A Unified Method for Solving Inverse, Forward, and HybridManipulator Dynamics using Factor Graphs

Mandy Xie, Frank Dellaert (presented by Dominic Guri)

Introduction

1. ***Inverse Dynamics***

- *solve for forces & torques* (for control and planning) for a desired trajectories (positions, velocities, and accelerations)
- often used by Newton-Euler N_E — very efficient, and recursion-based

2. ***Forward Dynamics***

- solves joint accelerations when torques are applied at known positions and velocities
  - different from simulation problem with requires motion integration to obtain positions and velocities
- solutions:
  - Composite-Rigid-Body Algorithm (CRBA) — can be numerically unstable
  - Articulated-Body Algorithm (ABA) — takes advantage of sparsity

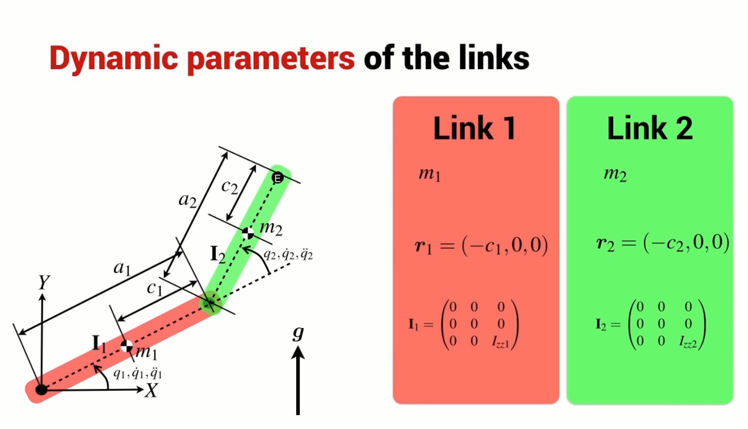
3. ***Hybrid Dynamics (partially both inverse and forward)***

- solves for unknown forces and accelerations, given forces at some joints and accelerations at other joints

NB: all the above methods do not handle kinematic loops

NB: rare for a single algorithm to solve all 3 problems — proposed methods use filtering and smoothing

Dynamics Problems:



Dynamic parameters of the links

Link 1

m_1

$\boldsymbol{r}_1 = (-c_1, 0, 0)$

$\mathbf{I}_1 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & I_{zz1} \end{pmatrix}$

Link 2

m_2

$\boldsymbol{r}_2 = (-c_2, 0, 0)$

$\mathbf{I}_2 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & I_{zz2} \end{pmatrix}$

Figure 1: QUT Robot Academy

Existing Methods

1. ***Inverse Dynamics***

- Recursive Newton-Euler [42,21]
- others [49, 57]

2. ***Forward Dynamics***

- Composite-Rigid-Body Algorithm (CRBA) [60,20]
- Articulated-Body Algorithm (ABA) [18]

3. ***Hybrid Dynamics (partially both inverse and forward)***

- Articulated-Body Hybrid Dynamics Algorithm [19]

4. ***Kinematic Loops***

- [19]
- Rodriguez [52,51] — filtering and smoothing based, **inverse** & **forward**
  - applied to loops in [53]
- Rodriguez [54] ⇒
  - Jain [33] — serial chain dynamics
  - Ascher [4] — CRBA & ABA as elimination methods for solving forward dynamics

Contributions

- **factor graph** as a unified description of **classical dynamics** algorithms and **new algorithms** (graph theory-based ⇒ sparse linear systems)

1. unified method for forward, inverse and hybrid for **chains & loops**
2. factor graph representation for dynamics problems — with better visualization of underlying equations
3. discover **new dynamics equations** due to different elimination algorithms

Manipulator Dynamics Review

- Lynch & Park [43] — modern geometric view, exposition and notation
  - borrows from Brockett [10], and Murray [48]
  - leads to modern differential geometry

Newton-Euler Equations

$$\begin{aligned} f_b &= m\dot{v}_b && +\omega_b \times m v_b \\ \tau_b &= \mathcal{I}_b \dot{\omega}_{b\,b} && +\omega_b \times \mathcal{I}_b \omega_b \end{aligned}$$

where m, $\mathcal{I}_b$, v_b, and ω_b are mass, inertia, linear and angular velocity in the body-frame.

