

Вычислительно эффективные алгоритмы решения прямой и обратной задач динамики

Artemov K.

- Что есть прямая и обратная задачи динамики
- Как оценивается эффективность алгоритмов
- Уравнения движения
- Алгоритмы основанные на уравнениях Лагранжа, Ньютона-Эйлера, Аппеля и Кейна
- Обзор некоторых алгоритмов

Прямая и обратная задачи динамики

Прямая задача динамики:

Заданы: обобщенные
силы/моменты;

Найти: определить
траекторию движения.

$$\ddot{q} = \phi(q, \dot{q}, \tau)$$

Обратная задача динамики:

Задана: траектория
движения;

Найти: обобщенные
силы/моменты.

$$\tau = \phi(q, \dot{q}, \ddot{q})$$

где τ – вектор обобщенных сил/моментов робота, q, \dot{q} и \ddot{q} –
векторы положения, скорости и ускорения

Временная сложность алгоритмов:

- $O(1)$ – постоянное время;
- $O(n)$ – линейное время;
- $O(n^2)$ – квадратичное время;
- и др.

Вычислительная сложность алгоритмов:

- количество сложений;
- количество умножений.

- уравнение в конфигурационном пространстве:

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) = \tau$$

где q, \dot{q} и \ddot{q} – положение, скорость и ускорение звена робота, τ – силы/моменты приложенные к звену, $M(q)$ – тензор инерции, $C(q, \dot{q})$ – матрица Кориолисовых и центробежных сил, $G(q)$ – вектор гравитации;

- уравнение в операционном пространстве:

$$\Lambda(x)\ddot{x} + \mu(x, \dot{x}) + \rho(x) = f$$

где x – положение схвата, \dot{x} – скорость схвата, f – действующие силы на схват, $\Lambda(x)$ – матрица инерции в операционном пространстве, $\mu(x, \dot{x})$ – содержит Кориолисовы и центробежные составляющие, $\rho(x)$ – вектор гравитации.

Таблица: Алгоритмы основанные на уравнениях Эйлера-Лагранжа

Форма уравнений	Авторы алгоритма	Число операций		Прямая задача
		*	+	
Лагранж	Uicker/Kahn	66271	51548	+
	Vukobratovic/ Potconjak	37189	5652	+
	Hollerbach 3x3	2195	1719	-
	Renaud	992	776	+

Таблица: Алгоритмы основанные на уравнениях Ньютона-Эйлера

Форма уравнений	Авторы алгоритма	Число операций		Прямая задача
		*	+	
Ньютон-Эйлер	Vukobratovic/ Stepanenko	2907	2068	+
	Walker/Orin	1771	1345	+
	Wang/Ravani	1659	1252	+
	Luh/Walker/ Paul	792	662	-
	Balafoutis/ Patel/Misra	489	420	-

Таблица: Алгоритмы основанные на других уравнениях

Форма уравнений	Авторы алгоритма	Число операций		Прямая задача
		*	+	
Аппель	Попов	2929	2500	+
Кейн	Ма/Хи	1020	851	-

Уравнения динамики для робота с n степенями свободы:

$$P_i = \sum_{j=i}^n \left(\sum_{k=1}^j \left[tr \left(\frac{\partial W_j}{\partial q_i} J_j \frac{\partial W_j^T}{\partial q_k} \right) \ddot{q}_k \right. \right. \\ \left. \left. + \sum_{k=1}^j \sum_{l=1}^j \left[tr \left(\frac{\partial W_j}{\partial q_i} J_j \frac{\partial^2 W_j^T}{\partial q_k \partial q_l} \right) \dot{q}_k \dot{q}_l \right] - m_j \vec{q}^T \frac{\partial W_j}{\partial q_i} r_{jo} \right) \right]$$

где P_i – сила/момент привода, W_i – матрица трансформации от базовой к локальной системе координат i -ого звена, J_i – матрица инерции i -ого звена в локальной системе координат, m_i – масса звена i , r_{jo} – вектор от центра масс звена i к началу базовой системы координат, выраженный в локальной системе координат звена i , \vec{g} – вектор гравитации

Алгоритм Uiker и Kahn (продолж.)

