ICINCO2010

# Dynamic Modeling of  Robots using Recursive Newton-Euler Techniques

**Wisama KHALIL**

**IRCCyN- Ecole Centrale de Nantes-France**

**wisama.khalil@irccyn.ec-nantes.fr**

## Motivation

- The research on Dynamic Modeling is one of the most ancient topics in Robotics.
- There is now a consensus for using Recursive Newton-Euler for serial manipulators: Efficiency and facility of implementation
- The aim of this presentation is to show the generalization of these techniques for other systems.

## Outline

- **Recall of the MDH description**
- **The inverse and direct dynamic models of different robotics systems will be presented.**
- **We start by rigid tree structure robots.**
- **These algorithms will be generalized for closed loop robots, parallel robots, and robots with lumped elasticity.**
- **At the end the case of robots with moving base will be treated.**

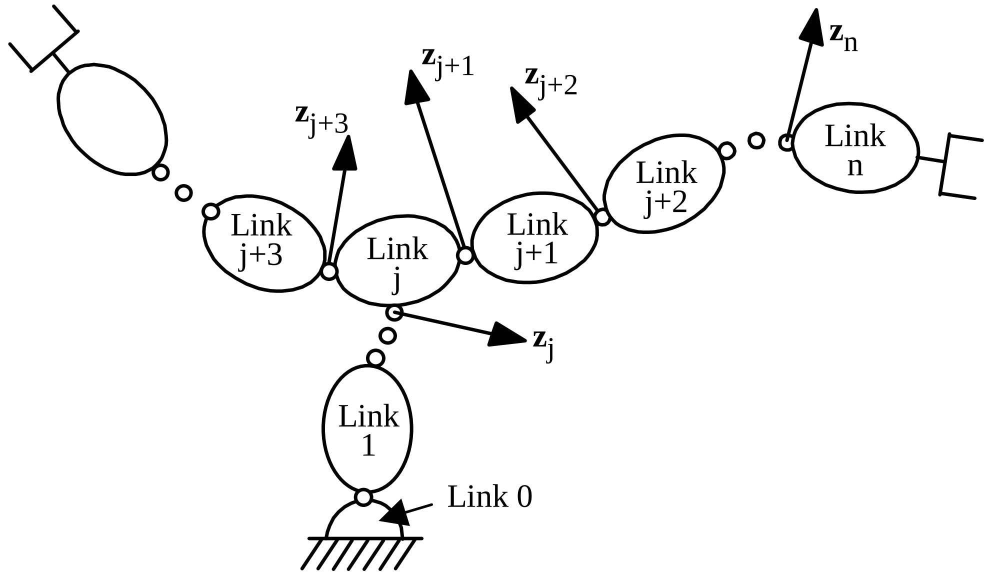
## 2- Description of The Kinematics Of Robots

- The kinematics will be described using the Modified Denavit and Hartenberg Method (Khalil and Kleinfinger, 1986).
- Different advantages wrt classical DH method:

- Extension to tree and closed loop robots,
- Calculation of base inertial parameters using closed loop rules,
- More logical choices,
- No reasons to continue to use classical notations

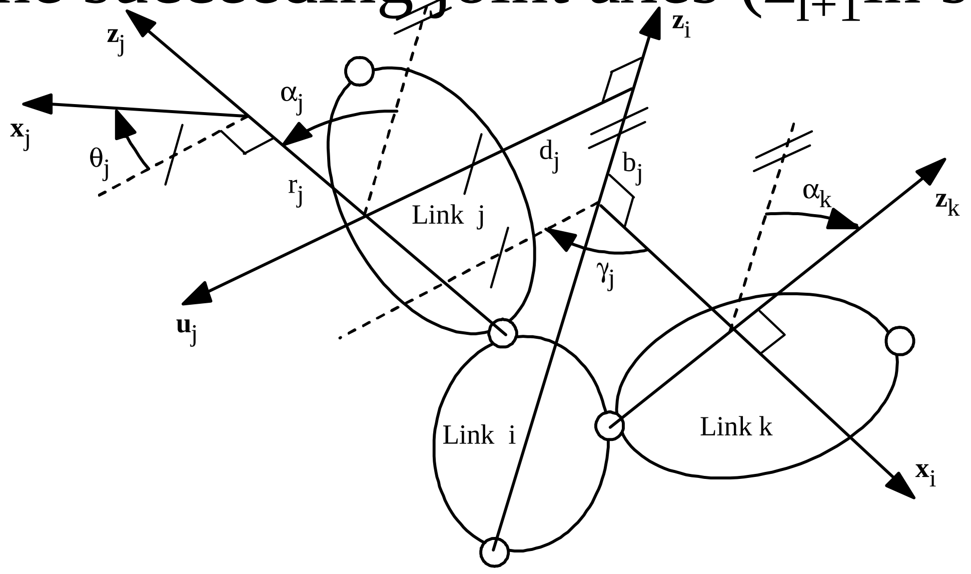
## 2.1 Geometric description of tree structure robots

- n+1 links (link 0 is the bas), n joints (Rot-Prismatic),
- frame j fixed with link j,
- Link j is articulated on joint j,
- a(j) indicates the link antecedent to j, a(j) = j-1 in serial robot.



# Definition of frame i

- $\mathbf{z}_i$ is along the axis of joint i;
- $\mathbf{x}_i$ is taken along the common normal between $\mathbf{z}_i$ and one of the succeeding joint axes ($\mathbf{z}_{i+1}$in serial robots).



${}^{i}\mathbf{T}_j$ = **Rot**(**z**, ♑$_j$) **Tran**(**z**, b$_j$) **Rot**(**x**, ♋$_j$) **Tran**(**x**, d$_j$) **Rot**(**z**, $\theta_j$) **Tran**(**z**, r$_j$)

The serial structure is a special case of a tree structure where a(j) = j-1, $\gamma_j$ = 0, and $b_j$ =0.

## 2.2 Description of closed loop structure

- The system is composed of L joints and n + 1 links,
- The number of independent closed loops is equal to:

  $B = L - n$
- The joints are either active (motorized) or passive.

