SNATI
Seminar Nasional
Aplikasi Teknologi Informasi 2004
Yogyakarta, 19 Juni 2004

Pemodelan Dinamik Robot Enam Derajat Kebebasan

Syachril Fauzie
Jurusan Teknik Komputer, Unikom
Jl. Dipati Ukur No. 112-114-116, Bandung
Telp. (022) 2504119, 2503371, Faks. (022) 2533754
e-mail: SyachrilFauzie@yahoo.com

Abstract

Precise controlling at one particular high-speed manipulator require realistic dynamics model from an arm (end effector). Used approach here is to pass simulation modeling made in a few function by using approach of equations pursuant to co-ordinate concept, style and Force. Where from this equation will be obtained the level of torsi at each joint which depend on inertia of manipulator, gravitation, friction, style of corolois and is centrifugal. Maximal values and minimum values from every this equation is later expected can useful in designing mechanical structure and controller.

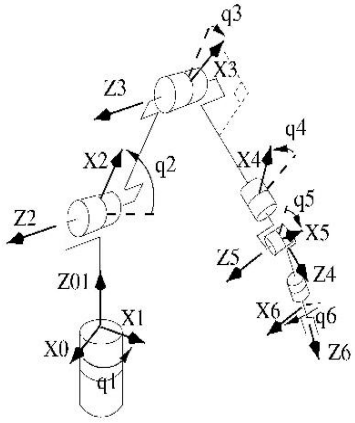
Keywords: *lagrange-euler, corolois, torque, joint.*

1. Pendahuluan

Studi dinamika manipulator adalah studi yang mempelajari formulasi matematis dari persamaan gerak manipulator yang menghasilkan persamaan-persamaan dinamika yang dapat menggambarkan sifat-sifat dinamika manipulator itu sendiri. Persamaan dinamika ini merupakan persamaan yang menghubungkan besaran-besaran dinamik seperti energi kinetik, energi potensial , gaya gravitasi, fiktif, dengan konstanta dinamik, orientasi, posisi, kecepatan, dan percepatan sendi-ruas (*joint-link*).

Analisa dinamika yang dibahas dalam paper ini adalah : Masalah dinamika langsung (*Forward Dynamics*) dan masalah dinamika invers (*Inverse Dynamics*) dari sebuah manipulator 6 DOF (gambar 1). Masalah dinamika langsung berhubungan dengan gerak yang akan diperoleh dari manipulator bila pada sendi-sendi diberikan gaya atau momen tertentu. Sedangkan masalah dinamika invers berhubungan dengan gaya yang harus diberikan pada artikulasi untuk suatu gerak dan lintasan (*trayektori*) '*end effector'* tertentu. Banyak metode yang dikembangkan disini, diantaranya Formulasi Lagrange-Euler, Generalized D'Alembert, Newton-Euler, Recursive Newton-Euler dan Recursive Lagrange-Euler. Model pergerakan dari manipulator ini akan ditampilkan daam simulasi dengan program Matlab dimana digunakan pendekatan analisa dari metode di atas.

Gambar 1. Puma 560 (6 DOF)



Gambar 2. Struktur dari masing-masing sendi

2. Dinamika Lagrange-Euler

Sistem dinamika yang kompleks dapat dimodelkan dengan relatif sederhana oleh persamaan Lagrange. Dimisalkan K dan P menjelaskan energi kinetik dan energi potensial pada lengan. Fungsi Lagrange adalah selisih antara energi kinetik dengan energi potensial.

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = F_i \qquad \ldots\ldots (1)$$

$$i = 1,2,.....,n$$

dimana $\dot{q}_i$ = vektor kecepatan sendi

Persamaan (1) berlaku umum untuk sistem derajat kebebasan banyak.

2.1 Kecepatan sendi

Kecepatan adalah turunan pertama dari fungsi posisi terhadap waktu. Untuk memperoleh kecepatan gerak suatu titik terlebih dahulu dicari vector posisi titik tersebut dalam ruang. Bila $^{i}\vec{P}$ adalah vektor posisi yang menyatakan susunan sumbu koordinat sendi ke-i maka representasi $^{i}\vec{P}$ adalah:

$$\vec{P} = T_i \, {}^{i}\vec{P} \qquad \ldots\ldots (2)$$

sehingga kecepatan titik tersebut adalah:

$$\vec{V} = \frac{d\vec{P}}{dt} = \left[\sum_{j=1}^{i} \frac{\partial T_i}{\partial q_j}\dot{q}_j\right] {}^{i}\vec{P} \qquad \ldots\ldots (3)$$