- Geometrically, the equations above are combined by
  - wrench $\mathcal{F}_b = [\tau_b,\ f_b]^T$
  - twist $\mathcal{V}_b = [\omega_b,\ v_b]^T$
  - skew-symmetric matrics: $[\omega_b] = R^T\dot{R}$

$$\mathcal{F}_b = \mathcal{G}_b \dot{\mathcal{V}}_b - \left[ad_{\mathcal{V}_b}\right]^T \mathcal{G}_b \mathcal{V}_b$$

$$\mathcal{G}_b = \begin{bmatrix} \mathcal{I}_b & 0 \\ 0 & m\mathbf{I} \end{bmatrix}, \qquad \left[ad_{\mathcal{V}_b}\right] = \begin{bmatrix} [\omega_b] & 0 \\ [v_b] & [\omega_b] \end{bmatrix}$$

Recursive Dynamics

- Dynamic Constraints:

$$\begin{aligned}
\textit{velocity constraint} \quad 0 &= \mathcal{V}_i - [Ad_{T_{i,i-1}(\theta_i)}]\mathcal{V}_{i-1} - \mathcal{A}_i\dot{\theta}_i \\
\textit{acceleration constraint} \quad 0 &= \dot{\mathcal{V}}_i - [Ad_{T_{i,i-1}(\theta_i)}]\dot{\mathcal{V}}_{i-1} - \mathcal{A}_i\ddot{\theta}_i - [ad_{\mathcal{V}_i}]\mathcal{A}_i\dot{\theta}_i \\
\textit{force balance} \quad 0 &= Ad^T_{T_{i,i-1}(\theta_i)}\mathcal{F}_{i+1} - \mathcal{F}_i + \mathcal{G}_i\dot{\mathcal{V}}_i - [ad_{\mathcal{V}_i}]^T\mathcal{G}_i\mathcal{V}_i \\
\textit{applied torque} \quad 0 &= \mathcal{F}_i^T\mathcal{A}_i - \tau_i
\end{aligned}$$

- transformation between links: $T_{i,i-1}$
  - associated adjoint transformation: $Ad_{T_{i,i-1}}(\theta_i)$
- screw axis for joint i: $\mathcal{A}_i$

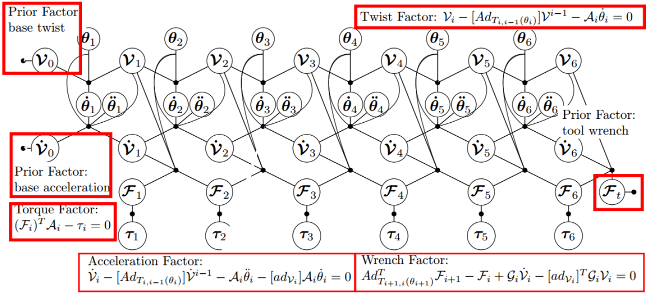
Dynamic Factor Graphs

- Apply structural dynamic equations:

$$
\begin{aligned}
\mathcal{V}_i - [Ad_{T_{i,i-1}(\theta_i)}]\mathcal{V}_{i-1} - \mathcal{A}_i\dot{\theta}_i &= 0\\
\dot{\mathcal{V}}_i - [Ad_{T_{i,i-1}(\theta_i)}]\dot{\mathcal{V}}_{i-1} - \mathcal{A}_i\ddot{\theta}_i - [ad_{\mathcal{V}_i}]\mathcal{A}_i\dot{\theta}_i &= 0\\
Ad^T_{T_{i,i-1}(\theta_i)}\mathcal{F}_{i+1} - \mathcal{F}_i + \mathcal{G}_i\dot{\mathcal{V}}_i - [ad_{\mathcal{V}_i}]^T\mathcal{G}_i\mathcal{V}_i &= 0\\
\mathcal{F}_i^T\mathcal{A}_i - \tau_i &= 0
\end{aligned}
$$

- Nodes (i.e. Variables):
  - $\mathcal{V}_i$ — twist
  - $\dot{\mathcal{V}}_i$ — acceleration
  - $\mathcal{F}_i$ — wrenches
  - θ_i — joint angles
  - $\dot{\theta}_i$ — joint velocity
  - $\dot{\theta}_i$ — joint acceleration
  - τ_i — torques
- NB: $(\theta_i, \dot{\theta}_i)$ are assumed to be known in all problems, therefore $\mathcal{V}_i$ can be evaluated ahead of time
- Additonally, $\mathcal{V}_0$ and $\mathcal{F}_t$ are known

