат Y'. Второй поворот производится вокруг оси OY_3' на угол ϑ (отсчитывается против часовой стрелки), в результате получается система координат Y''. Наконец, последний поворот производится вокруг оси OY_1'' на угол φ (также против часовой стрелки), в результате получается система координат Y.



Угол ψ называется углом курса, угол ϑ — углом тангажа, φ — углом крена.

3.2 Преобразование в кватернион

Параметры Эйлера (Родрига-Гамильтона) могут быть найдены через самолётные углы по формулам:

$$
\begin{aligned}
\lambda_0 &= \cos\frac{\psi}{2}\cos\frac{\vartheta}{2}\cos\frac{\varphi}{2} - \sin\frac{\psi}{2}\sin\frac{\vartheta}{2}\sin\frac{\varphi}{2},\\
\lambda_1 &= \sin\frac{\psi}{2}\sin\frac{\vartheta}{2}\cos\frac{\varphi}{2} + \cos\frac{\psi}{2}\cos\frac{\vartheta}{2}\sin\frac{\varphi}{2},\\
\lambda_2 &= \sin\frac{\psi}{2}\cos\frac{\vartheta}{2}\cos\frac{\varphi}{2} + \cos\frac{\psi}{2}\sin\frac{\vartheta}{2}\sin\frac{\varphi}{2},\\
\lambda_3 &= \cos\frac{\psi}{2}\sin\frac{\vartheta}{2}\cos\frac{\varphi}{2} - \sin\frac{\psi}{2}\cos\frac{\vartheta}{2}\sin\frac{\varphi}{2}.
\end{aligned}
\tag{3.1}
$$

Эти формулы можно получить, найдя выражение координат вектора конечного поворота через углы Крылова.

3.3 Преобразование из кватерниона

Формулы для обратного преобразования имеют вид:

$$
\begin{aligned}
\psi &= \operatorname{arctg}\frac{\lambda_0\lambda_2 - \lambda_1\lambda_3}{\lambda_0^2 + \lambda_1^2 - 0.5},\\
\vartheta &= \arcsin\big(2\left(\lambda_1\lambda_2 + \lambda_0\lambda_3\right)\big),\\
\varphi &= \operatorname{arctg}\frac{\lambda_0\lambda_1 - \lambda_2\lambda_3}{\lambda_0^2 + \lambda_2^2 - 0.5}.
\end{aligned}
\tag{3.2}
$$

Их можно вывести, сопоставив выражения через кватернионы и через углы Крылова для попарных углов между осями координат X и Y.

4 Постановка задачи определения ориентации движущегося объекта в инерциальной системе координат

Рассмотрим задачу определения ориентации движущегося объекта, рассматриваемого как твёрдое тело, в инерциальной системе координат по известному (измеренному) вектору его абсолютной угловой скорости.

Пусть *известны проекции вектора абсолютной угловой скорости* объекта на связанные с ним координатные оси в текущий момент времени t:

$$\omega_k = \omega_k(t), \quad (k = 1, \ldots, 3) \tag{4.1}$$

или приращения интегралов от них:

$$\varphi_k = \int_{t_{n-1}}^{t_n} \omega_k(t)dt, \quad (k = 1, \ldots, 3). \tag{4.2}$$

Информация вида (4.1) называется *мгновенной информацией* об абсолютном угловом (вращательном) движении объекта, а информация вида (4.2) — *интегральной*. Как правило, мгновенная информация выдаётся гироскопическими измерителями абсолютной угловой скорости объекта, имеющими аналоговый (непрерывный) выход, а интегральная информация — гироскопами, имеющими цифровой (дискретный) выход.

Задача заключается в том, чтобы *определить ориентацию объекта* в инерциальной системе координат в текущий момент времени t:

$$\lambda_j = \lambda_j(t), \quad (j = 0, \ldots, 3). \tag{4.3}$$

При этом заданы *начальные условия* — ориентация объекта в начальный момент времени t_0:

$$\lambda_j(t_0) = \lambda_j^0, \quad (j = 0, \ldots, 3). \tag{4.4}$$

С математической точки зрения задача заключается в *непрерывном интегрировании* на бортовом компьютере в реальном времени системы дифференциальных уравнений (2.4) (или эквивалентных ей (2.5) и (2.6)) при заданных начальных условиях интегрирования (4.4) и известной информации вида (4.1) или (4.2).

5 Алгоритмы определения ориентации объекта

В основе решения задачи определения ориентации движущегося объекта лежит численное интегрирование системы дифференциальных уравнений (2.4):

$$2\frac{d\boldsymbol{\lambda}}{dt} = \boldsymbol{\lambda} \circ \boldsymbol{\omega}(t).$$

Для её интегрирования могут быть использованы стандартные методы численного интегрирования дифференциальных уравнений, такие как метод Рунге-Кутта, метод Эйлера, метод трапеций, — использующие мгновенную информацию (4.1), а также численные методы, специально разработанные для интегрирования системы (2.4), — использующие либо мгновенную информацию (4.1), либо интегральную информацию (4.2).

