Automatic Generation of Dynamics for Modular Robots with Hybrid Geometry

I-Ming Chen  Guilin Yang
School of Mechanical and Production Engineering
Nanyang Technological University
Singapore 639798

Abstract—Manual derivation of the dynamic model of a modular robot is almost impossible because it may have very different geometries and DOFs through module reconfiguration. This paper presents an algorithm to automatically generate the closed-form equation of motion of a modular robot from a kinematic graph based representation of the assembly configuration. We consider modular robots with the more general branching geometry. The formulation of the dynamic model is started with recursive Newton-Euler algorithm. The generalized velocity, acceleration, and forces are expressed in terms of linear operations on se(3), the Lie algebra of the Euclidean group SE(3). Based on the equivalence relationship between the recursive formulation and the closed-form Lagrangian formulation, we use the accessibility matrix of the kinematic graph to assist the construction of the closed-form equation of motion of a modular robot. Applications of the closed-form dynamic model of a branching robot are in robot design, calibration, and motion optimization.

1 Introduction

A modular reconfigurable robot system consists of a collection of individual link and joint components which can be assembled together to form robots with different DOFs and geometries (e.g., serial type, branching type, etc.). Such a system can provide flexibility, economics, swiftness, and upgradability that a conventional industrial robot with fixed geometry fails to deliver. Several prototyping systems have been demonstrated in various research institutions [5], [10], [21], [22]. Applications of such modular systems have been seen in rapid deployable robot systems for hazardous material handling [15], space stationed autonomous systems [1], and manufacturing systems [4].

The conventional way to derive the dynamic model of a robot depends on the geometry of the robot. Once it is given, for instance, a 6-DOF articulate PUMA type or a 4-DOF SCARA type, the dynamic model can be derived manually. For a modular robot system, one can build numerous robots with distinct geometries out of an inventory of robot modules [3]. Therefore, it is possible and impractical to derive the dynamic model of every modular robot geometry case by case and store it as a library function in the computer.

This paper presents a methodology to automatically generate the closed-form dynamic model of a modular robot from a kinematic graph based representation of the assembly configuration of a modular robot, termed Assembly Incidence Matrix (AIM) [2], [3]. This matrix contains information regarding the geometry of a completed robot and module assembly sequence. The more general branching type robots are considered as they are gaining popularity in industry, construction, and multi-legged walking systems [20]. The formulation of the dynamic model is started with recursive Newton-Euler algorithm [11], [12], [13], [19]. The generalized velocity, acceleration, and forces are expressed in terms of linear operations on se(3), the Lie algebra of the Euclidean group SE(3) [14]. Inertia properties of the link modules can be obtained from the AIM. Based on the relationship between the recursive formulation and the closed-form Lagrangian formulation for serial robot dynamics discussed in [9], [17], [18], we use an accessibility matrix [7] to assist the construction of the closed-form equation of motion of a branching type modular robot. Applications of the closed-form dynamic model are in robot design, motion optimization, optimal control and high-level manipulation.

2 Robot Modules

A novel modular reconfigurable robot system has been proposed by Chen and Yang [4] for factory automation purpose. The main feature of this system is to use a minimal number of modules to construct a robot geometry for a specific task so as to increase the swiftness of task operation and bandwidth of the system. All modules are designed as cubes so that they can be connected in many different orientations. Three types of modules are considered: cubic modules, revolute modules (Fig. 1), and prismatic modules (Fig. 2). The latter two types of modules contain built-in rotary and linear actuators respectively. They are considered as "links". Modules are connected through connectors. The connectors are considered as "joints" between the modules. The con-
necting points on the modules are termed connecting sockets. The connector is treated as a “revolute joint” when fastened to the rotating socket of the revolute module; a “prismatic joint” when fastened to the translating socket of the prismatic module; a “virtual joint” when fastened to two fixed sockets of any modules. The virtual joint does not allow any joint displacement, i.e., the two modules are rigidly connected together.

