Material for lecture 3 –
Special methods for dynamics in robotics - inverse dynamic problem, special methods for direct problem solution (recursive methods)

Prerequisites:

- inverse kinematics of mechanisms
- dynamics of mechanisms using Newton-Euler equations
- dynamics of mechanisms using Lagrange equations

Literature: V. Stejskal, M. Valášek: Kinematics and Dynamics of Machinery, Marcel Dekker 1996, New York.

Inverse dynamic problem

Based on Lagrange's equations of mixed type

$$\underline{M(s)}\,\ddot{\underline{s}} = \underline{\Phi(s)}^{T}\,\underline{\lambda} + \underbrace{\underline{\nu}(\underline{s},\dot{\underline{s}})}_{\underline{T(s)}\,\underline{m_D} + \underline{q}(\underline{s},\dot{\underline{s}})}$$

drives-couples and forces

Inverse d. problem: requested motion, unknown couples and forces in drives + Lagrange's multipliers

Inverse dynamic problem

$$\begin{bmatrix} \underline{\Phi}_{(s)}^{T} & \underline{T}_{(s)} \end{bmatrix} \begin{bmatrix} \underline{\lambda} \\ \underline{m_p} \end{bmatrix} = -\underline{M}_{(s)}\,\ddot{\underline{s}} + \underline{g}_{(s,\dot{s})}$$

Basic situation: μ L. multipliers

$d = m$ driving forces and couples

$$\mu + m = m$$

Redundantly actuated system (overactuated)

$d > m$ ... more drives than DOF

Underactuated system: $d < n$

Evaluation of requested motion of robot chain

- Based on requested relative motions in drives
- Based on requested end-effector motion

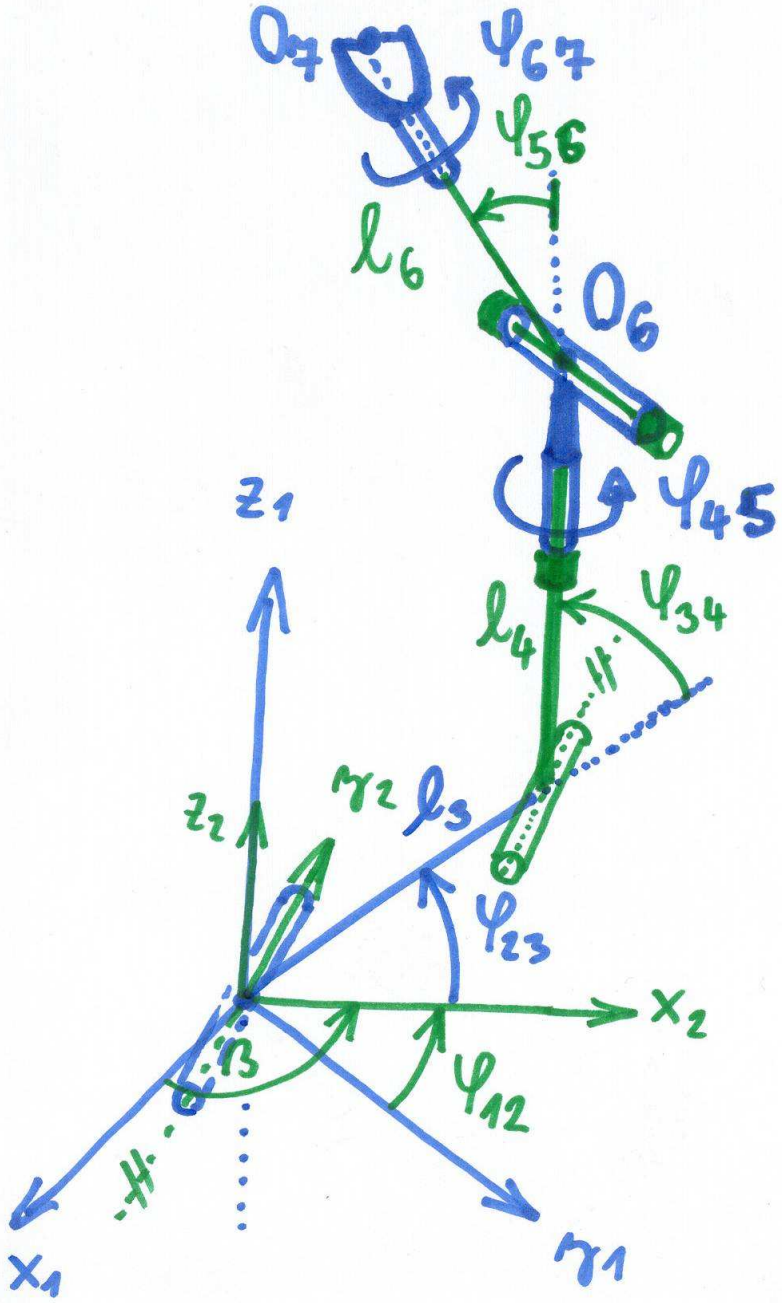
↓

Needs inverse kinematics!