Матрица W_i выражается, как:

$$W_i = A_1^0 A_2^1 \dots A_i^{i-1}$$

где A_i^{i-1} – матрица трансформации 4×4 из $i-1$ в i системам координат

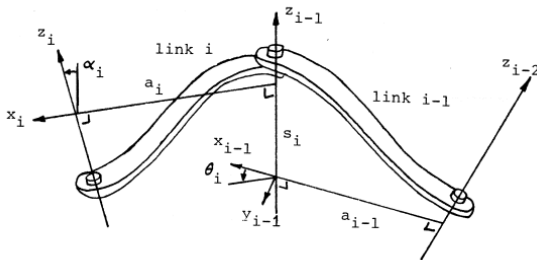


Рис.: Оси координат двух соседних звеньев

Формулы для вычисления частных производных матриц W_i :

$$\frac{\partial W_i}{\partial q_k} = W_{k-1} \frac{\partial A_k^k}{\partial q_k} W_i, \quad (k \leq i)$$

$$\frac{\partial^2 W_i}{\partial q_k \partial q_l} = W_{k-1} \frac{\partial A_k^k}{\partial q_k} W_{l-1} \frac{\partial A_l^l}{\partial q_l} W_i, \quad (k < l \leq i)$$

$$= W_{k-1} \frac{\partial^2 A_k^k}{\partial q_k^2} W_i, \quad (k = i)$$

Алгоритм Uiker и Kahn (продолж.)

Алгоритм – нерекурсивный.

Решение прямой задачи динамики:

$$P = H(q)\ddot{q} + \dot{q}^T C(q)\dot{q} + g(q)$$

Решение обратной задачи динамики:

$$\ddot{q} = H(q)^{-1}(\dot{q}^T C(q)\dot{q} + g(q)) - P$$

Вычислительная сложность:

- количество сложений порядка n^4 ;
- количество умножений порядка n^4 .

Для вычисления частных производных в уравнении:

$$P_i = \sum_{j=i}^n \left[\text{tr} \left(\frac{\partial W_j^i}{\partial q_i} W_j J_j \ddot{W}_j^T \right) - m_j \vec{g}^T \frac{\partial W_j^i}{\partial q_i} W_j r_{io} \right]$$

применены рекурсивные отношения:

$$W_j = W_{j-1} A_j^{j-1}$$

$$\dot{W}_j = \dot{W}_{j-1} A_j^{j-1} + W_{j-1} \frac{\partial A_j^{j-1}}{\partial q_j} \dot{q}_j$$

$$\ddot{W}_j = \ddot{W}_{j-1} A_j^{j-1} + 2\dot{W}_{j-1} \frac{\partial A_j^{j-1}}{\partial q_j} \dot{q}_j + W_{j-1} \frac{\partial^2 A_j^{j-1}}{\partial q_j^2} \dot{q}_j^2 + W_{j-1} \frac{\partial A_j^{j-1}}{\partial q_j} \ddot{q}_j$$

Если представить уравнение движения в следующей форме:

$$P_i = tr\left(\frac{\partial W_i}{\partial q_i}\right)D_i - \vec{g}^T \frac{\partial W_i}{\partial q_i} c_i$$

где

$$D_i = J_i \ddot{W}_i^T + A_{i+1}^i D_{i+1}$$

$$c_i = m_i r_{i0} + A_{i+1}^i c_{i+1}.$$

Угловое ускорение \ddot{W}_i^T вычисляется в прямой рекурсии от базы к схвату. D_i и c_i вычисляются в обратной рекурсии от схвата к базе.

Таим образом, полученные соотношения позволяют достичь $30n - 592$ операций умножения и $675n - 464$ операций сложения.

Алгоритм позволяет рассчитывать инверсную динамику кинематических цепочек с фиксированной базой.

Вычисления производятся в четыре этапа:

- Этап 1: Вычисляется скорость и ускорения для каждого звена, начиная с известных скорости и ускорения базы и заканчивая схватом

$$v_i = {}^i X_{p(i)} v_{p(i)} + \Phi_i \dot{q}_i, (v_o = 0),$$

где v_i – скорость звена i , Φ_i – тип сочленения j , и \dot{q}_i – вектор скорости.