- To determine the geometric parameters:

  a) Construct an equivalent tree structure having n joints by virtually cutting each closed chain at one of its passive joints.

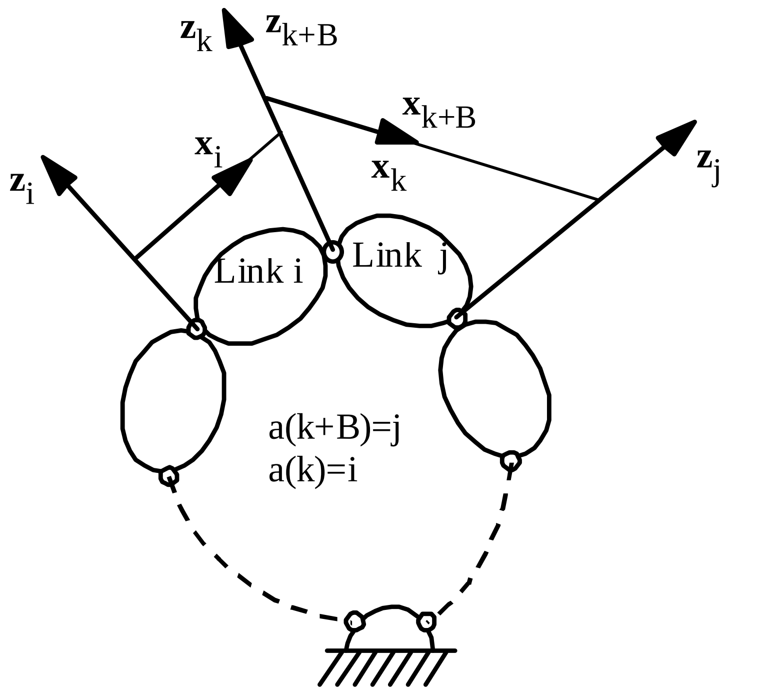
  b) number the cut joint as: $n+1,\ldots,L$

  c) For each cut joint k define two supplementary frames (k and k+B) on one of the links connected by this joint. The transformation matrix defining k wrt a(k) is defined in terms of $q_k$ where as the definition of frame k+B wrt its antecedent is constant.

•

The joint variables are denoted as:

$$\mathbf{q} = \begin{bmatrix} \mathbf{q}_{\mathrm{tr}} \\ \mathbf{q}_{\mathrm{c}} \end{bmatrix}, \mathbf{q}_{\mathrm{tr}} = \begin{bmatrix} \mathbf{q}_{\mathrm{a}} \\ \mathbf{q}_{\mathrm{p}} \end{bmatrix}$$



Frames around a cut joint:

Constraint Equations:

$$^{k+B}\mathbf{T}_j \ldots {}^{i}\mathbf{T}_k = \mathbf{I}_4$$

$$^{0}V_k = {}^{0}V_{k+B}$$

$$\mathbf{J}_k \dot{\mathbf{q}}_{\mathbf{b1}} = \mathbf{J}_{k+B} \dot{\mathbf{q}}_{\mathbf{b2}}$$

$$V_j = \begin{bmatrix} \mathbf{V}_j^{T} & \boldsymbol{\omega}_j^{T} \end{bmatrix}^{T}$$

# 3 Dynamic Modeling of Tree Structure Robots

## Definitions

- **Inverse Dynamic Model (IDyM)** $\boldsymbol{\Gamma} = \mathbf{f}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})$
- **Direct Dynamic Model (DDyM)** $\ddot{\mathbf{q}} = \mathbf{f}(\boldsymbol{\Gamma}, \mathbf{q}, \dot{\mathbf{q}})$
- **Lagrange Equations** $\Gamma = \frac{\mathrm{d}}{\mathrm{dt}}\left[\frac{\partial L}{\partial \dot{\mathbf{q}}}\right]^T - \left[\frac{\partial L}{\partial \mathbf{q}}\right]^T$
- **IDyM** $\boldsymbol{\Gamma} = \mathbf{A}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{H}(\mathbf{q}, \dot{\mathbf{q}})$ , **DDyM** $\ddot{\mathbf{q}} = -(\mathbf{A})^{-1}(\boldsymbol{\Gamma} - \mathbf{H})$

- **Newton-Euler equations:** giving the external forces and moments on a link j about the origin of frame j are written as:

$$ {}^{j}\mathrm{F}_{j} = {}^{j}\mathrm{J}_{j}\,{}^{j}\dot{\mathrm{V}}_{j} + \begin{bmatrix} {}^{j}\boldsymbol{\omega}_{j} \times \left( {}^{j}\boldsymbol{\omega}_{j} \times {}^{j}\mathbf{MS}_{j} \right) \\ {}^{j}\boldsymbol{\omega}_{j} \times \left( {}^{j}\mathbf{J}_{j}\,{}^{j}\boldsymbol{\omega}_{j} \right) \end{bmatrix} $$

$$ {}^{j}\mathrm{F}_{j} = \begin{bmatrix} {}^{j}\mathbf{F}_{j} \\ {}^{j}\mathbf{M}_{j} \end{bmatrix},\; {}^{j}\mathrm{J}_{j} = \begin{bmatrix} \mathrm{M}_{j}\mathbf{I}_{3} & -{}^{j}\mathbf{M\hat{S}}_{j} \\ {}^{j}\mathbf{M\hat{S}}_{j} & {}^{j}\mathbf{J}_{j} \end{bmatrix} $$

- Mj, **MSj** and **Jj** are the mass, the first moments, and the inertia matrix about the origin of link j.

## 3.2 Calculation of the Inverse dynamics using recursive NE algorithm

- The algorithm (Luh, Walker and Paul, 1980) consists of two recursive computations: forward and backward The forward, from link 1 to link n, computes the link velocities and accelerations and the dynamic wrench on each link.
- The backward equations, from link n to the base, provide the reaction wrenches on the links and consequently the joint torques.