Untuk menghitung energi kinetik, harus dicari kuadrat dari kecepatan, yaitu:

$$\left|\vec{V}\right|^2 = \left|\frac{d\vec{p}}{dt}\right|^2 = tr(\vec{v}.\vec{v}^{t}) \qquad \ldots\ldots (4)$$

dimana tr adalah operator 'trace'

$$\left|\vec{V}\right|^2 = tr\left[\sum_{j=1}^{i}\sum_{k=1}^{i}\frac{\partial T_i}{\partial q_i}\,{}^{i}\vec{p}\;{}^{i}\vec{p}^{\,t}\,\frac{\partial T_i^{\,t}}{\partial q_k}\dot{q}_j\,\dot{q}_k\right] \qquad \ldots\ldots (5)$$

2.2 Energi Kinetik dan Energi Potensial

Energi kinetik satu partikel yang masanya dm pada ruas ke-i, dinyatakan dengan $dk_i = \frac{1}{2}dm(\vec{v})^2$. Persamaan ini disubstitusikan dengan persamaan (5) menjadi:

$$k_i = \int_{ruas} dk_i = \frac{1}{2}tr\left[\sum_{j=1}^{i}\sum_{k=1}^{i}\frac{\partial T_i}{\partial q_j}\left(\int {}^{i}\vec{p}\;{}^{i}\vec{p}^{\,t}\,dm\right)\frac{\partial T_i^{\,t}}{\partial q_k}\dot{q}_j\,\dot{q}_k\right]$$

dimana

$\int {}^{i}\vec{p}\;{}^{i}\vec{p}^{\,t}\,dm = J_i$ adalah Matriks Inersia Semu (*Pseudo Inertia Matric*).

Sedangkan persamaan energi potensial benda dalam ruang gravitasi $\vec{g} = (g_x, g_y, g_z, 0)^t$, dalam hal ini dari sendi ke-i dengan vektor titik pusat massa ${}^{i}\vec{p}_i$ dinyatakan dengan $\vec{p}_i$.

Untuk keenam ruas didapat:

$$p = \sum_{i=1}^{6} p_i = -\sum_{i=1}^{6} m_i\,\vec{g}^{\,t}\;{}^{i}T_i\;{}^{i}\vec{p}_i \qquad \ldots\ldots (6)$$

; m_i adalah masa ekivalen dari ruas/bagian ruas

2.3 Persamaan Dinamika

Proses menurunkan persamaan di atas beserta manipulasi matematis akan menghasilkan persamaan dalam bentuk:

$$F_i = \sum_{j=1}^{6} D_{ij}\,\ddot{q}_j + Ia_i\,\ddot{q}_i + \sum_{j=1}^{6}\sum_{k=1}^{6} D_{ijk}\,\ddot{q}_j\,\ddot{q}_k + D_i \qquad \ldots\ldots (7)$$

dimana

$$D_{ij} = \sum_{p=mak_{ij}}^{6} trace\left(\frac{\partial T_p}{\partial q_j}J_p\frac{\partial T_p^{\,t}}{\partial q_i}\right) \qquad \ldots\ldots (8)$$

$$D_{ijk} = \sum_{p=mak_{ijk}}^{6} trace\left(\frac{\partial^2 T_p}{\partial q_j}J_p\frac{\partial T_p^{\,t}}{\partial q_i}\right) \qquad \ldots\ldots (9)$$

$$D_i = \sum_{j=i}^{6} -m_p\,\vec{g}^{\,t}\frac{\partial T_p}{\partial q_i}\,{}^{p}\vec{r}_p \qquad \ldots\ldots (10)$$

dengan ${}^{p}\vec{p}_p$ adalah vector pusat masa ruas/bagian ruas p yang dinyatakan pada susunan sumbu koordinat.

Bentuk pertama persamaan (7) adalah sebuah bentuk percepatan yang menerangkan gaya inersia pada lengan. Bentuk kedua adalah bentuk gaya korolois dan sentrifugal, sedangkan bentuk ketiga adalah gaya gravitasi dan bentuk keempat adalah gaya gesek. Gaya Korolois dan sentrifugal dapat diabaikan apabila lengan robot bergerak pada kecepatan rendah, sedangkan faktor yang sangat mempengaruhi terhadap besarnya torsi adalah gaya inersia dan gravitasi.