3 Module Assembly Representation

The Assembly Incidence Matrix (AIM) representation of a modular robot assembly is proposed by Chen [2], [3]. It is based on the concept of kinematic graphs from mechanism analysis and synthesis [8]. A robot manipulator, like a kinematic chain, is a collection of links and joints, and thus, admits a graph representation in which vertices represent links and edges represent joints. For example, a 5-DOF branching type modular robot and its kinematic graph are shown in Fig. 3 and 4 respectively. It is known that a graph can be represented numerically as a vertex-edge incidence matrix in which the entries contain only 0s and 1s [7]. Entry \((i, j)\) is equal to 1 if edge \(j\) is incident on vertex \(i\), otherwise, it is equal to zero. The AIM is formed by substituting the 1s with the labels of the connected sockets on respective modules. The AIM of the robot of Fig. 3 is given as follows.

\[
\begin{pmatrix}
(x, y) & (x, y) & (-x, y) & 0 & 0 & 0 & 1_{p_1} \\
(0, 0) & (0, 0) & (0, 0) & 0 & 0 & 1_{c_2} \\
0 & 0 & 0 & (z, y) & 0 & 0 & 1_{r_1} \\
0 & 0 & 0 & (z, x) & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & (-z, x) & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & (-z, y) & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\]

The incidence matrix describes the topology of the robot. The labels are now written in a port vector format indicating the relative orientation of the module coordinate frames [4]. The types of links and joints used in the robot assembly are shown in the additional row and column in the matrix. \(1_{p_i}\), \(1_{r_i}\), and \(1_{c_i}\) denote prismatic, revolute, and cubic link modules respectively. \(J_{p_i}\), \(J_{r_i}\), and \(J_{c_i}\) denote prismatic, revolute, and virtual joints respectively.

4 Module Traversing Orders

In the recursive algorithm of a serial type robot, the forward iterations of velocities and accelerations and the backward iterations of forces follow the natural order of links. However, for a branching type robot, no such natural order exists. The order of the links of a branching robot depends on the graph traversing algorithm employed [6]. The order of the links of a branching robot is denoted by the graph traversing algorithm employed [6]. Let \(G = (V, E)\) represent the kinematic graph of a branching modular robot with \(n + 1\) link modules. \(V = \{v_0, v_1, \ldots, v_n\}\) represents the set of modules. The fixed base module is denoted by \(v_0\) and is always the starting point for the traversing algorithm. The rest modules are labeled by their traversing orders \(i\). The traversing orders of the links in the robot of Fig. 3 are indicated by the numbers on the vertices of the graph of Fig. 4. This order is obtained by using Depth-First-Search algorithm [6]. Note that the farther the module is away from the base, the larger of its traversing order. A branching (tree-structured) robot with \(n + 1\) modules has \(n\) joints. Let \(E = \{e_1, \ldots, e_{n-1}\}\) represent the set of joints. Joint \(e_i\) is designated as the connector preceding link module \(v_i\).

With a given traversing order, the robot graph \(G\) can be converted to a directed graph (or digraph) \(G\). Let \(e_j = (v_i, v_j)\) be an edge of the graph \(G\) and \(i < j\). An arrow is drawn from \(v_i\) to \(v_j\) as edge \(e_j\) leaves vertex \(v_i\) and enters vertex \(v_j\). Suffice to say, link \(v_i\) precedes link \(v_j\). An example of the directed graph is also shown in Fig. 4. From a digraph with \(n\) vertices, an \(n \times n\) accessibility
Fig. 5: Link assembly \( j \) connected to link \( i \)

Matrix can be defined to indicate the accessibility among the vertices.

**Definition 1** The accessibility matrix of a directed kinematic graph \( \mathcal{G} \) of a modular robot with \( n+1 \) modules (vertices) is an \((n+1) \times (n+1)\) matrix, \( \mathcal{A}(\mathcal{G}) = [a_{ij}] \) \((i,j = 0, \ldots, n)\) such that \( a_{ij} = 1 \) if there is a directed path of length one or more from \( v_i \) to \( v_j \); \( a_{ij} = 0 \), otherwise.