Ускорение:

$$a_i = {}^i X_{p(i)} a_{p(i)} + \Phi_i \ddot{q}_i + \dot{\Phi}_i \dot{q}_i, (a_o = 0),$$

где a_i – ускорение звена j , \ddot{q}_i – вектор ускорения обобщенных переменных сочленения j .

Для того, чтобы учесть влияние ускорения свободного падения на систему, возможно инициализировать $a_o = -g$ вместо нуля. В этом случае a_i будет характеризовать ускорение звена i с учетом ускорения свободного падения.

- Этап 2: На этом этапе рассчитываются силы, которые нужно приложить к звену, чтобы оно приобрело рассчитанное на Этапе 1 ускорение

$$f_i^a = I_i a_i + v_i \times I_i v_i,$$

где I_i вектор инерции звена i , f_i^a – равнодействующая сила на звено i .

- Этап 3: Вычисляется вектор сил через каждое из звеньев

$$f_i = f_i^a - {}^i X_{p(i)} f_i^e + \Sigma_{j \in c(i)} {}^i X_{p(i)} f_j,$$

где f_i – сила передающаяся через звено j , f_i^e – сумма всех внешних сил, действующих на звено i , $c(i)$ множество потомков звена i , $i = n, \dots, 1$.

f_i^e может представлять собой пружины, демпферы, контакты с внешней средой, а также ускорение свободного подения, если он не задан на Этапе 2 и прочее, т.е. заранее известна.

- Этап 4: Вычисляются обобщенные моменты/силы

$$\tau_i = \Phi_i^T f_i$$

Рекурсивный алгоритм Ньютона-Эйлера

```
inputs:  $\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, model, {}^0\mathbf{f}_i^e$ 
output:  $\boldsymbol{\tau}$ 
model data:  $N_B, jtype(i), p(i), X_L(i), I_i$ 

 $\mathbf{v}_0 = \mathbf{0}$ 
 $\mathbf{a}_0 = -\mathbf{a}_g$ 
for  $i = 1$  to  $N_B$  do
     $X_J(i) = xjcalc(jtype(i), q_i)$ 
     ${}^iX_{p(i)} = X_J(i) X_L(i)$ 
    if  $p(i) \neq 0$  then
         ${}^iX_0 = {}^iX_{p(i)} {}^{p(i)}X_0$ 
    end
     $\Phi_i = pcalc(jtype(i), q_i)$ 
     $\dot{\Phi}_i = pdcalc(jtype(i), q_i, \dot{q}_i)$ 
     $\mathbf{v}_i = {}^iX_{p(i)} \mathbf{v}_{p(i)} + \Phi_i \dot{q}_i$ 
     $\zeta_i = \dot{\Phi}_i \dot{q}_i + \mathbf{v}_i \times \Phi_i \dot{q}_i$ 
     $\mathbf{a}_i = {}^iX_{p(i)} \mathbf{a}_{p(i)} + \Phi_i \ddot{q}_i + \zeta_i$ 
     $\mathbf{f}_i = I_i \mathbf{a}_i + \mathbf{v}_i \times I_i \mathbf{v}_i - {}^iX_0^{-T} {}^0\mathbf{f}_i^e$ 
end
for  $i = N_B$  to 1 do
     $\boldsymbol{\tau}_i = \Phi_i^T \mathbf{f}_i$ 
    if  $p(i) \neq 0$  then
         $\mathbf{f}_{p(i)} = \mathbf{f}_{p(i)} + {}^iX_{p(i)}^T \mathbf{f}_i$ 
    end
end
end
```

Входные данные: траектория движения, модель робота, внешние силы;

Выходными данные: моменты приводов;

Функция **jtype** – возвращает тип звена;

Функция **xjtype** – матрицу трансформации для указанного типа звена;

Функции **pcalc** и **pdcalc** вычисляют Φ_i и $\dot{\Phi}_i$;

X – матрица трансформации.

Алгоритмы основанные на уравнениях Аппеля

Динамика механизма описывается уравнениями

Гиббса-Аппеля:

$$\frac{\partial G}{\partial \ddot{q}_i} = Q_i, (i = 1, \dots, n)$$

где G – функция "энергии ускорения" Q_i – обобщенная сила.