3. Metode Pemodelan

Pemodelan dinamika meliputi unsur Dinamika Langsung dan Dinamika Inverse. Pada dinamika langsung akan mencari besarnya percepatan sendi (*joint acceleration*) dari suatu nilai yang diberikan berdasarkan fungsi *accel* yang dibuat. Besarnya kecepatan sendi disini tidak hanya berdasarkan fungsi *accel* saja tetapi adalah integrasi dari beberapa fungsi yang pada akhirnya nanti akan menentukan besarnya torsi dari sebuah sendi (*joint torsi*). Hasil akhir pergerakan akan diplot dengan waktu dalam suatu grafik fungsi. Sedangkan pada dinamika invers menghitung torsi dari sebuah sendi yang dibutuhkan untuk menghasilkan posisi sendi, kecepatan dan percepatan. Pada bagian ini digunakan Formulasi recursive Newton-Euler sebagai bentuk algoritma matriks yang efisien untuk menghitung besar dinamika invers. Dinamika invers memerlukan inersia dan masa dari setiap ruas/bagian ruas sebagai parameter kinematik.Yang pada akhirnya akan menentukan posisi setiap titik dari suatu trayektorinya yang juga akan diplot dengan waktu dalam suatu bentuk grafik.

4. Analisa dan Hasil Pemodelan

4.1 Dinamika Langsung

Dinamika langsung menitik beratkan pada hal menentukan percepatan, kecepatan dan torsi dari setiap perubahan posisi pada sebuah manipulator. Dimisalkan sebuah manipulator ini pada posisi nol. Percepatan yang diberikan akan didapat dari suatu fungsi: accel(p560, qz, zeros(1,6), zeros(1,6)). Di mana accel adalah fungsi percepatan sendi, P560 adalah fungsi dari sebuah manipulator 6 DOF dimana aslinya adalah Puma560. Apabila diintegrasikan semua fungsi diatas maka didapat besarnya percepatan berdasarkan tabel1.

Tabel 1. Besar percepatan dari masing-masing sendi

Sendi (joint)	***Percepatan***
Ke-1	0.2462
Ke-2	8.8629
Ke-3	3.1462
Ke-4	0.0021
Ke-5	0.0603
Ke-6	0.0001

a. Fungsi Percepatan Sendi

```
function qdd = accel(robot, Q, qd,
torque)
        n = robot.n;
        if nargin == 2,
                 q = Q(1:n);
                qd = Q(n+1:2*n);
                torque =
Q(2*n+1:3*n);
        else
                q = Q;
                if length(q) ==
robot.n,
                        q = q(:);
                        qd = qd(:);
                end
        end
```

```
        % menghitung inersia
        %  Besar torsi dari
percepatan setiap joint dengan
tanpa dipengaruhi gravitasi
        M = rne(robot,
ones(n,1)*q', zeros(n,n), eye(n),
[0;0;0]);
        % menghitung gaya
grafitasi dan gaya koriolis
        tau = rne(robot, q', qd',
zeros(1,n));
        qdd = inv(M) * (torque(:)
- tau');
```

Untuk melihat hasil pemodelannya digunakan integrasi fungsi ini dengan `fdyn` dan `fdyn2` yang nantinya akan menghitung torsi sebuah sendi.

b. Fungsi `Fdyn` dan `Fdyn2`

```
function  [t,  q,  qd]  =  fdyn(robot,
t0, t1, torqfun, q0, qd0, varargin)
        v = ver;
        if str2num(v(1).Version)<6,
        end
        n = robot.n;
        if nargin == 3,
                torqfun = 0;
                x0 = zeros(2*n,1);
        elseif nargin == 4,
                x0 = zeros(2*n, 1);
        elseif nargin >= 6,
                x0 = [q0(:); qd0(:)];
        end

        [t,y]  =  ode45('fdyn2',  [t0
    t1],  x0,  [],  robot,  torqfun,
    varargin{:});
        q = y(:,1:n);
        qd = y(:,n+1:2*n);
```

```
function xd = fdyn2(t, x, flag,
robot, torqfun, varargin)

        n = robot.n;

        q = x(1:n);
        qd = x(n+1:2*n);

        % mengevaluasi fungsi torsi
        if isstr(torqfun)
                tau = feval(torqfun,
t, q, qd, varargin{:});
        else
                tau = zeros(n,1);
        end
                qdd = accel(robot,
x(1:n,1), x(n+1:2*n,1), tau);
        xd = [x(n+1:2*n,1); qdd];
```