For instance, the accessibility matrix of \( \mathcal{G} \) in Fig. 4 is

\[
\begin{bmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

5 Modular Robot Dynamics

Let \( v_i \) and \( v_j \) be two adjacent links connected by joint \( e_j \) as shown in Fig. 5 and \( v_i \) precede \( v_j \). Denote the module coordinate frame on \( v_i \) by frame \( i \), \( v_j \) by frame \( j \). Frame \( i \) and \( j \) are located at the centroid of the cubic modules not the center of the masses. Note that joint \( e_j \) and link \( v_j \) are rigidly connected together, termed Link assembly \( j \).

5.1 Newton-Euler Equation for Link Assembly

Now consider two coordinate frames located in link assembly \( j \) shown in Fig. 6. Link frame \( j \) is at the centroid of the cubic module. Frame \( j \) is the body (or local) coordinate frame used in the modular robot kinematics [4]. The mass frame \( j^* \) is at the center of mass of the entire link assembly \( j \). In general, these two frames do not coincide with each other. Let \( T_{j^*j} = (p_{j^*j}, R_{j^*j}) \in SE(3) \) be the configuration of link frame \( i \) relative to mass frame \( j^* \), where \( p_{j^*j} = (p_x, p_y, p_z)^T \in \mathbb{R}^{3 \times 1} \) and \( R_{j^*j} \in SO(3) \) are the position vector and rotation matrix respectively. Note that \( T_{j^*j} \) is a constant because frame \( j \) and \( j^* \) belong to the same rigid link assembly. The Newton-Euler equation of this rigid link assembly with respect to its body coordinate at the center of the mass is thus [14],

\[
F_{j^*} = \begin{bmatrix} I_{j^*} & 0_{j^*} \end{bmatrix} + \begin{bmatrix} -J_{j^*} & m_{j^*}v_{j^*} \end{bmatrix} \times m_{j^*}v_{j^*} \]
\[
- \begin{bmatrix} I_{j^*} & 0_{j^*} \end{bmatrix} \begin{bmatrix} J_{j^*} \end{bmatrix} + \begin{bmatrix} -J_{j^*} & m_{j^*}v_{j^*} \end{bmatrix} \times m_{j^*}v_{j^*} \]
\]

(1)

Based on the fact that \( v_{j^*} \times m_{j}v_{j} = 0 \), this equation can be written as

\[
F_{j^*} = \begin{bmatrix} m_{j}I_{j^*} & 0_{j^*} \end{bmatrix} \begin{bmatrix} \dot{v}_{j^*} \\
\dot{w}_{j^*} \end{bmatrix} - \begin{bmatrix} 0_{j^*} & m_{j}I_{j^*} \end{bmatrix} \begin{bmatrix} \dot{v}_{j^*} \\
\dot{w}_{j^*} \end{bmatrix} \]
\]

(2)

where \( F_{j^*} \in \mathbb{R}^{6 \times 1} \) is the resultant wrench applied to the center of mass relative to frame \( j^* \); \( f_{j^*} \) and \( \tau_{j^*} \in \mathbb{R}^{3 \times 1} \) are the total forces and moments applied to the link assembly; \( M_{j^*} = \begin{bmatrix} m_{j} & 0 \\
0 & J_{j^*} \end{bmatrix} \in \mathbb{R}^{6 \times 6} \) is the generalized mass matrix. The total mass of link assembly \( j \) is \( m_j \) (sum of link \( v_j \) and joint \( e_j \)). \( J_{j^*} \) is the inertia tensor of the link assembly about mass frame \( j^* \). Transforming equation (2) into the adjoint representation, we have

\[
F_{j^*} = M_{j^*} \dot{V}_{j^*} - ad^{T}_{V_{j^*}} \left( M_{j^*} \dot{V}_{j^*} \right) \]

(3)

The following notations are adopted in (3):