Inverse kinematics for positions – nonlinear alg. eq.

— " — for velocities – Linear alg. eq.

— " — for accelerations – Linear alg. eq.



Inverse kinematics - example

Given: End-effector motion T_{17}

dimensions: l_3, l_4, l_6

Goal: Relative motions in joints + Matlab implementations

```
function [uhly] = Inversni_kin_uloha_robot_MME(T17,l3,l4,l6,konfi1,konfi2)
 % vypocet bodu O6
 j7=T17(1:3,2);
 u07=T17(1:3,4);
 u06=u07-l6*j7;
 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 % vypocet uhlu fi12
 x106=u06(1);
 y106=u06(2);
 % help atan2
 beta=atan2(y106,x106);
 fi12=beta-pi/2;
 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 % vypocet uhlu fi34
 z206=u06(3); % z206=z106
 y206=-sin(fi12)*x106+cos(fi12)*y106;
 cosfi34=(y206^2+z206^2-l3^2-l4^2)/(2*l3*l4);
 % konfi1=1 vybere konfiguraci fi34 mezi 0 a pi
 % konfi1=2 vybere konfiguraci fi34 mezi pi a 2*pi
 fi34=acos(cosfi34);
 if konfi1==1
   fi34=fi34;
 elseif konfi1==2
    fi34=2*pi-fi34;
```

```
else
    % nejasná volba
 end
 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 % vypocet uhlu fi23
 % reseni soustavy linearnich rovnic pro sin(fi23) a cos(fi23)
 Matice_soustavy=[-l4*sin(fi34) l3+l4*cos(fi34)
              l3+l4*cos(fi34) l4*sin(fi34)];
 Vektor_pravych_stran=[y206
                  z206];
 Vysledek= Matice_soustavy\Vektor_pravych_stran;
 sinfi23=Vysledek(1);
 cosfi23=Vysledek(2);
 fi23=atan2(sinfi23,cosfi23);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Vypocet matice smerovych kosinu S47
S12=[cos(fi12)  -sin(fi12)    0
    sin(fi12)   cos(fi12)    0
      0           0         1];

S23=[  1      0          0
      0    cos(fi23)  -sin(fi23)
      0    sin(fi23)   cos(fi23)];
```

```
S34=[  1      0           0
       0    cos(fi34)  -sin(fi34)
       0    sin(fi34)   cos(fi34)];
S17=T17(1:3,1:3);
 S47=S34'*S23'*S12'*S17;
 % vypocet uhlu fi56
  fi56=acos(S47(2,2));
 if konfi2==1
    fi56=fi56;
 elseif konfi2==2
     fi56=2*pi-fi56;
 else
     % nejasná volba
 end
 % vypocet uhlu fi67
 cosfi67=S47(2,3)/(-sin(fi56));
 sinfi67=S47(2,1)/(sin(fi56));
 fi67=atan2(sinfi67,cosfi67);
  % vypocet uhlu fi45
 cosfi45=S47(3,2)/(sin(fi56));
 sinfi45=S47(1,2)/(sin(fi56));
 fi45=atan2(sinfi45,cosfi45);
 uhly=[fi12;fi23;fi34;fi45;fi56;fi67];
```