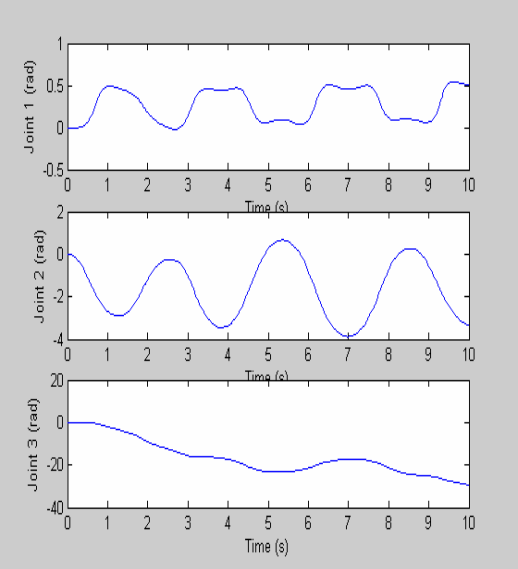
```
tic
[t q qd] = fdyn(nofriction(p560), 0, 10);
   toc
```

Di sini P560 didefinisikan untuk objek P560 dimana kinematika dan dinamikanya menggambarkan karakteristik dari Puma 560 dengan menggunakan standar Denavit-Hartenberg. Objek yang digambar sebagaimana gambar 3 mencakup nilai inersia dan rasio gearnya.

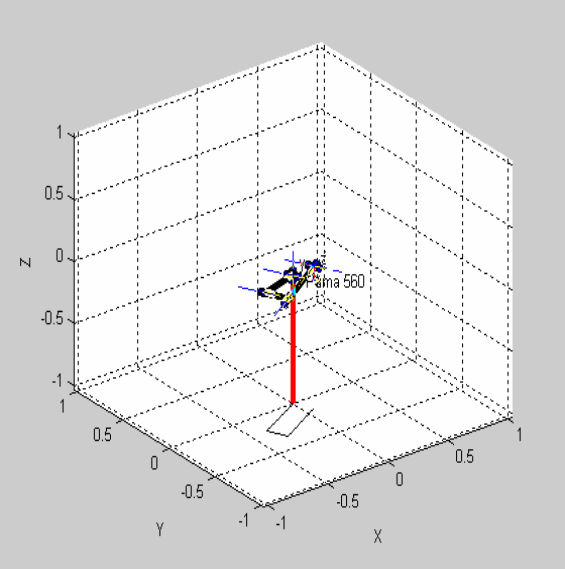
c. Fungsi Manipulator

```
clear L
L{1} = link([ pi/2 0      0       0       0], 'standard');
L{2} = link([ 0    .4318  0       0       0], 'standard');
L{3} = link([-pi/2 .0203  0       .15005  0], 'standard');
L{4} = link([pi/2 0       0       .4318   0], 'standard');
L{5} = link([-pi/2 0      0       0       0], 'standard');
L{6} = link([0    0       0       0       0], 'standard');

L{1}.m = 0;
L{2}.m = 17.4;
L{3}.m = 4.8;
L{4}.m = 0.82;
L{5}.m = 0.34;
L{6}.m = .09;

L{1}.r = [ 0    0     0 ];
L{2}.r = [ -.3638  .006    .2275];
L{3}.r = [ -.0203  -.0141  .070];
L{4}.r = [ 0    .019    0];
L{5}.r = [ 0    0     0];
L{6}.r = [ 0    0     .032];

L{1}.I = [  0       0.35    0       0       0       0];
L{2}.I = [  .13     .524    .539    0       0       0];
L{3}.I = [   .066   .086    .0125   0       0       0];
L{4}.I = [  1.8e-3  1.3e-3  1.8e-3  0       0       0];
L{5}.I = [  .3e-3   .4e-3   .3e-3   0       0       0];
L{6}.I = [  .15e-3  .15e-3  .04e-3  0       0       0];

L{1}.Jm =  200e-6;
L{2}.Jm =  200e-6;
L{3}.Jm =  200e-6;
L{4}.Jm =  33e-6;
L{5}.Jm =  33e-6;
L{6}.Jm =  33e-6;

L{1}.G =  -62.6111;
L{2}.G =  107.815;
L{3}.G =  -53.7063;
L{4}.G =  76.0364;
L{5}.G =  71.923;
L{6}.G =  76.686;

% Friksi
L{1}.B =   1.48e-3;
L{2}.B =   .817e-3;
L{3}.B =    1.38e-3;
L{4}.B =   71.2e-6;
L{5}.B =   82.6e-6;
L{6}.B =   36.7e-6;

L{1}.Tc = [ .395  -.435];
L{2}.Tc = [ .126  -.071];
L{3}.Tc = [ .132  -.105];
L{4}.Tc = [ 11.2e-3 -16.9e-3];
L{5}.Tc = [ 9.26e-3 -14.5e-3];
L{6}.Tc = [ 3.96e-3 -10.5e-3];

qz = [0 0 0 0 0 0]; % sudut nol
qr = [0 pi/2 -pi/2 0 0 0];
qstretch = [0 0 -pi/2 0 0 0];
p560 = robot(L, 'Puma 560', 'Unimation', 'params of 8/95');
clear L
p560.name = 'Puma 560';
p560.manuf = 'Unimation';
```

Simulasi pergerakannya terhadap waktu dapat dilihat dari grafik 3.