- \( V_{j^*} = \begin{bmatrix} \dot{v}_{j^*} \\
\dot{w}_{j^*} \end{bmatrix} \in \mathbb{R}^{6 \times 1} \) is the generalized body velocity. \( v_{j^*} \) and \( w_{j^*} \) are \( 3 \times 1 \) vectors defining body translational velocity, \( v_{j^*} = (v_x, v_y, v_z)^T \), and the angular velocity, \( w_{j^*} = (\omega_x, \omega_y, \omega_z)^T \), respectively.
- \( \dot{v}_{j^*} \) and \( \dot{w}_{j^*} \in \mathbb{R}^{3 \times 3} \) are skew symmetric matrices related to \( v_{j^*} \) and \( w_{j^*} \), respectively.
- \( \dot{V}_{j^*} = \begin{bmatrix} I_{j^*} & 0_{j^*} \\
0_{j^*} & I_{j^*} \end{bmatrix} \)
- \( \dot{M}_{j^*} = \begin{bmatrix} -\dot{w}_{j^*} & -\dot{w}_{j^*} \end{bmatrix} \in \mathbb{R}^{6 \times 6} \) is the transpose of adjoint matrix \( \dot{m}_{j^*} \), related to \( V_{j^*} \) [14] and

\[
ad^{T}_{V_{j^*}} = \dot{m}_{j^*} \]
\]

(4)

Equation (3) is relative to mass frame \( j^* \). To be compatible with the kinematic model of modular robots described in [4], it must be transformed to the module frame \( j \). According to the wrench transformation mentioned in [14], the wrench \( F_{j^*} \) can be transformed into an equivalent wrench \( F_j \) expressed in link frame \( j \) by

\[
F_j = Ad^{T}_{T_{j^*j}} F_{j^*} \]

(5)
where \(\text{Ad}^T_{j^*j}\) is the transpose of \(\text{Ad}_{j^*j}\) and
\[
\text{Ad}^T_{j^*j} = (\text{Ad}_{j^*j})^T = \begin{bmatrix}
R^T_{j^*j} & \rho_j^* R^T_{j^*j} \\
0 & \rho_j^* \end{bmatrix}
\]
\[
= \begin{bmatrix}
R^T_{j^*j} & 0 \\
-R^T_{j^*j} R^T_{j^*j} & R^T_{j^*j}
\end{bmatrix}
\]  
(6)

Note that \(\rho_j^*\) is the 3 x 3 skew symmetric matrix representation of \(p_j^*\). Similarly, \(\text{Ad}_{j^*j}\) and \(\text{Ad}^T_{j^*j}\) can be transformed into the equivalent \(V_j\) and \(V_j\) in link frame \(j\) as follows.
\[
V_j = \text{Ad}_{j^*j} V_j 
\]
(7)
\[
V_j = \text{Ad}^T_{j^*j} \dot{V}_j
\]
(8)

Substituting equations (5), (7), and (8) into (3), and using the identity \([16]\),
\[
\text{Ad}^T_{j^*j} \text{ad} \dot{v}_i = \text{ad} \text{Ad}^T_{j^*j} V_j \text{Ad}^T_{j^*j}
\]
we have
\[
F_j = M_j \dot{v}_j - \text{ad} \text{Ad}^T_{j^*j} V_j \text{Ad}^T_{j^*j}
\]
(10)

This is the Newton-Euler equation of link assembly \(i\) written in the link frame \(i\), not the frame at the center of mass. The generalized mass matrix \(M_j\) is relative to link frame \(j\) and has the following form:
\[
M_j = \begin{bmatrix}
M_{j^*j} & M_j \rho_j^* R^T_{j^*j} \\
-M_j \rho_j^* & R^T_{j^*j} (J_j^* - M_j \rho_j^* R^T_{j^*j})
\end{bmatrix}
\]
(11)

For simplicity, we assume that the coordinate axes of mass frame \(j^*\) are parallel to the axes of link frame \(j\), then \(R^T_{j^*j} = I\). The generalized mass matrix \(M_j\) becomes
\[
M_j = \begin{bmatrix}
M_{j^*j} & M_j \rho_j^* R^T_{j^*j} \\
-M_j \rho_j^* & J_j^* - M_j \rho_j^* R^T_{j^*j}
\end{bmatrix}
\]
(12)

Equations (3) and (10) are of the same form which shows the coordinate invariance property of Newton-Euler equation in body coordinates.