- It provides: $\Gamma = \mathbf{NE}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \mathbf{f}_e, \mathbf{m}_e)$

- **The forward equations** (for j=1,…,n):

$$ {}^{j}V_{j} = {}^{j}T_{i}\,{}^{i}V_{i} + \dot{q}_{j}\,{}^{j}\mathfrak{a}_{j} $$

$$ {}^{j}\dot{V}_{j} = {}^{j}T_{i}\,{}^{i}\dot{V}_{i} + {}^{j}\boldsymbol{\gamma}_{j} + \ddot{q}_{j}\,{}^{j}\mathfrak{a}_{j} $$

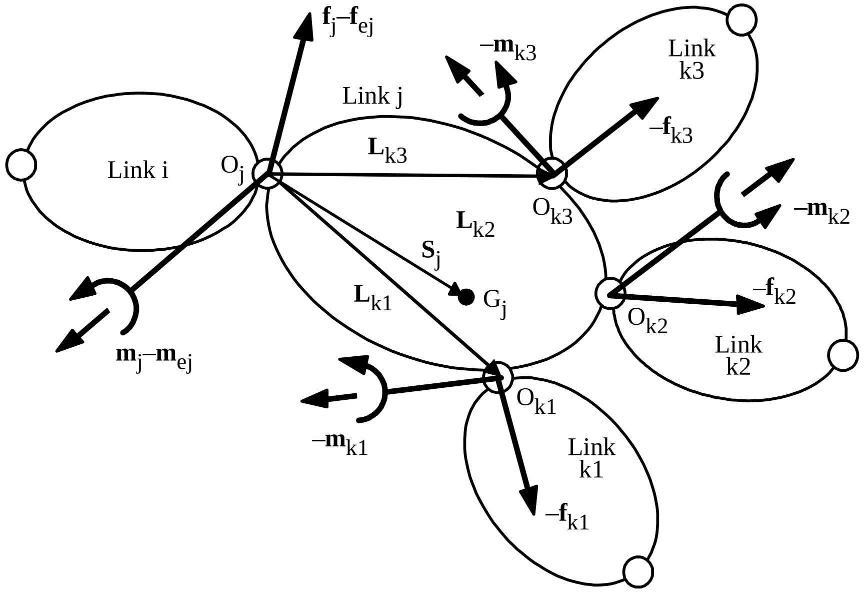
$$ {}^{j}\boldsymbol{\gamma}_{j} = \begin{bmatrix} {}^{j}\mathbf{R}_{i}\left[\,{}^{i}\boldsymbol{\omega}_{i}\times\left({}^{i}\boldsymbol{\omega}_{i}\times{}^{i}\mathbf{P}_{j}\right)\right] + 2\boldsymbol{\sigma}_{\mathbf{j}}\mathbf{(}{}^{j}\boldsymbol{\omega}_{i}\times\dot{q}_{j}\,{}^{j}\mathbf{a}_{j}\mathbf{)} \\ \bar{\boldsymbol{\sigma}}_{\mathbf{j}}\,{}^{j}\boldsymbol{\omega}_{i}\times\dot{q}_{j}\,{}^{j}\mathbf{a}_{j} \end{bmatrix} $$

$$ {}^{j}F_{j} = {}^{j}\mathbf{J}_{j}\,{}^{j}\dot{V}_{j} + \begin{bmatrix} {}^{j}\boldsymbol{\omega}_{j}\times\left({}^{j}\boldsymbol{\omega}_{j}\times{}^{j}\mathbf{MS}_{j}\right) \\ {}^{j}\boldsymbol{\omega}_{j}\times\left({}^{j}\mathbf{J}_{j}\,{}^{j}\boldsymbol{\omega}_{j}\right) \end{bmatrix} $$

Eqs.16- 23 in the proceeding.

## **The backward equations** (for j= n,…,1):

- .



$$ {}^{j}\mathbb{f}_j = {}^{j}F_j + \sum_{k} {}^{k}T_j^{T}\, {}^{k}\mathbb{f}_k + {}^{j}\mathbb{f}_{ej} $$ Where a(k) = j,

$$ \Gamma_j = {}^{j}\mathbb{a}_j^{T}\, {}^{j}\mathbb{f}_j + Ia_j \ddot{q}_j + Fs_j \operatorname{sign}(\dot{q}_j) + Fv_j \dot{q}_j $$

Eqs.26-31 in the proceeding.

## 3.3 Computation of the direct dynamic model

- The DyDM can be calculated from Lagrange model:

$$\ddot{\mathbf{q}} = -\left(\mathbf{A}\right)^{-1}\left(\boldsymbol{\Gamma} - \mathbf{H}\right)$$

- Two methods based on Newton-Euler methods can be used:

  - the first is based on calculating the **A** and **H** matrices using Newton-Euler inverse dynamic model (procedure Walker and Orin)

  - the second method (due to Featherstone) is based on a recursive Newton-Euler algorithm that does not explicitly calculate the matrix **A** and has a computational cost that varies linearly with the number of degrees of freedom of the robot.

- For tree structure robots, the second method is more efficient, but the first method can be used for closed loop robots and some other systems.

## 3.3.1 Using the inverse dynamic model to calculate the direct dynamic model

- By comparing **(Walker and Orin 1982):**

$$\Gamma = \mathrm{NE}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \mathbf{f}_e, \mathbf{m}_e)\ ,\ \Gamma = \mathbf{A}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{H}(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{f}_e, \mathbf{m}_e)$$

**i)** $\mathbf{H}(\mathbf{q}, \dot{\mathbf{q}})$ is equal to $\Gamma$ if $\ddot{\mathbf{q}} = \mathbf{0}$,

ii) the ith column of **A** is equal to $\Gamma$ if:

$$\ddot{\mathbf{q}} = \mathbf{u}_i, = \mathbf{0},\ \dot{\mathbf{q}} = \mathbf{0}\ ,\ \mathbf{g} = \mathbf{0},\ \mathbf{fe} = \mathbf{0},\ \mathbf{me} = \mathbf{0}$$

where $\mathbf{u}_i$ is the $(n\times1)$ unit vector whose ith element is equal to 1, and the other elements are zeros.