Критериями при выборе метода служат требуемая точность вычисления параметров ориентации и объём необходимых вычислений.

5.1 Алгоритм, реализующий метод Рунге-Кутта четвёртого порядка

Система (2.4) представляет собой систему дифференциальных уравнений, которую можно записать в общем виде:

$$\frac{d\boldsymbol{\lambda}}{dt} = \boldsymbol{f}(t, \boldsymbol{\lambda}). \tag{5.1}$$

Тогда алгоритм решения этой системы, реализующий классический метод Рунге-Кутта четвёртого порядка, запишется в виде:

$$\lambda_n = \lambda_{n-1} + \frac{1}{6}(\boldsymbol{k}_1 + 2\boldsymbol{k}_2 + 2\boldsymbol{k}_3 + \boldsymbol{k}_4), \tag{5.2}$$

где

$$\begin{gathered}
\boldsymbol{k}_1 = h\boldsymbol{f}(t_{n-1}, \boldsymbol{\lambda}_{n-1}), \\
\boldsymbol{k}_2 = h\boldsymbol{f}(t_{n-1} + \frac{1}{2}h, \boldsymbol{\lambda}_{n-1} + \frac{1}{2}\boldsymbol{k}_1), \\
\boldsymbol{k}_3 = h\boldsymbol{f}(t_{n-1} + \frac{1}{2}h, \boldsymbol{\lambda}_{n-1} + \frac{1}{2}\boldsymbol{k}_2), \\
\boldsymbol{k}_1 = h\boldsymbol{f}(t_{n-1} + h, \boldsymbol{k}_3).
\end{gathered} \tag{5.3}$$

Здесь $h = t_n - t_{n-1}$ — шаг интегрирования.

Этот алгоритм имеет четвёртый порядок точности, что означает, что накапливающая погрешность алгоритма есть величина $O(h^4)$.

Следует отметить, что фактически этот алгоритм является *двухшаговым* алгоритмом: дискретность выдачи информации вдвое меньше дискретности входных данных. Иными словами, для вычисления ориентации тела в момент времени $t = t_n$ данный алгоритм использует данные об ориентации тела в момент времени $t = t_{n-1}$ и данные об абсолютной угловой скорости тела в моменты времени $t = t_{n-1}$, $t = \frac{t_{n-1}+t_n}{2}$ и $t = t_n$.

5.2 Общий вид специальных алгоритмов решения задачи

Приводимые далее алгоритмы, используемые в современных системах управления ориентацией движущихся объектов, строятся по общей рекуррентной схеме, имеющей в матричной записи вид:

$$\begin{pmatrix}\lambda_0(t_n)\\ \lambda_1(t_n)\\ \lambda_2(t_n)\\ \lambda_3(t_n)\end{pmatrix} = \begin{pmatrix}\lambda_0^* & -\lambda_1^* & -\lambda_2^* & -\lambda_3^*\\ \lambda_1^* & \lambda_0^* & \lambda_3^* & -\lambda_2^*\\ \lambda_2^* & -\lambda_3^* & \lambda_0^* & \lambda_1^*\\ \lambda_3^* & \lambda_2^* & -\lambda_1^* & \lambda_0^*\end{pmatrix} \begin{pmatrix}\lambda_0(t_{n-1})\\ \lambda_1(t_{n-1})\\ \lambda_2(t_{n-1})\\ \lambda_3(t_{n-1})\end{pmatrix} \tag{5.4}$$

или, в векторной записи, вид:

$$\lambda_0(t_n) = \lambda_0(t_{n-1})\lambda_0^* - \boldsymbol{\lambda}_v(t_{n-1}) \cdot \boldsymbol{\lambda}_v^*, \tag{5.5}$$

Здесь через λ_0 и $\boldsymbol{\lambda}_v$ обозначены, соответственно, скалярная и векторная часть кватерниона $\boldsymbol{\lambda}$ (т.е. компонента λ_0, и вектор, составленный из компонент $\lambda_1,\ \lambda_2,\ \lambda_3$).

Величины $\lambda_j^*, (j = 0, \ldots, 3)$ вычисляются на каждом шаге интегрирования по соотношениям, задаваемым выбранным алгоритмом.

5.3 Метод средней скорости

Этот алгоритм второго порядка точности, который может использовать как мгновенную, так и интегральную информацию.

Алгоритм, использующий мгновенную информацию о движении объекта:

$$\begin{gathered}\omega(t) = \sqrt{\omega_1^2(t) + \omega_2^2(t) + \omega_3^2(t)},\\ \varphi = \int_{t_{n-1}}^{t_n} \omega(t)dt,\\ \omega_{in} = \omega_i(t_n), \quad \omega_n = \omega(t_n),\\ \lambda_0^* = \cos(\frac{\varphi}{2}),\\ \lambda_j^* = \frac{\omega_{in}}{\omega_n}\sin(\frac{\varphi}{2}).\end{gathered} \tag{5.6}$$

Алгоритм, использующий интегральную информацию:

$$\begin{gathered}\varphi = \sqrt{\varphi_1^2 + \varphi_2^2 + \varphi_3^2},\\ \lambda_0^* = \cos(\frac{\varphi}{2}),\\ \lambda_j^* = \frac{\varphi_i}{\varphi_n}\sin(\frac{\varphi}{2}).\end{gathered} \tag{5.7}$$

Алгоритм средней скорости имеет второй порядок точности, т.е. погрешность решения, даваемого этим алгоритмом, есть величина $O(h^2)$.