### 5.2 Recursive Newton-Euler Algorithm

The recursive algorithm is a two-step iteration process. For a branching type robot, the generalized velocity and acceleration of each link are propagated from the base to the tips of all branches. The generalized force of each link is propagated backward from the tips of the branches to the base. At the branching module, generalized forces transmitted back from all branches are summed up.

**Forward Iteration**

The general velocity and acceleration of the base link will be given initially,
\[
\dot{V}_b = V_0 = (0, 0, 0, 0, 0, 0)^T
\]
(14)
\[
\ddot{V}_b = \ddot{V}_0 = (0, 0, 0, 0, 0, 0)^T
\]
(15)

where \(V_0\) and \(\dot{V}_0\) are expressed in the base frame \(0\). We assume that the base frame coincides with the spatial reference frame. The generalized acceleration (15) is initialized with the gravitation acceleration \(g\) so that later forward iteration steps do not have to consider the gravity term. Referring to Fig. 5, the recursive body velocity and acceleration equations can be written as
\[
V_j = \text{Ad}^T_{j^*j} (V_i) + s_j \dot{q}_j
\]
(16)
\[
\dot{V}_j = \text{Ad}^T_{j^*j} (\dot{V}_i) + \text{ad} \text{Ad}^T_{j^*j} (s_j \dot{q}_j) + s_j \ddot{q}_j
\]
(17)

where all the quantities, if not specified, are expressed in link frame \(j\).

- \(V_j\) and \(\dot{V}_j\) are the generalized velocity and acceleration of link assembly \(j\) written in link frame \(j\).
- \(\dot{q}_j\) and \(\ddot{q}_j\) are the velocity and acceleration of joint \(e_j\) respectively.
- \(\text{Ad}^T_{j^*j}\) is the adjoint representation of \(T_{j^*j}(q_j)\), where \(T_{j^*j}(q_j) = S_{\theta j} T_{j^*j}\) is the position of frame \(j^*\) relative to frame \(i\) with joint displacement \(q_j\) and \(\text{Ad}^T_{j^*j} = (\text{Ad}_{j^*j})^{-1}\).
- \(s_j \in R^{6\times1}\) is the twist coordinates of joint \(e_j\), and \(s_j = (0, 0, 0, 0, 0, 0)^T\) for a virtual joint.

Both \(\text{Ad}^T_{j^*j}\) and \(s_j \dot{q}_j\) are elements of \(se(3)\), and
\[
\text{ad} \text{Ad}^T_{j^*j} (s_j \dot{q}_j) = -\text{ad} s_j \dot{q}_j (\text{Ad}^T_{j^*j} (V_i)).
\]
(18)

Substituting (16) into (18), we find that
\[
\text{ad} \text{Ad}^T_{j^*j} (s_j \ddot{q}_j) = -\text{ad} s_j \ddot{q}_j (\text{Ad}^T_{j^*j} (V_i)).
\]
(19)

Therefore, equation (17) can also be written as
\[
\dot{V}_j = \text{Ad}^T_{j^*j} (V_i) + \text{ad} s_j \dot{q}_j (V_j) + s_j \ddot{q}_j
\]
(20)

**Backward Iteration**

The backward iteration of a branching robot starts from all the pendant link assembly simultaneously. Let \(V_{PD} \subseteq V\) be set of the pendant links of a branching robot. For every pendant link assembly \(d_i\) (\(V_{di} \in V_{PD}\), the Newton-Euler equation (10) can be written as
\[
F_{di} = -F_{di} + M_{di} \dot{V}_{di} - \text{ad} \text{Ad}^T_{di} (M_{di} \dot{V}_{di})
\]
(21)

where \(F_{di}\) is the wrench exerted on link assembly \(V_{di}\) by its parent (preceding) link relative to frame \(d_i\); \(F_{di}^e\) is the external wrench exerted on \(V_{di}\). Note that the total wrench \(F_d = F_{di} + F_{di}^e\).