## 3.3.2 Recursive NE computation of the direct dynamic model

Three recursive calculations:

i- Forward: As the first recursive of IDyM, when joint acc =0

$$ {}^{j}\boldsymbol{\gamma}_{j} = \begin{bmatrix} {}^{j}\mathbf{R}_{i}\left[ {}^{i}\boldsymbol{\omega}_{i} \times \left( {}^{i}\boldsymbol{\omega}_{i} \times {}^{i}\mathbf{P}_{j} \right) \right] + 2\sigma_{j}\left( {}^{j}\boldsymbol{\omega}_{i} \times \dot{q}_{j}\, {}^{j}\mathbf{a}_{j} \right) \\ \bar{\sigma}_{j}\, {}^{j}\boldsymbol{\omega}_{i} \times \dot{q}_{j}\, {}^{j}\mathbf{a}_{j} \end{bmatrix} $$

$$ {}^{j}\boldsymbol{\beta}_{j} = -\,{}^{j}\mathbf{f}_{ej} - \begin{bmatrix} {}^{j}\boldsymbol{\omega}_{j} \times \left( {}^{j}\boldsymbol{\omega}_{j} \times {}^{j}\mathbf{MS}_{j} \right) \\ {}^{j}\boldsymbol{\omega}_{j} \times \left( {}^{j}\mathbf{J}_{j}\, {}^{j}\boldsymbol{\omega}_{j} \right) \end{bmatrix} $$

**ii- Backward: Calculation of some inertial matrices, giving joint acc. and forces on link j in terms of the link accelerations of link a(j).**

.

$$ \ddot{q}_{j} = H_{j}^{-1}\left[ -\,{}^{j}\mathbf{a}_{j}^{T}\, {}^{j}\mathbb{J}_{j}^{*} \left( {}^{j}\mathbb{T}_{i}\, {}^{i}\dot{\mathbb{V}}_{i} + {}^{j}\gamma_{j} \right) + \tau_{j} + {}^{j}\mathbf{a}_{j}^{T}\, {}^{j}\beta_{j}^{*} \right] \qquad (41) $$

$$ {}^{j}\mathbb{F}_{j} = \begin{bmatrix} {}^{j}\mathbf{f}_{j} \\ {}^{j}\mathbf{m}_{j} \end{bmatrix} = {}^{j}\mathbb{K}_{j}\, {}^{j}\mathbb{T}_{i}\, {}^{i}\dot{\mathbb{V}}_{i} + {}^{j}\alpha_{j} \qquad (42) $$

**iii- Forward: since the acceleration of link 0 is known “-g”, then we can calculate the link acceleration for links 1,…, n, and the corresponding joint accelerations.**

## Second recursive equations:

For j = n ... 1, compute:

$$H_j = ({}^j a_j^T \, {}^j J_j^* \, {}^j a_j + Ia_j) \qquad (36)$$

$${}^j K_j = {}^j J_j^* - {}^j J_j^* \, {}^j a_j \, H_j^{-1} \, {}^j a_j^T \, {}^j J_j^* \qquad (37)$$

$${}^j \alpha_j = {}^j K_j \, {}^j \gamma_j + {}^j J_j^* \, {}^j a_j \, H_j^{-1} (\tau_j + {}^j a_j^T \, {}^j \beta_j^*) - {}^j \beta_j^* \qquad (38)$$

If $a(j) \neq 0$, calculate also:

$${}^i \beta_i^* = {}^i \beta_i^* - {}^j T_i^T \, {}^j \alpha_j \qquad (39)$$

$${}^i J_i^* = {}^i J_i^* + {}^j T_i^T \, {}^j K_j \, {}^j T_i \qquad (40)$$

These equations are initialized by

${}^j J_j^* = {}^j J_j$ and ${}^j \beta_j^* = {}^j \beta_j$.

# 4 Inverse Dynamic Modeling of Closed Loop Robots

- To calculate the Inverse dynamic model of closed loop robots:

i) calculate the inverse dynamic model of the equivalent tree structure robot, in which the joint variables satisfy the constraints of the loop.

ii) closed loop torques of the active joints ☝c are obtained by projecting the tree structure torques ☝$_{tr}$ on the motorized joints using the transpose of the Jacobian matrix of the tree structure variables (or velocities) in terms of the active joint variables (or velocities).

$$☝c = \mathbf{G}^T ☝_{tr}$$

where:

$$\mathbf{G} = \frac{\partial \mathbf{q}_{tr}}{\partial \mathbf{q}_a} = \frac{\partial \dot{\mathbf{q}}_{tr}}{\partial \dot{\mathbf{q}}_a}$$

$$\mathbf{\Gamma}_c = \mathbf{\Gamma}_a + \frac{\partial \dot{\mathbf{q}}_p}{\partial \dot{\mathbf{q}}_a} \mathbf{\Gamma}_p$$

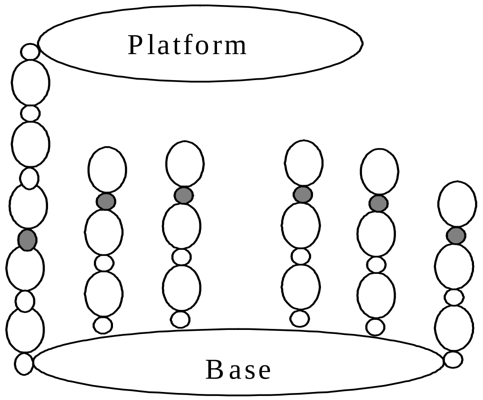
## Direct dynamic model of closed loop robots:

- There is no recursive method to obtain it. It can be computed using the inverse dynamic model using (Walker and Orin) procedure in order to obtain the matrices $\mathbf{A}_c$ and $\mathbf{H}_c$ of the following relation:

$$\Gamma_c = \mathbf{A}_c(\mathbf{q}_{tr})\ddot{\mathbf{q}}_a + \mathbf{H}_c(\mathbf{q}_{tr}, \dot{\mathbf{q}}_{tr})$$