5.4 Одношаговый алгоритм третьего порядка точности

Этот алгоритм может использовать только интегральную информацию о движении объекта.

Скалярная запись алгоритма:

$$\begin{gathered}
\varphi_i = \int_{t_{n-1}}^{t_n} \omega_i(t)dt, \quad \varphi_i^* = \int_{t_{n-2}}^{t_{n-1}} \omega_i(t)dt, \\
\varphi = \sqrt{\varphi_1^2 + \varphi_2^2 + \varphi_3^2}, \\
\alpha = \frac{1}{2} - \frac{1}{48}\varphi^2, \\
\beta = \frac{1}{24}, \\
\varphi_{12} = \varphi_2\varphi_1^* - \varphi_1\varphi_2^*, \\
\varphi_{23} = \varphi_3\varphi_2^* - \varphi_2\varphi_3^*, \\
\varphi_{31} = \varphi_1\varphi_3^* - \varphi_3\varphi_1^*, \\
\lambda_0^* = 1 - \frac{1}{8}\varphi^2, \\
\lambda_1^* = \alpha\varphi_1 + \beta\varphi_{23}, \\
\lambda_2^* = \alpha\varphi_2 + \beta\varphi_{31}, \\
\lambda_3^* = \alpha\varphi_3 + \beta\varphi_{12}.
\end{gathered} \tag{5.8}$$

Как видно, этот алгоритм для своей работы требует известные значения интегральной информации не только на текущем временном отрезке $[t_{n-1}; t_n]$, но и на предыдущем $[t_{n-2}; t_{n-1}]$. Отсюда следует, что для корректной работы этот алгоритма необходимо знать ориентацию объекта не только при начальном времени $t = t_0$ (которая задана в начальных данных), но и его ориентацию в следующий момент времени $t = t_1$. Следовательно, для этого алгоритма нужен так называемый *разгон*: первый временной шаг необходимо осуществить не этим алгоритмом, а каким-либо другим алгоритмом (например, методом средней скорости).

Погрешность, даваемая этим алгоритмом, имеет третий порядок роста относительного временного шага, т.е. есть величина $O(h^3)$.

5.5 Двухшаговый алгоритм третьего порядка точности

Этот алгоритм использует интегральную информацию о движении объекта.

Скалярная запись алгоритма:

$$\varphi_i^0 = \int_{t_{n-1}}^{t_{n-1}+h/2} \omega_i(t)dt, \quad \varphi_i' = \int_{t_{n-1}+h/2}^{t_n} \omega_i(t)dt,$$

$$\varphi_i = \int_{t_{n-1}}^{t_n} \omega_i(t)dt = \varphi_i^0 + \varphi_i',$$

$$\varphi = \sqrt{\varphi_1^2 + \varphi_2^2 + \varphi_3^2},$$
$$\alpha_1 = \varphi_3' \varphi_2^0 - \varphi_2' \varphi_3^0,$$
$$\alpha_2 = \varphi_1' \varphi_3^0 - \varphi_3' \varphi_1^0,$$
$$\alpha_3 = \varphi_2' \varphi_1^0 - \varphi_1' \varphi_2^0, \tag{5.9}$$
$$\lambda_0^* = 1 - \frac{1}{8}\varphi^2,$$
$$\lambda_i^* = \left(\frac{1}{2} - \frac{1}{48}\varphi^2\right)\varphi_i + \frac{1}{3}\alpha_i, \quad (i = 1, \ldots, 3)$$

Этот алгоритм также имеет третий порядок точности, т.е. погрешность есть величина $O(h^3)$.

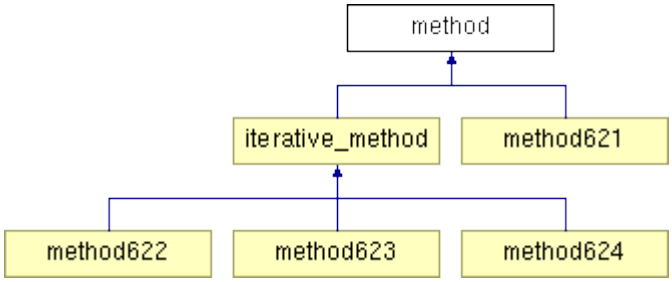
6 Описание программы, реализующей алгоритмы определения ориентации