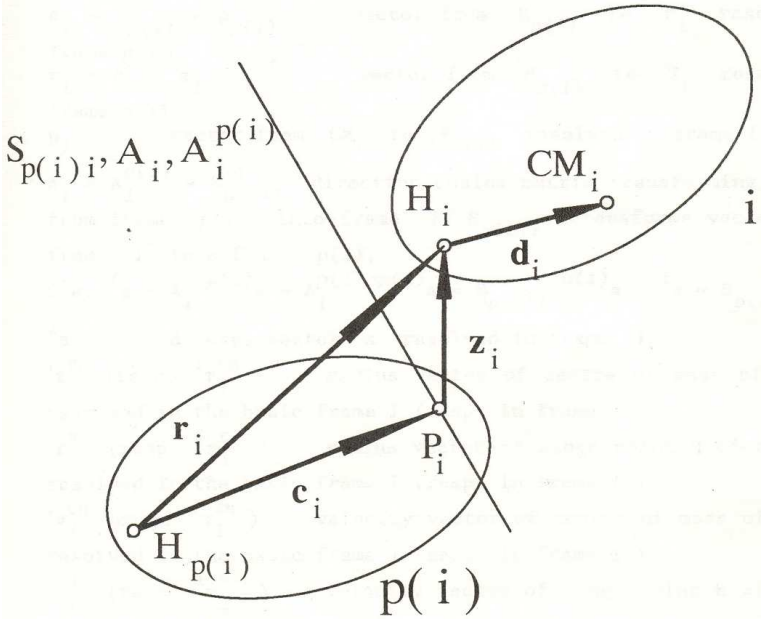
Recursive dynamic method

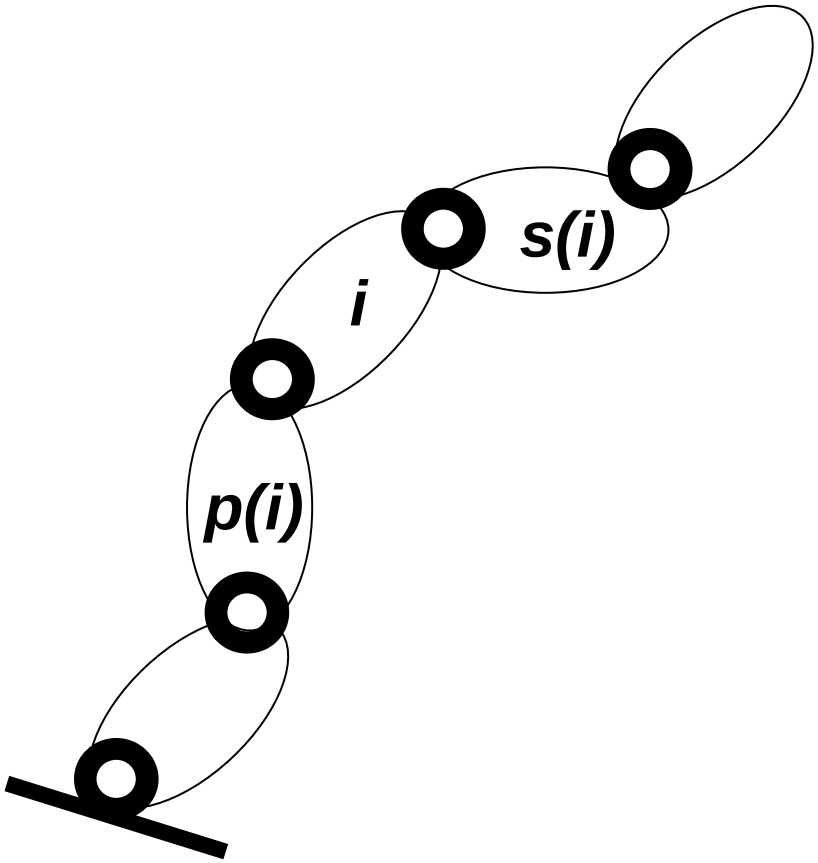
(for exam not necessary in detail)

The special method of efficient solution of direct dynamic problem with O(n) computational complexity. The method known in different variants.
The primary target is the solution of the open robotic chain.

s(i) ... index succeeding body in chain

p(i) ... index of previous body in chain





Newton-Euler equations for body

Dynamic equations:

$$I_i^H \; a_i^H = f_i - C_{s(i)}^T \; f_{s(i)} + f_{Ei}^H + \beta_i^H$$

Vector of acting forces to i-th body:

$$f_{Ei}^H = f_{Ai}^H + \Phi_i \; \lambda_i - C_{s(i)}^T \; \Phi_{s(i)} \; \lambda_{s(i)}$$

Reaction vector:

$$f_i = [\tau_{ix}, \tau_{iy}, 0, f_{ix}, f_{iy}, f_{iz}]^T$$

Composed transformation matrix from i-th coord. system to s(i)-th :

$$C_{s(i)} = \begin{bmatrix} A_{s(i)}^i & 0 \\ -A_{s(i)}^i \; (\tilde{c}_{s(i)} + \tilde{z}_{s(i)}) & A_{s(i)}^i \end{bmatrix}$$

$$\Phi_i = [0, 0, 1, 0, 0, 0]^T$$

Composite inertial matrix for i-th body:

$$I_i^H = \begin{bmatrix} {}^i I_i^H & m_i \tilde{d}_i \\ m_i \tilde{d}_i^T & m_i E \end{bmatrix} = \begin{bmatrix} ({}^i I_i^{CM} - m_i \tilde{d}_i^2) & m_i \tilde{d}_i \\ -m_i \tilde{d}_i & m_i E \end{bmatrix}$$