## 5. Inverse Dynamic Modeling of Parallel Robots

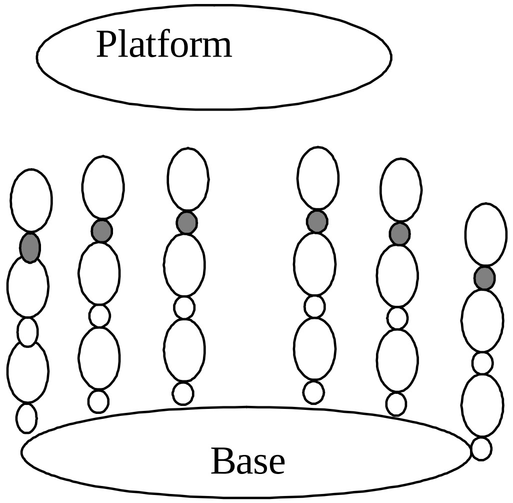
- The robot is composed of a fixed base and a mobile platform. They are connected by m parallel legs. It is a system with multiple closed loops.
- The Equivalent Tree structure (platform attached to a leg)



The dynamic model will $\mathbf{F}_{\mathbf{P}}$ be obtained in terms of the joint variales. The joints connection the platform to the leg must be obtained too.

## Second solution

- The system is decomposed into 2 subsystems: the platfor and the legs:



The platform dynamics $\mathbf{F_P}$ is calculated as a function of the Cartesian variables, whereas the dynamics of the legs ☝$_i$ are calculated as a function of the joint variables of the legs.

- $\mathbf{F_P}$ is calculated by the Newton-Euler equation of the platform, $\Gamma_i$ is the inverse dynamic model of leg i.

- Thus the dynamic model of the parallel structure is given by the following equation:

$$\mathbf{\Gamma = J_P^T F_P} + \sum_{i=1}^{m} \left( \frac{\partial \dot{\mathbf{q}}_i}{\partial \dot{\mathbf{q}}_a} \right)^{T} \Gamma_i$$

- $\mathbf{J_P}$ is the $(6 \times n)$ kinematic Jacobian matrix of the robot,

- $\dfrac{\partial \dot{\mathbf{q}}_i}{\partial \dot{\mathbf{q}}_a}$ is calculated by the following relation, which exploits the parallel structure of the robot:

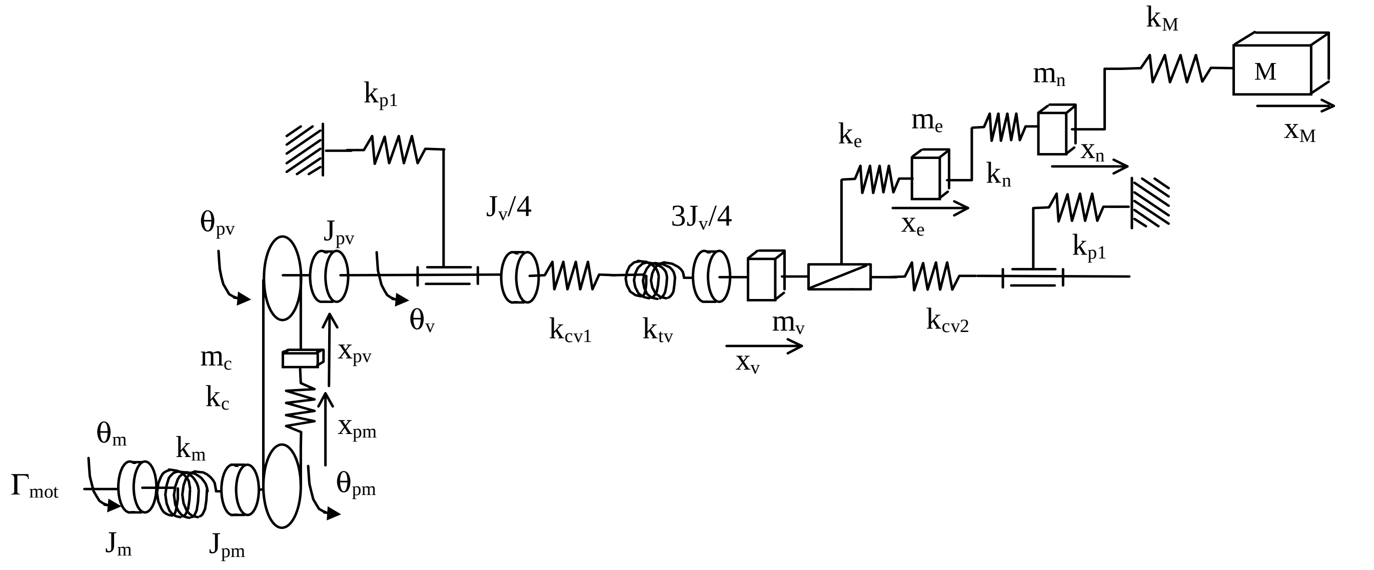
$$\frac{\partial \dot{\mathbf{q}}_i}{\partial \dot{\mathbf{q}}_a} = \frac{\partial \dot{\mathbf{q}}_i}{\partial \mathbf{v}_i} \frac{\partial \mathbf{v}_i}{\partial V_P} \frac{\partial V_P}{\partial \dot{\mathbf{q}}_a} = \mathbf{J_i^{-1}}\, \mathbf{J_{vi}}\, \mathbf{J}_p$$

- The inverse dynamic model $$\mathbf{\Gamma = J_p^T} \left[ F_P + \sum_{i=1}^{m} \mathbf{J_{vi}^T}\, \mathbf{J_i^{-T}} \mathbf{\Gamma_i} \right]$$

## 6. Inverse Dynamic Modeling Of Robots With Elastic Joints

- The description of the system can be done by **MDH**, just note that the joint torque will be ☝$_j$ = - 🖓$q_j$ $K_j$

➓ 🖓$q_i$ = $q_i - q_{0i}$ , $K_i$ is the joint stiffness.