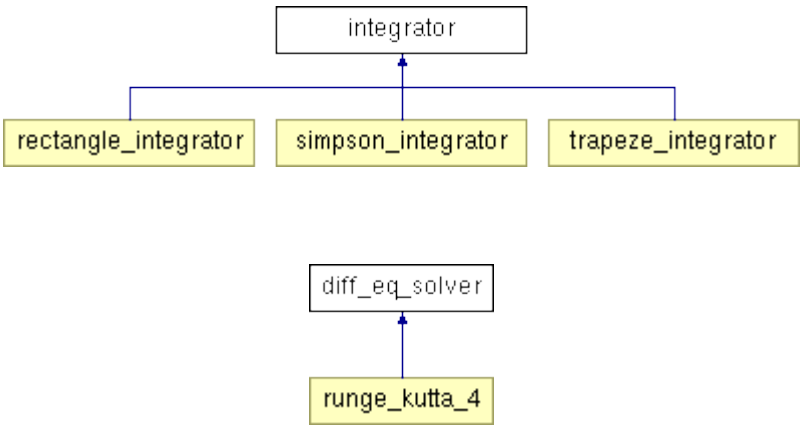
Для реализации был выбран язык программирования C++, который является современным языком программирования, сочетающим в себе как высокую *производительность*, так и поддержку *объектно-ориентированного программирования*. Оба аспекта были важны в данном исследовании: первый — поскольку описываемые здесь алгоритмы должны работать с огромными массивами данных, особенно на стадии тестирования, а второй — поскольку хотелось выстроить иерархию алгоритмов и реализовать алгоритмы так, чтобы они легко состыковывались друг с другом.

Были созданы следующие классы:

- `quaternion` — кватернион;
- `plane_angles` — углы Крылова;
- `method` — интерфейс (каркас) для любого алгоритма решения задачи определения ориентации;
- `iterative_method` — интерфейс для любого итеративного алгоритма решения задачи определения ориентации;
- `method621, method622, method623, method624` — алгоритмы решения задачи определения ориентации (алгоритм сведения к системе дифференциальных уравнений, метод средней скорости, одношаговый и двухшаговый алгоритмы соответственно);
- `integrator` — интерфейс для любого алгоритма численного вычисления интегралов;
- `diff_eq_solver` — интерфейс для любого алгоритма решения системы дифференциальных уравнений;
- `runge_kutta_4` — алгоритм Рунге-Кутта четвёртого порядка решения произвольной системы дифференциальных уравнений;
- `rectangle_integrator, trapeze_integrator, simpson_integrator` — алгоритмы численного вычисления интегралов (алгоритм прямоугольников, трапеций и Симпсона соответственно).

Иерархия наследования классов:





Все алгоритмы реализованы с использованием основных принципов ООП: абстракции данных, инкапсуляции и полиморфизма. Благодаря этому каждый алгоритм представляет собой отдельный блок — “чёрный ящик“, который может состыковываться с другими алгоритмами или обобщёнными функциями, работающими вне зависимости от метода, который ей передали. Например, обобщённая функция `run_method` принимает в качестве аргумента указатель на класс `method`, два указателя на вектор-функции, генерирующую углы Крылова и их производные, а также начальное и конечное времена и шаг по времени. Эта функция будет работать с любыми возможными алгоритмами и входными данными, поскольку она использует только интерфейсы, общие для всех алгоритмов.

Также следует отметить, что при реализации активно использовалась известная библиотека *Boost* для языка C++, которая содержит огромное количество вспомогательных функций и расширений языка. В данном случае преимущественно использовались средства линейной алгебры, а также средства по работе с указателями на функции.

Кроме того, вся программа полностью документирована (все функции и классы имеют комментарии, подробно описывающие их назначение, входные и выходные данные и условия работы), для этого использовалось известное средство документирования *Doxygen*.

Текст программы приведён в приложении.

7 Тестирование алгоритмов

7.1 Методика тестирования

Для тестирования различных алгоритмов применялась методика *математического моделирования* на компьютере.

Эта методика заключается в следующем: рассмотрим какой-либо вид движения тела, поддающийся простому аналитическому описанию. В данной работе рассматриваются два вида колебаний: незатухающие гармонические (7.1) и затухающие гармонические (7.2):

$$\begin{cases} \psi = \psi_+ \sin(\omega_\psi t + \varphi_1^*), \\ \vartheta = \vartheta_+ \sin(\omega_\vartheta t + \varphi_2^*), \\ \gamma = \gamma_+ \sin(\omega_\gamma t + \varphi_3^*); \end{cases} \tag{7.1}$$

$$\begin{cases} \psi = \psi_m e^{-\sigma_1 t} \sin(\omega_\psi t), \\ \vartheta = \vartheta_m e^{-\sigma_2 t} \sin(\omega_\vartheta t), \\ \gamma = \gamma_m e^{-\sigma_3 t} \sin(\omega_\gamma t), \end{cases} \tag{7.2}$$

где

- $\psi_+, \vartheta_+, \gamma_+$ — амплитуды гармонических колебаний;
- $\psi_m, \vartheta_m, \gamma_m$ — амплитуды затухающих колебаний;
- $\omega_\psi, \omega_\vartheta, \omega_\gamma$ — амплитуды затухающих колебаний;
- $\varphi_i^* (i = 1, \ldots, 3)$ — начальные фазы колебаний;
- $\sigma_i (i = 1, \ldots, 3)$ — коэффициенты затухания.