Factors


$\boldsymbol{\theta}_1$ $\boldsymbol{\theta}_2$ $\boldsymbol{\theta}_3$ $\boldsymbol{\theta}_4$ $\boldsymbol{\theta}_5$ $\boldsymbol{\theta}_6$
$\mathcal{V}_0$ $\mathcal{V}_1$ $\mathcal{V}_2$ $\mathcal{V}_3$ $\mathcal{V}_4$ $\mathcal{V}_5$ $\mathcal{V}_6$
$\dot{\boldsymbol{\theta}}_1$ $\ddot{\boldsymbol{\theta}}_1$ $\dot{\boldsymbol{\theta}}_2$ $\ddot{\boldsymbol{\theta}}_2$ $\dot{\boldsymbol{\theta}}_3$ $\ddot{\boldsymbol{\theta}}_3$ $\dot{\boldsymbol{\theta}}_4$ $\ddot{\boldsymbol{\theta}}_4$ $\dot{\boldsymbol{\theta}}_5$ $\ddot{\boldsymbol{\theta}}_5$ $\dot{\boldsymbol{\theta}}_6$ $\ddot{\boldsymbol{\theta}}_6$
$\dot{\mathcal{V}}_0$ $\dot{\mathcal{V}}_1$ $\dot{\mathcal{V}}_2$ $\dot{\mathcal{V}}_3$ $\dot{\mathcal{V}}_4$ $\dot{\mathcal{V}}_5$ $\dot{\mathcal{V}}_6$
$\mathcal{F}_1$ $\mathcal{F}_2$ $\mathcal{F}_3$ $\mathcal{F}_4$ $\mathcal{F}_5$ $\mathcal{F}_6$ $\mathcal{F}_t$
τ_1 τ_2 τ_3 τ_4 τ_5 τ_6
Prior Factor: tool wrench

Prior Factor base twist

Twist Factor: $\mathcal{V}_i - [Ad_{T_{i,i-1}(\theta_i)}]\mathcal{V}^{i-1} - \mathcal{A}_i\dot{\theta}_i = 0$

Prior Factor: base acceleration

Torque Factor: $(\mathcal{F}_i)^T\mathcal{A}_i - \tau_i = 0$

Acceleration Factor: $\dot{\mathcal{V}}_i - [Ad_{T_{i,i-1}(\theta_i)}]\dot{\mathcal{V}}^{i-1} - \mathcal{A}_i\ddot{\theta}_i - [ad_{\mathcal{V}_i}]\mathcal{A}_i\dot{\theta}_i = 0$

Wrench Factor: $Ad^T_{T_{i+1,i}(\theta_{i+1})}\mathcal{F}_{i+1} - \mathcal{F}_i + \mathcal{G}_i\dot{\mathcal{V}}_i - [ad_{\mathcal{V}_i}]^T\mathcal{G}_i\mathcal{V}_i = 0$

Fig. 1: The Puma 560 dynamics factor graph, where black dots represent factors, and circles represent variables.

RRR Robot Arm

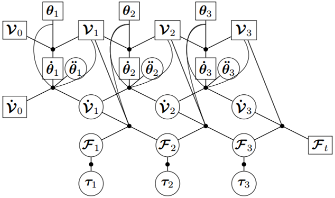


Fig. 3: Dynamic factor graph for a 3R robot, with *known* variables shown as square nodes.