- **The general form of the dynamic model is written as:**

$$\Gamma = \begin{bmatrix} \Gamma_r \\ \Gamma_f \end{bmatrix} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{12}^{\mathrm{T}} & \mathbf{A}_{22} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{q}}_r \\ \ddot{\mathbf{q}}_f \end{bmatrix} + \begin{bmatrix} \mathbf{H}_r \\ \mathbf{H}_f \end{bmatrix}$$

- Where r means rigid and f means flexible.
- The direct dynamic model **DDyM** can be obtained from this model by inverting the inertia matrix **A**. The recursive algorithm of rigid bodies can be used after taking into account the torques of flexible joints.

- To solve the IDyM from this general model we cannot specify the elastic accelerations. They must be calculated using the second row of the previous equation.

$$\Gamma_f = \begin{bmatrix} \mathbf{A}_{12}^{\mathrm{T}} & \mathbf{A}_{22} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{q}}_r \\ \ddot{\mathbf{q}}_f \end{bmatrix} + \mathbf{H}_f$$

Then the joint torques are calculated from the first row.

The Newton-Euler method can be adapted with three recursive steps.

## Inverse dynamics of systems with flexible joints

This algorithm consists of three recursive steps.

i) The first forward iteration is exactly the same as that of the direct dynamic model of rigid system: calculation of elements in terms of q and $\dot{\mathbf{q}}$

ii) The second backward recursive equations calculate the matrices giving the elastic accelerations and ${}^{j}\mathbf{f}_{j}$ as a function of the acceleration of link a(j). They can be calculated for j =n, ...,1, as follows:

## Second (backward) recursive equations:

- If joint j is elastic:

$$H_j = {}^j\mathbb{a}_j^T \, {}^j\mathbb{J}_j^{*} \, {}^j\mathbb{a}_j \qquad (72)$$

$${}^j\mathbb{K}_j = {}^j\mathbb{J}_j^{*} - {}^j\mathbb{J}_j^{*} \, {}^j\mathbb{a}_j \, H_j^{-1} \, {}^j\mathbb{a}_j^T \, {}^j\mathbb{J}_j^{*} \qquad (73)$$

$${}^j\alpha_j = {}^j\mathbb{K}_j \, {}^j\gamma_j + {}^j\mathbb{J}_j^{*} \, {}^j\mathbb{a}_j \, H_j^{-1} (-K_j \, \Delta q_j + {}^j\mathbb{a}_j^T \, {}^j\beta_j^{*}) - {}^j\beta_j^{*} \qquad (74)$$

- If joint j is rigid:

$${}^j\mathbb{K}_j = {}^j\mathbb{J}_j^{*} \qquad (75)$$

$${}^j\alpha_j = {}^j\mathbb{K}_j \, {}^j\gamma_j + {}^j\mathbb{J}_j^{*} \, {}^j\mathbb{a}_j \, \ddot{q}_j - {}^j\beta_j^{*} \qquad (76)$$

if $a(j) \neq 0$, calculate:

$${}^i\beta_i^{*} = {}^i\beta_i^{*} - {}^j\mathbb{T}_i^T \, {}^j\alpha_j \qquad (77)$$

$${}^i\mathbb{J}_i^{*} - {}^i\mathbb{J}_i^{*} + {}^j\mathbb{T}_i^T \, {}^j\mathbb{K}_j \, {}^j\mathbb{T}_i \qquad (78)$$

- The third recursive step calculates (for j = 1, ..., n) the acceleration of elastic joints and the joint torques for the rigid joints using the following equations:

$$ {}^{j}\mathbb{F}_j = \begin{bmatrix} {}^{j}\mathbf{f}_j \\ {}^{j}\mathbf{m}_j \end{bmatrix} = {}^{j}\mathbb{K}_j \, {}^{j}\mathbb{T}_i \, {}^{i}\dot{\mathbb{V}}_i + {}^{j}\alpha_j \qquad (79) $$

- if j is elastic:

$$ \ddot{q}_j = H_j^{-1} [ - {}^{j}\mathbf{a}_j^{T} \, {}^{j}\mathbb{J}_j^{*} \, ({}^{j}\mathbb{T}_i \, {}^{i}\dot{\mathbb{V}}_i + {}^{j}\gamma_j) - K_j \, \Delta q_j + {}^{j}\mathbf{a}_j^{T} \, {}^{j}\beta_j^{*} ] \qquad (80) $$

$$ {}^{j}\dot{\mathbb{V}}_j = {}^{j}\mathbb{T}_i \, {}^{i}\dot{\mathbb{V}}_i + {}^{j}\mathbf{a}_j \, \ddot{q}_j + {}^{j}\gamma_j \qquad (81) $$

- if j is rigid

$$ \Gamma_j = (\sigma_j \, {}^{j}\mathbf{f}_j + \bar{\sigma}_j \, {}^{j}\mathbf{m}_j)^{T} \, {}^{j}\mathbf{a}_j + Ia_j \, \ddot{q}_j \qquad (82) $$

## 7. Dynamic Modeling of Robots With Moving Base

- Examples: cars, mobile robots, mobile manipulators, walking robots, Humanoid robots, eel like robots, snakes like robots, flying robots, spatial vehicle, etc.
- The difference between these systems will be in the calculation of the interaction forces with the environment.
- In the previous sections the base is fixed thus the acceleration of the base is equal to zero, whereas in the case of a mobile base system the acceleration of the base must be determined in both direct and inverse dynamic models.