Входными данными для всех алгоритмов определения ориентации объекта являются проекции абсолютной угловой скорости $\omega_i, (i = 1, \ldots, 3)$ или их интегралы $\varphi_i = \int_{t_{n-1}}^{t_n} \omega_i(t)dt, (i = 1, \ldots, 3)$, которые можно сформировать по известным законам движения, пользуясь соотношениями:

$$\begin{aligned} \omega_1 &= \dot{\gamma} + \dot{\psi} \sin\vartheta, \\ \omega_2 &= \dot{\vartheta} \sin\gamma + \dot{\psi} \cos\vartheta \cos\gamma, \\ \omega_3 &= \dot{\vartheta} \cos\gamma - \dot{\psi} \cos\vartheta \sin\gamma. \end{aligned} \tag{7.3}$$

Формулы для вычисления производных:

- для гармонических колебаний:

$$\begin{aligned} \dot{\psi} &= \psi_+ \omega_\psi \cos(\omega_\psi t + \varphi_1^*), \\ \dot{\vartheta} &= \vartheta_+ \omega_\vartheta \cos(\omega_\vartheta t + \varphi_2^*), \\ \dot{\gamma} &= \gamma_+ \omega_\gamma \cos(\omega_\gamma t + \varphi_3^*); \end{aligned} \tag{7.4}$$

- для затухающих колебаний:

$$\begin{aligned}\dot{\psi} &= \psi_m e^{-\sigma_1 t}(\omega_\psi \cos(\omega_\psi t) - \sigma_1 \sin(\omega_\psi t)),\\ \dot{\vartheta} &= \vartheta_m e^{-\sigma_2 t}(\omega_\vartheta \cos(\omega_\vartheta t) - \sigma_2 \sin(\omega_\vartheta t)),\\ \dot{\gamma} &= \gamma_m e^{-\sigma_2 t}(\omega_\gamma \cos(\omega_\gamma t) - \sigma_3 \sin(\omega_\gamma t)).\end{aligned} \tag{7.5}$$

Стоит отметить, что вычисление интегральных данных сопряжено с вычислениями интегралов, которые приходится производить численно. В данной работе рассматриваются метод прямоугольников, метод трапеций и метод Симпсона и то, насколько выбор того или иного алгоритма и шага интегрирования влияет на результат.

Также необходимо задать *начальные условия интегрирования*, для чего необходимо найти значение кватерниона по известным значениям углов Крылова.

После вычисления начальных условий и входных данных можно производить *запуск* того или иного алгоритма. Получаемое в результате решение в виде кватернионов следует перевести в углы Крылова, и сравнить с теоретически ожтдаемыми значениями этих углов (полученными по известным уравнениям движения). Получаемая в результате погрешность (вычисленная как максимум из модулей разностей между соответствующими углами Крылова) и является основной оценкой алгоритма.

7.2 Тестирование на коротком временном интервале с простыми гармоническими колебаниями малой амплитуды и с большой частотой

Рассматривается интервал времени от $t = 0$ до $t = 1$ с. Колебания рассматриваются гармонические, с единичными (1 градус) амплитудами и частотами, и с нулевыми начальными фазами:

$$\begin{gathered}\psi_+ = \vartheta_+ = \gamma_+ = 1 \text{ град},\\ \omega_\psi = \omega_\vartheta = \omega_\gamma = \pi \text{ рад/с},\\ \varphi_1^* = \varphi_2^* = \varphi_3^* = 0 \text{ рад}.\end{gathered}$$

Рассмотрим зависимость погрешностей от величины шага h по времени ($h = t_{i+1} - t_i, i = 0, \ldots$):

Метод	Рунге-Кутта	Средней скорости мгнов.	Средней скорости интегр.	Одношаговый	Двухшаговый
Порядок	4	2	2	3	3
h (с)	Погрешность (град)				
0.1	0.0000000322516	0.0008415913080	0.0000003298575	0.0000000505767	0.0000000000660
0.01	0.0000000000032	0.0000841919897	0.0000000033016	0.0000000000508	0.0000000000000
0.001	0.0000000000000	0.0000084192944	0.0000000000330	0.0000000000001	0.0000000000000
0.0001	0.0000000000000	0.0000008419304	0.0000000000008	0.0000000000004	0.0000000000004
0.00001	0.0000000000000	0.0000000841932	0.0000000000002	0.0000000000002	0.0000000000002

Из этой таблицы можно сделать вывод, что все методы сходятся, т.е. уменьшение шага интегрирования приводит к уменьшению погрешности, причём ско-

рость этого уменьшения сообразуется с порядками погрешностей алгоритмов, полученными теоретически. Единственный метод, выбивающийся из общего ряда — метод средней скорости по мгновенным данным; судя по результатам тестирования, этот метод работает как метод первого порядка точности.

Наилучшие результаты показывали здесь двухшаговые методы: метод Рунге-Кутта и двухшаговый метод по интегральной информации.

Все эти закономерности будут сохраняться и при последующих тестированиях.