Velocity squares dependent forces :

$$\beta_i^H = - \begin{bmatrix} {}^i\omega_i \times {}^i I_i^H \; {}^i\omega_i \\ m_i \; {}^i\omega_i \times ({}^i\omega_i \times d_i) \end{bmatrix}$$

Recursive relations for accelerations

Recursive computation of acceleration of reference point H of i-th body:

$$\mathbf{a}_i^H = \mathbf{C}_i \, \mathbf{a}_{p(i)}^H + \Phi_i \, \ddot{s}_i + \eta_i$$

Transformation matrix from p(i)-th coordinate system to i-th :

$$\mathbf{C}_i = \begin{bmatrix} \mathbf{A}_i^{p(i)} & 0 \\ -\mathbf{A}_i^{p(i)} \, (\tilde{\mathbf{c}}_i + \tilde{\mathbf{z}}_i) & \mathbf{A}_i^{p(i)} \end{bmatrix}$$

Velocity squares dependent forces :

$$\eta_i = \begin{bmatrix} (\mathbf{A}_i^{p(i)} \, {}^{p(i)}\omega_{p(i)}) \times {}^{i}\omega_{p(i)i} \\ \mathbf{A}_i^{p(i)} \, ({}^{p(i)}\tilde{\omega}^2_{p(i)} \, (\tilde{\mathbf{c}}_i + \tilde{\mathbf{z}}_i)) + (\mathbf{A}_i^{p(i)} \, {}^{p(i)}\omega_{p(i)}) \times 2 \, \dot{\mathbf{s}}_i \end{bmatrix}$$

Adjustment of equations of motions

$$I_i^H \, a_i^H = f_i + f_{Ei}^H + \beta_i^H$$ Equations of motion for last body

$$\Phi_i^T \, I_i^H \, a_i^H = \Phi_i^T \, f_i + \Phi_i^T \, (f_{Ei}^H + \beta_i^H)$$ Elimination of reaction forces f_i

Substitution of recursive formula for acceleration

$$\Phi_i^T \, I_i^H \, \Phi_i \, \dot{s}_i + \Phi_i^T \, I_i^H \, C_i \, a_{p(i)}^H + \Phi_i^T \, I_i^H \, \eta_i = \Phi_i^T \, (f_{Ei}^H + \beta_i^H)$$

Evaluation of relative acceleration in kinematic joint

$$\dot{s}_i = M_i^{-1} \, (\Phi_i^T \, (f_{Ei}^H + \beta_i^H) - \Phi_i^T \, I_i^H \, (C_i \, a_{p(i)}^H + \eta_i)) \qquad M_i = \Phi_i^T \, I_i^H \, \Phi_i$$

Modifications of equations of motion of previous body

$$I_{p(i)}^H \, a_{p(i)}^H = f_{p(i)} - C_i^T \, f_i + f_{Ep(i)}^H + \beta_{p(i)}^H$$

$$I_{p(i)}^{H*} = I_{p(i)}^H + C_i^T \, N_i \, C_i$$

$$\beta_{p(i)}^{H*} = \beta_{p(i)}^H - C_i^T \, \gamma_i$$

$$(I_{p(i)}^H + C_i^T \, N_i \, C_i) \, a_{p(i)}^H = f_{p(i)} + f_{Ep(i)}^H + \beta_{p(i)}^H - C_i^T \, \gamma_i$$

Adjustment of equations of motions-detail

Evaluated relative acceleration in kinematic joint is substituted to recursive relation for acceleration and acceleration to equations of motion.

$$I_i^H\,(C_i a_{p(i)}^H + \Phi_i M_i^{-1}(\Phi_i^T(f_{Ei}^H + \beta_i^H) - \Phi_i^T I_i^H(C_i a_{p(i)}^H + \eta_i)) + \eta_i) = f_i + f_{Ei}^H + \beta_i^H$$

The reaction forces can be evaluated from the equation

$$(E - I_i^H\,\Phi_i M_i^{-1}\Phi_i^T) I_i^H (C_i a_{p(i)}^H + \eta_i) - (E - I_i^H\,\Phi_i M_i^{-1}\Phi_i^T)(f_{Ei}^H + \beta_i^H) = f_i$$