Now traverse the links in the branching robot backward from the pendant links. Let \(V_{HI}\) be the set of successors of link \(v_i\). For every link assembly \(i\), the Newton-Euler equation (10) can be written in the following form:
\[
F_i = \sum_{j \in V_{HI}} \text{Ad}^T_{j^*j} (F_j) - F_i^e + M_i \dot{V}_i - \text{ad} \text{Ad}^T_{j^*j} (M_i \dot{V}_i)
\]
(22)

where all quantities, if not specified, are expressed in link frame \(i\); \(F_i \in R^{6\times1}\) is the wrench exerted to link
assembly i by its predecessor; \( F_j \in R^{6 \times 1} \) is the wrench exerted by link assembly i to the successor \( v_j \in V_{Hi} \) expressed in link frame j; \( F_l \) is the external wrench applied to link assembly i. Basically, the total wrench \( F_i = F_i - \sum_{j \in V_{Hi}} Ad_{r_{j_i}}^T(F_j) + F_i^e \).

The applied torque/force to link assembly i by the actuator at its input joint \( e_i \), can be calculated by

\[
\tau_i = s_i^T F_i \tag{23}
\]

5.3 Closed Form Equations of Motion

By expanding the recursive Newton-Euler equations (14), (15), (16), (20), (21), and (22) in the body coordinates iteratively, we obtain the generalized velocity, generalized acceleration, and generalized force equations in matrix form:

\[
V = GS\dot{q} \tag{24}
\]

\[
\dot{V} = G_{T_0} \dot{V}_0 + GS\ddot{q} + GA_1V \tag{25}
\]

\[
F = G^TF^e + G^TMV + G^TA_2MV \tag{26}
\]

\[
\tau = S^TF \tag{27}
\]

where

\( V = \text{column}[V_1, V_2, \ldots, V_n] \in R^{6n \times 1} \) is the generalized body velocity vector;

\( V = \text{column}[\dot{V}_1, \dot{V}_2, \ldots, \dot{V}_n] \in R^{6n \times 1} \) is the generalized body acceleration vector;

\( F = \text{column}[F_1, F_2, \ldots, F_n] \in R^{6n \times 1} \) is the body wrench vector;

\( \tau = \text{column}[\tau_1, \tau_2, \ldots, \tau_n] \in R^{6 \times 1} \) is the applied joint torque/force vector;

\( q = \text{column}[\dot{q}_1, \dot{q}_2, \ldots, \dot{q}_n] \in R^{6n \times 1} \) is the joint velocity vector;

\( \dot{q} = \text{column}[\ddot{q}_1, \ddot{q}_2, \ldots, \ddot{q}_n] \in R^{6n \times 1} \) is the joint acceleration vector;

\( \dot{V}_0 = (0, 0, g, 0, 0, 0)^T \in R^{6 \times 1} \) is the generalized acceleration of the base link;

\( S = \text{diag}[s_1, s_2, \ldots, s_n] \in R^{6n \times n} \) is the joint twist matrix written in the respective body coordinates;

\( M = \text{diag}[M_1, M_2, \ldots, M_n] \in R^{6n \times 6n} \) is the total generalized mass matrix;

\( A_1 = \text{diag}[-ad_{r_{i_1}i}, -ad_{r_{i_2}i}, \ldots, -ad_{r_{i_n}i}] \in R^{6n \times 6n}; \)

\( A_2 = \text{diag}[-ad_{r_{i_1}i}, -ad_{r_{i_2}i}, \ldots, -ad_{r_{i_n}i}] \in R^{6n \times 6n}; \)

\( F^e = \text{column}[F_1, F_2, \ldots, F_n] \in R^{6n \times 1} \) is the external wrench vector;

\[
G_{T_0} = \begin{bmatrix}
Ad_{r_{i_1}i} \\
Ad_{r_{i_2}i} \\
\vdots \\
Ad_{r_{i_n}i}
\end{bmatrix} \in R^{6n \times 6}
\]