7.3 Тестирование на коротком временном интервале с простыми гармоническими колебаниями большой амплитуды с большой частотой

Рассматривается интервал времени от $t = 0$ до $t = 1$ с. Колебания рассматриваются гармонические, со следующими параметрами:

$$\psi_+ = \vartheta_+ = \gamma_+ = 1 \text{ рад},$$
$$\omega_\psi = \omega_\vartheta = \omega_\gamma = \pi \text{ рад/с},$$
$$\varphi_1^* = \varphi_2^* = \varphi_3^* = 0 \text{ рад}.$$

Тогда таблица погрешностей принимает вид:

Метод	Рунге-Кутта	Средней скорости мгнов.	Средней скорости интегр.	Одношаговый	Двухшаговый
Порядок	4	2	2	3	3
h (с)	Погрешность (град)				
0.1	0.0000969099508	2.6989112083922	0.1088511050989	0.0117821682282	0.0027049477996
0.01	0.0000000091458	0.2697553063813	0.0010890287444	0.0000116138557	0.0000025923826
0.001	0.0000000000010	0.0269672730583	0.0000108903390	0.0000000115942	0.0000000025802
0.0001	0.0000000000003	0.0026966375356	0.0000001089017	0.0000000000116	0.0000000000042
0.00001	0.0000000000002	0.0002696628477	0.0000000010887	0.0000000000044	0.0000000000044

Здесь видно, что алгоритм Рунге-Кутта четвёртого порядка даёт самый точный результат из всех рассматриваемых алгоритмов (более подробно она обсуждается в заключении).

Кроме того, видно, что метод средней скорости по мгновенным данным даёт весьма неточные результаты на данном тесте, гораздо хуже остальных алгоритмов.

7.4 Тестирование на коротком временном интервале с простыми гармоническими колебаниями большой амплитуды с малой частотой

Интервал времени по-прежнему от $t = 0$ до $t = 1$ с, а параметры колебаний имеют вид:

$$\psi_+ = \vartheta_+ = \gamma_+ = 1 \text{ рад},$$
$$\omega_\psi = \omega_\vartheta = \omega_\gamma = \pi \text{ град/с},$$
$$\varphi_1^* = \varphi_2^* = \varphi_3^* = 0 \text{ рад}.$$

Тогда вычисленные погрешности равны:

Метод	Рунге-Кутта	Средней скорости мгнов.	Средней скорости интегр.	Одношаговый	Двухшаговый
Порядок	4	2	2	3	3
h (с)	Погрешность (град)				
0.1	0.0000000000002	0.0011564741964	0.0000005142758	0.0000000518134	0.0000000000047
0.01	0.0000000000000	0.0001156583839	0.0000000051428	0.0000000000519	0.0000000000000
0.001	0.0000000000000	0.0000115659476	0.0000000000514	0.0000000000001	0.0000000000000
0.0001	0.0000000000000	0.0000011565959	0.0000000000006	0.0000000000002	0.0000000000002
0.00001	0.0000000000000	0.0000001156597	0.0000000000012	0.0000000000012	0.0000000000012

7.5 Тестирование на длинном временном интервале с заданными гармоническими колебаниями

Теперь будем рассматривать временной интервал от $t = 0$ до $t = 1$ ч. Параметры гармонических колебаний возьмём для примера следующие:

$$\psi_+ = 1 \text{ рад}, \quad \vartheta_+ = 2 \text{ рад}, \quad \gamma_+ = 3 \text{ рад},$$
$$\omega_\psi = \pi \text{ рад/с}, \quad \omega_\vartheta = \frac{\pi}{2} \text{ рад/с}, \quad \omega_\gamma = 2\pi \text{ рад/с},$$
$$\varphi_1^* = \varphi_2^* = \varphi_3^* = 0 \text{ рад}.$$

Тогда в зависимости от выбираемого алгоритма и временного шага получаем следующие результаты:

Метод	Рунге-Кутта	Средней скорости мгнов.	Средней скорости интегр.	Одношаговый	Двухшаговый
Порядок	4	2	2	3	3
h (с)	Погрешность (град)				
0.1	0.0037396267679	79.996653756418	0.0833313823799	0.0072496716827	0.0012670442269
0.01	0.0000000442476	2.9664310584381	0.0008539121193	0.0000010091075	0.0000003703403
0.001	0.0000000000834	0.0749251759227	0.0000085412111	0.0000000006442	0.0000000002751

7.6 Тестирование на длинном временном интервале с заданными гармоническими колебаниями

Входные данные аналогичны предыдущим, за исключением изменившихся частот:

$$\omega_\psi = \frac{\pi}{2} \text{ рад/с}, \quad \omega_\vartheta = \pi \text{ рад/с}, \quad \omega_\gamma = \pi \text{ рад/с},$$

Метод	Рунге-Кутта	Средней скорости мгнов.	Средней скорости интегр.	Одношаговый	Двухшаговый
Порядок	4	2	2	3	3
h (с)	Погрешность (град)				
0.1	0.0000853770520	72.863201363959	0.0105816361669	0.0021283435153	0.0000653513516
0.01	0.0000000017465	7.6552805854890	0.0001064811568	0.0000022843632	0.0000000363513
0.001	0.0000000000088	0.7663169435892	0.0000010648697	0.0000000022874	0.0000000000492