Inverse Dynamics: $\dot{\theta}_i \rightarrow \tau_i$

- Find joint torques τ_i, for a given joint accelerations $\dot{\theta}_i$

Inverse Dynamics Graph

- skip all the known nodes, i.e. square nodes — *but remain available relevant factors*

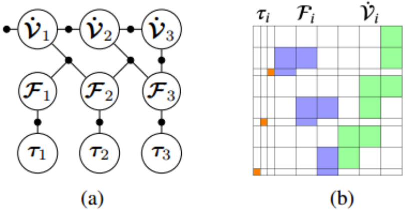
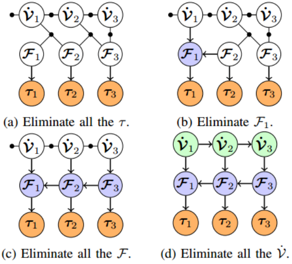


Fig. 4: (a) Simplified inverse dynamics factor graph (b) Block-sparse matrix corresponding to Fig. 4a

Elimination → (DAG)

- *elimination to DAG* — Directed Acyclic Graph
- Order: $\{\tau_3 \cdots \tau_1, \mathcal{F}_1 \cdots \mathcal{F}_3, \dot{\mathcal{V}}_3 \cdots \dot{\mathcal{V}}_1\}$



(a) Eliminate all the τ.

(b) Eliminate $\mathcal{F}_1$.

(c) Eliminate all the $\mathcal{F}$.

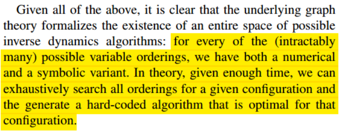
(d) Eliminate all the $\dot{\mathcal{V}}$.

Fig. 5: Steps in the Elimination Algorithm.

- Matches recursive Newton-Euler

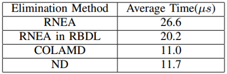
Symbolic Elimination

$$\begin{aligned} 0 &= \mathcal{F}_3^T \mathcal{A}_3 - \tau_3 \\ \tau_3 &= \mathcal{F}_3^T \mathcal{A}_3 \end{aligned}$$



Given all of the above, it is clear that the underlying graph theory formalizes the existence of an entire space of possible inverse dynamics algorithms: for every of the (intractably many) possible variable orderings, we have both a numerical and a symbolic variant. In theory, given enough time, we can exhaustively search all orderings for a given configuration and the generate a hard-coded algorithm that is optimal for that configuration.

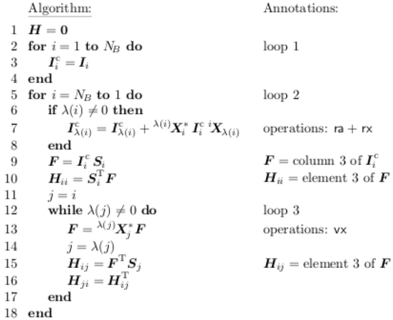
TABLE I: Numerical inverse dynamics for a PUMA 560



Elimination Method	Average Time(μs)
RNEA	26.6
RNEA in RBDL	20.2
COLAMD	11.0
ND	11.7

Forward Dynamics: $\{\theta, \dot{\theta}, \tau\} \rightarrow \ddot{\theta}$

CRBA — Composite Rigid Body Algorithm



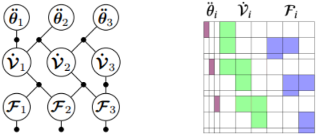
```
Algorithm:                                                        Annotations:

$\boldsymbol{H} = \boldsymbol{0}$
for $i = 1$ to $N_B$ do                                           loop 1
    $\boldsymbol{I}_i^c = \boldsymbol{I}_i$
end
for $i = N_B$ to $1$ do                                           loop 2
    if $\lambda(i) \neq 0$ then
        $\boldsymbol{I}_{\lambda(i)}^c = \boldsymbol{I}_{\lambda(i)}^c + {}^{\lambda(i)}\boldsymbol{X}_i^* \, \boldsymbol{I}_i^c \, {}^i\boldsymbol{X}_{\lambda(i)}$       operations: ra + rx
    end
    $\boldsymbol{F} = \boldsymbol{I}_i^c \boldsymbol{S}_i$                                  $\boldsymbol{F}$ = column 3 of $\boldsymbol{I}_i^c$
    $\boldsymbol{H}_{ii} = \boldsymbol{S}_i^{\mathrm{T}} \boldsymbol{F}$                                $\boldsymbol{H}_{ii}$ = element 3 of $\boldsymbol{F}$
    $j = i$
    while $\lambda(j) \neq 0$ do                                  loop 3
        $\boldsymbol{F} = {}^{\lambda(j)}\boldsymbol{X}_j^* \boldsymbol{F}$                           operations: vx
        $j = \lambda(j)$
        $\boldsymbol{H}_{ij} = \boldsymbol{F}^{\mathrm{T}} \boldsymbol{S}_j$                            $\boldsymbol{H}_{ij}$ = element 3 of $\boldsymbol{F}$
        $\boldsymbol{H}_{ji} = \boldsymbol{H}_{ij}^{\mathrm{T}}$
    end
end
```