Note that \( \mathcal{A}(G) = [a_{ij}] \in R^{(n+1) \times (n+1)} \) is the accessibility matrix of the robot’s kinematic digraph \( G \). The matrix \( G \) is termed a transmission matrix. The element \( a_{ij} Ad_{r_{ij}} \) relates link frame i and j if link \( v_i \) can reach \( v_j \). Substituting (24~26) to (27), we obtain the closed-form equation of motion for a branching modular robot with \( n + 1 \) modules (including the base module)

\[
M(q)\ddot{q} + C(q, \dot{q})\dot{q} + N(q) = \tau \tag{28}
\]

where

\[
M(q) = S^T G^T M G S \tag{29}
\]

\[
C(q, \dot{q}) = S^T G^T (M G A_1 + A_2 M) G S \tag{30}
\]

\[
N(q) = S^T G^T M G_{T_0} \dot{V}_0 + S^T G^T F^e \tag{31}
\]

\( M(q) \) is the mass matrix; \( C(q, \dot{q}) \) represents the centrifugal and Coriolis accelerations; \( N(q) \) represents the gravitational force and external forces.

5.4 Implementation

To automatically generate equation (28) from a given AIM, we need to determine the contents of the matrices \( M(q), C(q, \dot{q}) \) and \( N(q) \).

1. The joint twists in \( S \) are represented in local module frames and are usually parallel to the coordinate axes [4]. They can be obtained from the AIM directly and remain unchanged during the robot motion.

2. The inertia properties of robot modules can be measured individually. The AIM shows how the joints are connected to the cubic link modules. Then the generalized mass matrix \( M_j \) relative to the center of mass of the link assembly and \( M_j \) relative to the centroid of the cubic module of module j can be derived. Detail process to obtain these two matrices is described in [23].

3. The accessibility matrix of a graph can be obtained from the powers of its adjacency matrix [7]. The adjacency matrix of the robot’s kinematic graph can be derived from the AIM easily since the incidence relation implies the adjacency information of the vertices (or modules).

4. We proposed an algorithm, TreeRobotKinematics(), in [4] to determine the forward transformations of every module in a branching type modular robot with a set of joint displacements and an AIM, i.e, \( T_{ij}(0) \) and \( T_{ij}(q_j) \). Hence, all the adjoint operations in \( G \) and \( G_{T_0} \) can be calculated at a particular joint displacement \( q \).

The procedure to obtain the closed-form equation (28) can be summarized in the following steps:
In general, the dynamics of a robot manipulator can be formulated in either recursive Newton-Euler or closed-form Lagrangian style. The recursive Newton-Euler method is computationally efficient. However, many applications still rely on explicit closed-form expression of the equation of motion. In a branching modular robot, multiple manipulator branches are dynamically coupled. It is not intuitive and advisable to derive the closed-form dynamic model of a branching modular robot. Note that symbolic expression of the dynamic equation can be obtained through this procedure. It has been successfully implemented in Mathematica code.

6 Summary

In general, the dynamics of a robot manipulator can be formulated in either recursive Newton-Euler or closed-form Lagrangian style. The recursive Newton-Euler method is computationally efficient. However, many applications still rely on explicit closed-form expression of the equation of motion. In a branching modular robot, multiple manipulator branches are dynamically coupled. It is not intuitive and advisable to derive the closed-form dynamic model manually. In this paper, we introduce an algorithm to automatically generate closed-form dynamic model of a branching modular robot from a given AIM. The AIM contains information regarding robot geometry and DOFs. Generalized mass properties of the connected modules can also be derived from the AIM. The closed-form equation of motion is obtained through a recursive Newton-Euler algorithm using standard ideas from Lie groups and Riemannian geometry. The accessibility matrix technique is employed for the equation of motion of a branching robot. This equation of motion can be applied to robot design, calibration, motion optimization, and optimal control.

References