7.7 Тестирование на длинном временном интервале с заданными затухающими колебаниями

Входные данные имеют следующий вид:

$$\psi_+ = 15 \text{ град}, \quad \vartheta_+ = 5 \text{ град}, \quad \gamma_+ = 20 \text{ град},$$
$$\omega_\psi = 2\pi \text{ рад/с}, \quad \omega_\vartheta = \frac{\pi}{2} \text{ рад/с}, \quad \omega_\gamma = 2\pi \text{ рад/с},$$
$$\varphi_1^* = \varphi_2^* = \varphi_3^* = 0 \text{ рад},$$
$$\sigma_1 = \sigma_2 = \sigma_3 = 2.$$

Метод	Рунге-Кутта	Средней скорости мгнов.	Средней скорости интегр.	Одношаговый	Двухшаговый
Порядок	4	2	2	3	3
h (с)	Погрешность (град)				
0.1	0.0024837705733	2.9592524272408	0.0152945713849	0.0152945713849	0.0006680494361
0.01	0.0000002417724	0.1324571121292	0.0001740938499	0.0000354271349	0.0000002167174
0.001	0.0000000000242	0.0144337592767	0.0000017432564	0.0000000376775	0.0000000001673

7.8 Тестирование на длинном временном интервале с заданными затухающими колебаниями

Входные данные здесь имеют немного другой вид:

$$\psi_+ = 15 \text{ град}, \quad \vartheta_+ = 5 \text{ град}, \quad \gamma_+ = 20 \text{ град},$$
$$\omega_\psi = 2\pi \text{ рад/с}, \quad \omega_\vartheta = \pi \text{ рад/с}, \quad \omega_\gamma = 2\pi \text{ рад/с},$$
$$\varphi_1^* = \frac{\pi}{2} \text{ рад}, \quad \varphi_2^* = \frac{\pi}{3} \text{ рад}, \quad \varphi_3^* = 0 \text{ рад},$$
$$\sigma_1 = \sigma_2 = \sigma_3 = 1.$$

Метод	Рунге-Кутта	Средней скорости мгнов.	Средней скорости интегр.	Одношаговый	Двухшаговый
Порядок	4	2	2	3	3
h (с)	Погрешность (град)				
0.1	0.0023270251544	8.1343194707060	0.5351527651653	0.0976824343780	0.0022865670063
0.01	0.0000002396778	0.7939332786959	0.0054597765142	0.0001082829397	0.0000003063852
0.001	0.0000000000240	0.0792542868911	0.0000546086912	0.0000001091683	0.0000000001869

8 Анализ результатов тестирования

8.1 Анализ с точки зрения величины погрешности

Тестирование показало *корректность* всех описанных в работе алгоритмов: все они давали небольшую погрешность на входных колебаниях различного характера. Более того, чётко видно, что для всех алгоритмов, кроме алгоритма средней скорости для мгновенных данных, величина погрешности убывает с тем же порядком скорости, что и порядок, полученный теоретически. Скажем, если алгоритм имеет четвёртый порядок точности, то уменьшение шага h в 10 раз приводит к уменьшению погрешности примерно в 10^4 раз, что отчётливо прослеживается во всех результатах тестирования.

Единственный алгоритм, для которого экспериментальный порядок погрешности не совпал с теоретическим — это *алгоритм средней скорости по мгновенным данным*. Хотя теоретически его порядок погрешности равен двум, на практике он работает как метод первого порядка, что является неприемлемым во многих приложениях. Объяснить это расхождение теоретической и практической оценок не удалось.

Другой *алгоритм средней скорости — по интегральным данным* — также оказался среди аутсайдеров, хотя и подтвердил на практике свой второй порядок точности. Как показало тестирование, при большом временном интервале (1 ч) желательно использовать шаг $h = 0.01$ с или меньше, чтобы погрешность имела приемлемую величину (тысячные доли градуса). Однако уже при таком выборе шага другие алгоритмы оказываются лучше (по величине погрешности) как минимум в десятки-сотни раз; с уменьшением шага этот разрыв только возрастает.

Безусловными лидерами по величине погрешности оказались *двухшаговые алгоритмы*: метод решения системы дифференциальных уравнений Рунге-Кутта, и двухшаговый итерационный метод. Оба метода имеют четвёртый порядок точности, и заметно обходят все остальные алгоритмы практически при любых входных данных и шагах интегрирования. Разница между этими двумя методами и лучшим из остальных (одношаговым итеративным алгоритмом) составляет 10-100 раз уже при шаге $h = 0.1$ с; с уменьшением шага разница между ними только возрастает.

Шага $h = 0.1$ с достаточно для двухшаговых методов, чтобы получить погрешность не более 0.005 град. При уменьшении шага до $h = 0.01$ с погрешность ограничивается на проведённых тестах величиной 10^{-5} град. При шаге $h = 0.001$ с погрешность, даваемая этими алгоритмами, уже не превосходит 10^{-10}, что приближается к пределу представимости в разрядной сетке компьютера, и намного превосходит все мыслимые практические потребности.