Table 10.2: Calculations performed by the composite-rigid-body algorithm

ABA — Articulated Body Algorithm

Forward Dynamics: $\{\theta, \dot{\theta}, \tau\} \rightarrow \ddot{\theta}$

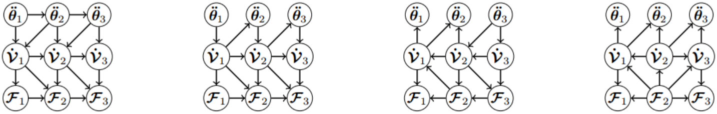
Forward Dynamics Graph



(a) (b)

Fig. 6: (a) Simplified forward dynamics graph and (b) corresponding block-sparse matrix.

CRBA & ABA equivalent Graphs



(a) CRBA ordering. (b) ABA ordering. (c) COLAMD ordering. (d) ND ordering.

Fig. 7: DAGs generated by solving forward dynamics factor graph with different variable elimination orderings.

TABLE II: Numerical forward dynamics for a PUMA 560

Elimination Method	Average Time(μs)
CRBA	51.8
ABA	25.5
COLAMD	11.2
ND	12.1

Hybrid Dynamics: $\{\ddot{\theta}_i, \tau_j\} \rightarrow \{\ddot{\theta}_j, \tau_i\}, \quad j \neq i$

- **Inverse-dynamics joints**: $\ddot{\theta}_i \rightarrow \tau_i$
- **Forward-dynamics joints**: $\tau_j \rightarrow \ddot{\theta}_j$

Example

- $\{\ddot{\theta}_1, \tau_2, \tau_3\} \rightarrow \{\tau_1, \ddot{\theta}_2, \ddot{\theta}_3\}$

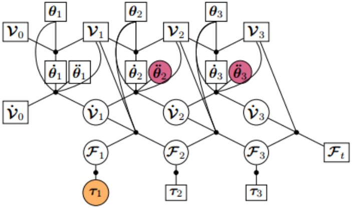


Fig. 8: Hybrid dynamics factor graph for a 3R robot.

Featherstone's Method

- Section VI-A: Hybrid Dynamics

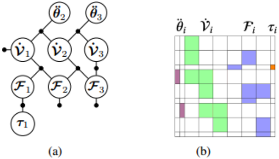


Fig. 9: (a) Simplified hybrid dynamics graph and (b) corresponding block-sparse.

Dynamics for Closed Kinematic Loops

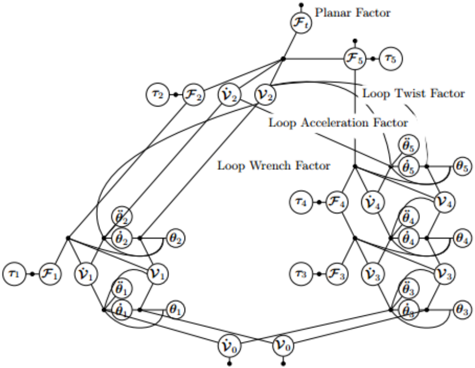


Fig. 12: Five-bar parallel manipulator dynamics factor graph, where black dots represent variable constraints, namely factors, and circles represent variable nodes.

- **Redundantly Actuated**

  Actuated Joints $>$ DOFs

- *infinitely* many values of τ that produce the same $\ddot{\theta}$
- unique solutions are obtained by:
  1. adding constraints, OR
  2. applying an optimality criterion — adding more factors to the graph

- **Example**
- minimum torque factors
- forward dynamics: overconstrained kinematic loops, *the constraint forces exerted by loop joints are underdetermined*
  - use less constrained joints//actuators
  - factor graphs: add planar constraint that reduces unknown forces

Notes

- Opportunities for parallelism (`VI-B`)— “being sophisticated about variable ordering and the possible resulting parallelism could yield large dividends. An important step forward in the understanding and analysis of variable elimination on graphs was the discovery of clique trees, that make the inherent parallelism in the elimination algorithm explicit”