## • 7.1 General form of the dynamic models

$$\begin{bmatrix} \mathbf{0}_{6x1} \\ \Gamma \end{bmatrix} = \mathbf{A} \begin{bmatrix} {}^0\dot{V}_0 \\ \ddot{\mathbf{q}} \end{bmatrix} + \mathbf{H}$$

- Where A is the inertia matrix of the robot, H contains the Centrifugal, Coriolis gravity and contact forces,

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{12}^{T} & \mathbf{A}_{22} \end{bmatrix}, \mathbf{H} = \begin{bmatrix} \mathbf{H}_1 \\ \mathbf{H}_2 \end{bmatrix}$$

- The inverse dynamic model is solved as follows:

$$ {}^0\dot{V}_0 = -\left(\mathbf{A}_{11}\right)^{-1}\left(\mathbf{H}_1 + \mathbf{A}_{12}\ddot{\mathbf{q}}\right), \text{then } \Gamma = \mathbf{A}_{12}^{T}\,\dot{V}_0 + \mathbf{A}_{22}\,\ddot{\mathbf{q}} + \mathbf{H}_2$$

- The direct dynamic model is solved as follows:

$$\begin{bmatrix} \dot{V}_0 \\ \ddot{\mathbf{q}} \end{bmatrix} = \mathbf{A}^{-1} \begin{bmatrix} -\mathbf{H}_1 \\ \Gamma - \mathbf{H}_2 \end{bmatrix}$$

- The calculation of **A** and **H** can be done by Lagrange method. They can also be calculated using the inverse dynamic model of tree structure and using the procedure of (Walker and Orin). The base can be taken into account by either of the following methods:

  - The velocity and acceleration of the base will be the initial conditions for the forward recursive calculation. The backward recursive calculation must continue to j=0, where this new iteration will provide the 6 equations of Newton-Euler equations of the base.

  - We suppose link 0 is a virtual link whose inertial parameters are equal to zero but has the velocity and acceleration of the base which is taken as Link 1. The six equations of the base will be those of $f_1 = 0$

## 7.2 Recursive NE calculation of the inverse dynamic model of robots with mobile base

- The inverse dynamic algorithm consists of three recursive equations (a forward, then a backward, then a forward).

- *i) Forward recursive calculation (*for j=1,…,n *):*

  In this step we calculate the screw transformation matrices, link velocities, and the elements of the accelerations and external wrenches on the links, which are independent of the acceleration of the robot base.

- .

$$ {}^{j}T_{i} = \begin{bmatrix} {}^{j}\mathbf{R}_{i} & -{}^{j}\mathbf{R}_{i}\,{}^{i}\hat{\mathbf{P}}_{j} \\ \mathbf{0}_{3x3} & {}^{j}\mathbf{R}_{i} \end{bmatrix} $$

$$ {}^{j}V_{j} = {}^{j}T_{i}\,{}^{i}V_{i} + \dot{q}_{j}\,{}^{j}a_{j} $$

$$ {}^{j}\dot{V}_{j} = {}^{j}T_{i}\,{}^{i}\dot{V}_{i} + {}^{j}\gamma_{j} + \ddot{q}_{j}\,{}^{j}a_{j} $$

$$ {}^{j}\boldsymbol{\gamma}_{j} = \begin{bmatrix} {}^{j}\mathbf{R}_{i}\left[{}^{i}\boldsymbol{\omega}_{i} \times \left({}^{i}\boldsymbol{\omega}_{i} \times {}^{i}\mathbf{P}_{j}\right)\right] + 2\sigma_{j}\,({}^{j}\boldsymbol{\omega}_{i} \times \dot{q}_{j}\,{}^{j}\mathbf{a}_{j}) \\ \bar{\sigma}_{j}\,{}^{j}\boldsymbol{\omega}_{i} \times \dot{q}_{j}\,{}^{j}\mathbf{a}_{j} \end{bmatrix} $$

$$ {}^{j}\boldsymbol{\zeta}_{j} = {}^{j}\boldsymbol{\gamma}_{j} + \ddot{q}_{j}\,{}^{j}\mathbf{a}_{j} $$

$$ {}^{j}\boldsymbol{\beta}_{j} = -{}^{j}f_{ej} - \begin{bmatrix} {}^{j}\boldsymbol{\omega}_{j} \times \left({}^{j}\boldsymbol{\omega}_{j} \times {}^{j}\mathbf{MS}_{j}\right) \\ {}^{j}\boldsymbol{\omega}_{j} \times \left({}^{j}\mathbf{J}_{j}\,{}^{j}\boldsymbol{\omega}_{j}\right) \end{bmatrix} $$

## *ii) Backward recursive equations:*

- In this step we obtain the base acceleration using the inertial parameters of the composite link 0, where the composite link 0 consists of all the links articulated on link 0.

$$ {}^{j}\mathbb{f}_{j} = {}^{j}\mathbf{J}^{c}_{j}\,{}^{j}\dot{\mathrm{V}}_{j} - {}^{j}\boldsymbol{\beta}^{c}_{j} $$

$$ {}^{j}\mathbf{J}^{c}_{j} = {}^{j}\mathbf{J}^{c}_{j} + \sum_{k} {}^{k}\mathrm{T}^{T}_{j}\,{}^{k}\mathbf{J}^{c}_{k}\,{}^{k}\mathrm{T}_{j} $$

$$ {}^{j}\boldsymbol{\beta}^{c}_{j} = {}^{j}\boldsymbol{\beta}^{c}_{j} - \sum_{k} {}^{k}\mathrm{T}^{T}_{j}\,{}^{k}\boldsymbol{\beta}^{c}_{k} + {}^{k}\mathrm{T}^{T}_{j}\,{}^{k}\mathbf{J}^{c}_{k}\,{}^{k}\boldsymbol{\zeta}_{k} $$

$$ {}^{0}\dot{\mathrm{V}}_{0} = \left({}^{0}\mathbf{J}^{c}_{0}\right)^{-1}\,{}^{0}\boldsymbol{\beta}^{c}_{0} $$

- *iii) Forward recursive equations (j= 1,…, n )*

After calculating the acceleration of the base, the wrench and the joint torques are obtained as:

$$ {}^{j}\dot{\mathrm{V}}_{j} = {}^{j}\mathrm{T}_{i}\, {}^{i}\dot{\mathrm{V}}_{i} + {}^{j}\boldsymbol{\zeta}_{j} $$

$$ {}^{j}\mathbb{f}_{j} = \begin{bmatrix} {}^{j}\mathbf{f}_{j} \\ {}^{j}\mathbf{m}_{j} \end{bmatrix} = {}^{j}\mathbf{J}_{j}^{\mathbf{c}}\, {}^{j}\dot{\mathrm{V}}_{j} - {}^{j}\boldsymbol{\beta}_{j}^{\mathbf{c}} $$

$$ \boldsymbol{\Gamma}_{\mathbf{j}} = {}^{j}\mathbb{f}_{j}^{\mathbf{T}}\, {}^{j}\mathbb{a}_{j} + \mathbf{F_{sj}}\, \mathbf{sign}\left(\dot{\mathbf{q}}_{\mathbf{j}}\right) + \mathbf{F_{vj}}\, \dot{\mathbf{q}}_{\mathbf{j}} + \mathbf{I_{aj}}\, \ddot{\mathbf{q}}_{\mathbf{j}} $$

It is to be noted that the inverse dynamic model can be used in the dynamic simulation of the mobile robot when the objective is to study the effect of the joint motions on the base. In this case the joint positions, velocities and accelerations trajectories are given.