Следует отметить, что хотя метод Рунге-Кутта имеет на один порядок точности выше, чем двухшаговый итеративный метод, на практике их результаты оказываются близкими друг к другу, хотя во многих случаях метод Рунге-Кутта всё же оказывается немного (иногда до 100 раз) точнее.

С другой стороны, и *одношаговый итеративный метод* даёт достаточно невысокую погрешность. Скажем, при шаге $h = 0.1$ с она не превосходит

0.1 градуса, чего уже может быть достаточно для практического применения. Учитывая, что метод имеет третий порядок точности, легко подобрать нужный шаг, обеспечивающий требуемую точность. Впрочем, если требуется очень высокая точность определения положения, то одношаговый метод будет требовать в сотни и больше раз больше шагов, чем двухшаговые методы. Таким образом, его применение целесообразно только в тех случаях, когда требуемая точность не слишком велика.

8.2 Анализ с точки зрения объёма требуемых вычислений

С точки зрения объёма вычислений, любой двухшаговый метод проигрывает одношаговым ровно в два раза. Эта разница может оказаться критичной в некоторых реальных приложениях, поэтому одношаговый метод может оказаться предпочтительнее, если требуемая точность позволяет применить его.

В остальном же, все методы выполняют примерно одинаковое количество операций за один шаг, поэтому выделить здесь какой-либо отдельный алгоритм невозможно.

8.3 Анализ с точки зрения входных данных

С этой точки зрения все алгоритмы разделяются на две группы: использующие мгновенную информацию, и использующие интегральную информацию. Следует отметить, что на практике в большинстве случаев датчики получают как раз *интегральную* информацию. Алгоритмы же, использующие мгновенную информацию, находят сравнительно редкое применение.

Вследствие этого замечания получаем, что самый точный из рассматриваемых алгоритмов — метод Рунге-Кутта — имеет этот серьёзный недостаток, — использование мгновенной информации, а не интегральной. С этой точки зрения более предпочтительными оказываются *одношаговый и двухшаговый итеративные* методы.

Следует также отметить в этом пункте и другой момент. Двухшаговые алгоритмы имеют *вдвое меньшую дискретность* выходной информации, чем одношаговые. Так, метод Рунге-Кутта для определения положения тела в момент времени $t = t_n$ использует не только информацию в текущий момент $t = t_n$ и предыдущий $t = t_{n-1}$, но и информацию об угловой скорости тела в средний момент времени $t = (t_n + t_{n-1})/2$. Аналогично, двухшаговому итеративному методу недостаточно интегральной информации об угловой скорости тела в отрезок времени $[t_{n-1}; t_n]$, ему требуется знать две величины: интегральную информацию в первую и во вторую половины этого временного отрезка.

Это приводит к тому, что фактически для работы двухшагового алгоритма датчики летательного аппарата надо опрашивать с временным шагом $h/2$, т.е. вдвое чаще, и вычислений производить вдвое больше, чем для одношагового алгоритма. Это является *минусом двухшаговых алгоритмов.*

8.4 Выводы

Объединяя всё вышесказанное, можно сделать следующие выводы:

- Оба *метода средней скорости* нецелесообразно применять на практике.
- *Метод Рунге-Кутта* является самым точным, однако имеет узкое приложение из-за применения мгновенных данных, и вдвое медленнее одношаговых алгоритмов.
- *Двухшаговый итеративный метод* также является весьма точным, и при этом использует интегральную информацию; однако он тоже вдвое медленнее одношаговых алгоритмов.
- *Одношаговый итеративный метод* имеет меньшую точность, чем два предыдущих, и потому целесообразен только при небольшой требуемой точности; однако он имеет преимущество над ними в производительности и объёме требуемых входных данных.

9 Заключение

В данной работе был рассмотрен вопрос определения ориентации твёрдого тела и современный подход к его решению. Была рассмотрена вся необходимая теоретическая база, и поставлена задача определения ориентации. Также приведены несколько алгоритмов решения этой задачи, которые затем были реализованы для тестирования и сравнения даваемых погрешностей. Произведён анализ результатов тестирования: с точки зрения типа входных данных, величины погрешности, объёма требуемых вычислений.

10 Список использованной литературы

1. Челноков Ю.Н. Инерциальная ориентация и навигация движущихся объектов / Ю.Н. Челноков. — Саратов: Изд-во Сарат. ун-та, 2002.
2. Челноков Ю.Н. Кватернионные и бикватернионные модели и методы механики твердого тела и их приложения / Ю.Н. Челноков. — М.: Физматлит, 2006.
3. Бухгольц Н.Н. Основной курс теоретической механики. Ч. 1 / Н.Н. Бухгольц. — М.: Наука, 1972.
4. Кнут Д.Э. Все про TEX/ Д.Э. Кнут. — М.: Вильямс, 2003.
5. Intel Architecture Optimization Manual. URL: `http://www.intel.com/design/pentium/manuals/242816.htm`.

[эту страницу печатать не надо]