The new substitutions are introduced

$$P_i = E - I_i^H\,\Phi_i M_i^{-1}\Phi_i^T$$

$$N_i = (E - I_i^H\,\Phi_i M_i^{-1}\Phi_i^T)\,I_i^H = P_i\,I_i^H$$

$$\gamma_i = N_i\,\eta_i - P_i\,(f_{Ei}^H + \beta_i^H)$$

Shortly:

$$f_i = N_i\,C_i\,a_{p(i)}^H + \gamma_i$$

The whole algorithm - 3 phases

for $i = 1$ *to* n *compute* $C_i, \Phi_i, \eta_i, f_{Ei}^H, \beta_i^H$

1

$$I_n^{H*} = I_n^H$$
$$\beta_n^{H*} = \beta_n^H$$

for $i = n$ *to* 1 *compute*

begin

2

$$M_i = \Phi_i^T I_i^{H*} \Phi_i$$
$$P_i = E - I_i^{H*} \Phi_i M_i^{-1} \Phi_i^T$$
$$N_i = P_i I_i^{H*}$$
$$\gamma_i = N_i \eta_i - P_i (f_{Ei}^H + \beta_i^{H*})$$

if $i > 1$ *then*

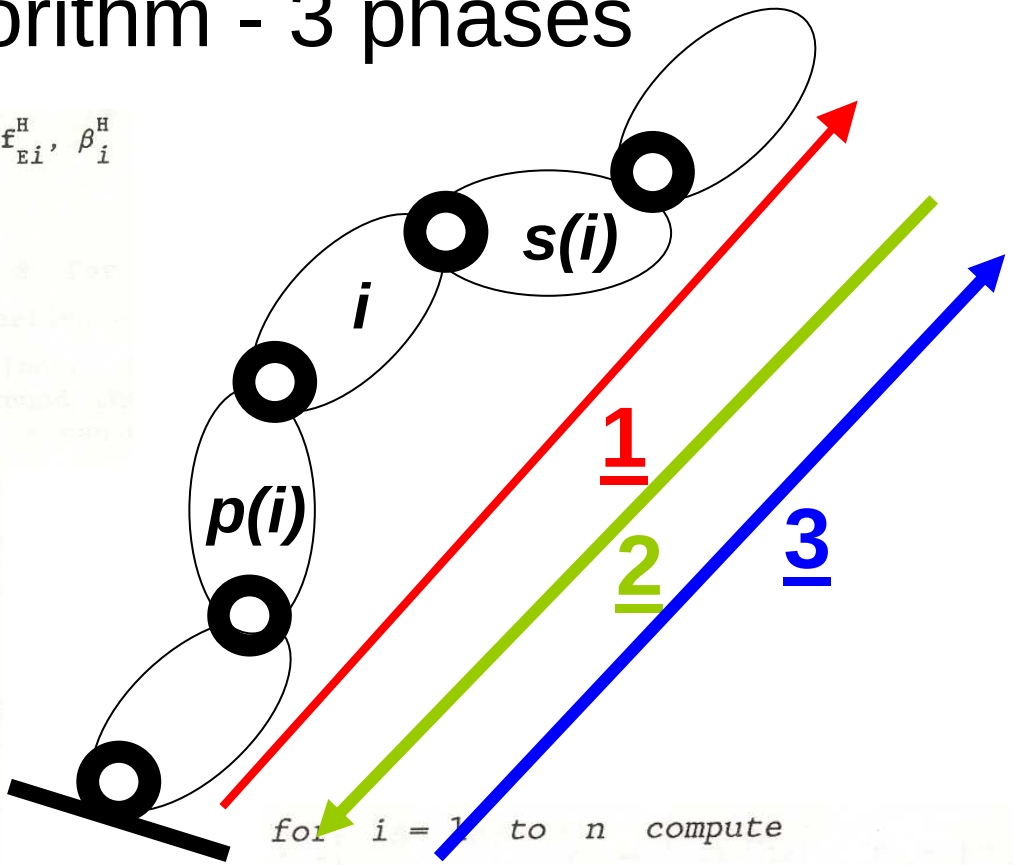
begin

$$I_{p(i)}^{H*} = I_{p(i)}^H + C_i^T N_i C_i$$
$$\beta_{p(i)}^{H*} = \beta_{p(i)}^H - C_i^T \gamma_i$$

end

end

$$a_{p(1)}^H = -g$$



for $i = 1$ *to* n *compute*

begin

$$\dot{s}_i = M_i^{-1} \Phi_i^T ((f_{Ei}^H + \beta_i^{H*}) - I_i^{H*} (C_i a_{p(i)}^H + \eta_i))$$
$$a_i^H = (C_i a_{p(i)}^H + \eta_i) + \Phi_i \dot{s}_i$$
$$\{ f_i = N_i C_i a_{p(i)}^H + \gamma_i \}$$

end

3