## 7.3 Recursive direct Dynamic model

- The direct dynamic model consists of three recursive calculations (forward, backward and forward):

*i) Forward recursive equations:*

We calculate the link Cartesian velocities and the terms of Cartesian accelerations and equilibrium equations of the links that are independent of the accelerations of the base and of the joints.

- .

$$ {}^{j}\boldsymbol{\gamma}_{j} = \begin{bmatrix} {}^{j}\mathbf{R}_{i}\left[ {}^{i}\boldsymbol{\omega}_{i} \times \left( {}^{i}\boldsymbol{\omega}_{i} \times {}^{i}\mathbf{P}_{j} \right) \right] + 2\sigma_{j}\,( {}^{j}\boldsymbol{\omega}_{i} \times \dot{q}_{j}\; {}^{j}\mathbf{a}_{j}) \\ \bar{\sigma}_{j}\; {}^{j}\boldsymbol{\omega}_{i} \times \dot{q}_{j}\; {}^{j}\mathbf{a}_{j} \end{bmatrix} $$

$$ {}^{j}\boldsymbol{\beta}_{j} = -\,{}^{j}\mathbb{f}_{ej} - \begin{bmatrix} {}^{j}\boldsymbol{\omega}_{j} \times \left( {}^{j}\boldsymbol{\omega}_{j} \times {}^{j}\mathbf{MS}_{j} \right) \\ {}^{j}\boldsymbol{\omega}_{j} \times \left( {}^{j}\mathbf{J}_{j}\; {}^{j}\boldsymbol{\omega}_{j} \right) \end{bmatrix} $$

*ii) Backward recursive equations:*

In this second step, we first initialise ${}^{n}\mathbb{J}_{n}^{*} = {}^{n}\mathbb{J}_{n}$, ${}^{n}\boldsymbol{\beta}_{n}^{*} = {}^{n}\boldsymbol{\beta}_{n}$ and then we calculate for j = n,…,1 the following elements, which permit to calculate ${}^{j}f_{j}$ and $\ddot{q}_{j}$ in terms of ${}^{i}\dot{\mathbb{V}}_{i}$ and will be used in the third recursive equations (these matrices can be obtained using a similar procedure as for the direct dynamic model of rigid links):

$$H_j = {}^{j}a_j^{T}\, {}^{j}\mathbb{J}_j^{*}\, {}^{j}a_j + Ia_j \qquad (103)$$

$$ {}^{j}\mathbb{K}_j = {}^{j}\mathbb{J}_j^{*} - {}^{j}\mathbb{J}_j^{*}\, {}^{j}a_j H_j^{-1}\, {}^{j}a_j^{T}\, {}^{j}\mathbb{J}_j^{*} \qquad (104)$$

$$ {}^{i}\mathbb{J}_i^{*} = {}^{i}\mathbb{J}_i + {}^{j}\mathbb{T}_i^{T}\, {}^{j}\mathbb{K}_j\, {}^{j}\mathbb{T}_i \qquad (105)$$

$$\tau_j - \Gamma_j - F_{sj}\,\mathrm{sign}(\dot{q}_j) - F_{vj}\,\dot{q}_j \qquad (106)$$

$$ {}^{j}\boldsymbol{\alpha}_j = {}^{j}\mathbb{K}_j\, {}^{j}\boldsymbol{\gamma}_j + {}^{j}\mathbb{J}_j^{*}\, {}^{j}a_j H_j^{-1}\left(\tau_j + {}^{j}a_j^{T}\, {}^{j}\boldsymbol{\beta}_j^{*}\right) - {}^{j}\boldsymbol{\beta}_j^{*} \qquad (107)$$

$$ {}^{i}\boldsymbol{\beta}_i^{*} = {}^{i}\boldsymbol{\beta}_i - {}^{j}\mathbb{T}_i^{T}\, {}^{j}\boldsymbol{\alpha}_j \qquad (108)$$

- *iii) Forward recursive equations:*
- At first, the base acceleration is calculated by the following relation:

$$ {}^{0}\dot{V}_{0} = \left( {}^{0}J_{0}^{*} \right)^{-1} \, {}^{0}\boldsymbol{\beta}_{0}^{*} $$

$$ \ddot{q}_{j} = H_{j}^{-1} \left[ -{}^{j}\mathbb{a}_{j}^{T} \, {}^{j}J_{j}^{*} \left( {}^{j}T_{i} \, {}^{i}\dot{V}_{i} + {}^{j}\boldsymbol{\gamma}_{j} \right) + \boldsymbol{\tau}_{j} + {}^{j}\mathbb{a}_{j}^{T} \, {}^{j}\boldsymbol{\beta}_{j}^{*} \right] $$

$$ {}^{j}\mathbb{f}_{j} = {}^{j}K_{j} \, {}^{j}T_{i} \, {}^{i}\dot{V}_{i} + {}^{j}\alpha_{j} $$

$$ {}^{j}\dot{V}_{j} = {}^{j}T_{i} \, {}^{i}\dot{V}_{i} + {}^{j}\mathbb{a}_{j}\ddot{q}_{j} + {}^{j}\boldsymbol{\gamma}_{j} $$

## Conclusion

- Other systems modeled with the same techniques:

  - Flexible links robots,

  - Non holonomic robots,

  - micro-continu robots,

  - Hybrid structure, where the robot is composed

  parallel modules, which are connected in serie.