Функция "энергии ускорения" может быть выражена как сумма:

$$G = \sum_{i=1}^n G_i$$

где G_i – функция Гиббса для звена i , которая, в свою очередь, определится из выражения:

$$G_i = \frac{1}{2} m_i w_i^2 + \frac{1}{2} \epsilon_i^T J_i \epsilon_i + 2(\omega_i \times J_i \omega_i) \epsilon_i$$

Подставляя в последнее выражения рекурсивные отношения для кинематики робота, получаем динамическую модель робота в форме:

$$H(q)\ddot{q} + h_c(q, \dot{q}) = Q$$

где $Q = P - g(q)$ – вектор обобщенных сил.

$$H(q)\ddot{q} + h(q, \dot{q}) = P$$

где $h(q, \dot{q}) = h_c(q, \dot{q}) + g(p)$.

Этот алгоритм позволяет решать как прямую, так и обратную задачи динамики. Вычислительная сложность:

$\frac{7}{3}n^3 + 27n^2 + \frac{722}{3}n + 9$ – умножений, $\frac{10}{3}n^3 + \frac{43}{2}n^2 + \frac{931}{6}n + 6$ – сложений.

Считается, что робот состоит из n соединенных твердых тел, каждое из которых имеет по 6 степеней свободы.

Относительная ориентация соседних звеньев описывается следующими четырьмя параметрами Эйлера:

$$\begin{aligned}\epsilon_{il} &= e_{il} \sin\left(\frac{q_i}{2}\right), (l = 1, 2, 3) \\ \epsilon_{i4} &= e_{i4} \cos\left(\frac{q_i}{2}\right).\end{aligned}$$

где $e_i = (e_{i1}, e_{i2}, e_{i3})$ – единичный вектор на оси, вокруг которой $i - 1$ локальная система координат переходит в i в результате вращения, q_i – угол, $e_{i4} = 1$. Можно показать, что параметры Эйлера эквивалентны формуле Родрига, когда сочленения с одной степенью свободы связаны.

Можно показать, что параметры Эйлера эквивалентны формуле Родрига, когда сочленения с одной степенью свободы связаны.

$$r' = r + 2\sin(\frac{q_i}{2})e_i \times (e_i \times r) + 2\sin(\frac{q_i}{2})\cos(\frac{q_i}{2})(e_i \times r)$$

где r – вектор до вращения, r' – вектор после вращения на угол q_i вокруг оси e_i . Сделав замену $\epsilon_i = e_i\sin(\frac{q_i}{2})$ и $\epsilon_{i4} = \cos(\frac{q_i}{2})$, получим:

$$r' = r + 2\epsilon_i \times (\epsilon_i \times r) + 2\sin(\frac{q_i}{2})\epsilon_{i4}(\epsilon_i \times r)$$

Кинематические соотношения в этом алгоритме состоят из выражения для скоростей и ускорений звеньев в терминах параметров Эйлера:

$$\omega_i = \sum_{k=1}^i \omega'_k$$

где ω_i – угловая скорость звена i относительно базовой системы координат, ω'_k – относительная угловая скорость звена k относительно звена $k - 1$. Это соотношение можно переписать в рекурсивной форме:

$$\omega_i = \omega_{i-1} + \omega'_i$$

Несложно заметить, что это уравнение совпадает с ранее рассмотренным.

Несмотря на постоянный рост мощности компьютеров, требование высокой вычислительной эффективности уравнений динамики остается критичным. Это объясняется тем, что:

- во-первых, в системах управления роботов используются как правило относительно медленные процессоры, и для решения уравнений динамики в реальном времени необходимы эффективные алгоритмы расчета
- во-вторых, сложность механических структур современных роботов (параллельных, с избыточными степенями подвижности, так называемых роботов-гуманоидов) требует эффективных методов расчета динамики для задач их моделирования и управления.

ВЖУУУХ!!!1!
СПАСИБО ЗА ВНИМАНИЕ!

Artemov K.

ITMO University

june, 2018

research supervisor: A.A. Kapitonov

СПАСИБО ЗА ВНИМАНИЕ!

Artemov K.
ITMO University
june 2018