Grafik 3. Perubahan posisi sendi terhadap waktu



Grafik 4. Model robot manipulator 6 derajat kebebasan

Pergerakan robot manipulator tergantung pada besarnya gaya sentrifugal dan gaya koriolis.

d. Fungsi Gaya Koriolis dan sentrifugal

```
function c = coriolis(robot, q, qd)
                    c = rne(robot, q, qd, zeros(size(q)), [0;0;0]);
function r = robot(L, a1, a2, a3)
    if nargin == 0
            r.name = 'noname';
            r.manuf = '';
            r.comment = '';
            r.link = {};
            r.n = 0;
            r.mdh = 0;
            r.gravity = [0; 0; 9.81];
r.base = eye(4,4);
r.tool = eye(4,4);
r.handle = [];
        r.q = [];
r.plotopt = {};
r.lineopt = {'Color', 'black', 'Linewidth', 4};
r.shadowopt = {'Color', 'black', 'Linewidth', 1};
r = class(r, 'robot');
    elseif isa(L, 'robot')
        r = L;
    if nargin == 2,
            r.link = a1;
        end
    else
        if nargin > 1,
            r.name = a1;
        else
r.name = 'noname';
        end
        if nargin > 2,
        r.manuf = a2;
        else
        r.manuf = '';
        end
        if nargin > 3,
r.comment = a3;
        else
        r.comment = '';
        end
        if isa(L, 'double')
```

```
            dh_dyn = L;
        clear L
for j=1:numrows(dh_dyn)
        L{j} = link(dh_dyn(j,:));
        end
        r.name = inputname(1);
        r.link = L;
elseif iscell(L) & isa(L{1}, 'link')
        r.link = L;
        else
error('unknown type passed to robot');
        end
        r.n = length(L);
        mdh = [];
for j = 1:length(L)
        mdh = [mdh L{j}.mdh];
           end
        if all(mdh == 0)
        r.mdh = mdh(1);
elseif all (mdh == 1)
        r.mdh = mdh(1);
        else
        error('robot has mixed D&H link conventions');
        end
        r.gravity = [0; 0; 9.81];
        r.base = eye(4,4);
        r.tool = eye(4,4);
        r.handle = [];
        r.q = [];
        r.plotopt = {};
        r.lineopt = {'Color', 'black', 'Linewidth', 4};
        r.shadowopt = {'Color', 'black', 'Linewidth', 1};
        r = class(r, 'robot');
        end
```

4.2 Dinamika Invers

Kecepatan sendi adalah 5 rad/s dan percepatan 1 rad/s^2. Torsi sendi didapat dari fungsi tau = rne(p560, qz, 5*ones(1,6), ones(1,6)).

Tabel 2. Besar torsi dari masing-masing sendi

Sendi (joint)	***Torsi***
Ke-1	79.4048
Ke-2	37.1694
Ke-3	13.5455
Ke-4	1.0728
Ke-5	0.9399
Ke-6	0.5119

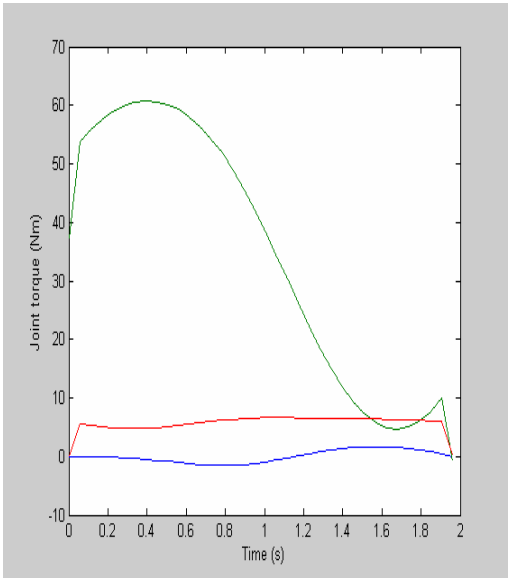
e. Fungsi rne

Fungsi ini adalah untuk menghitung Dinamika Invers dengan persamaan Recursive Newton-Euler.

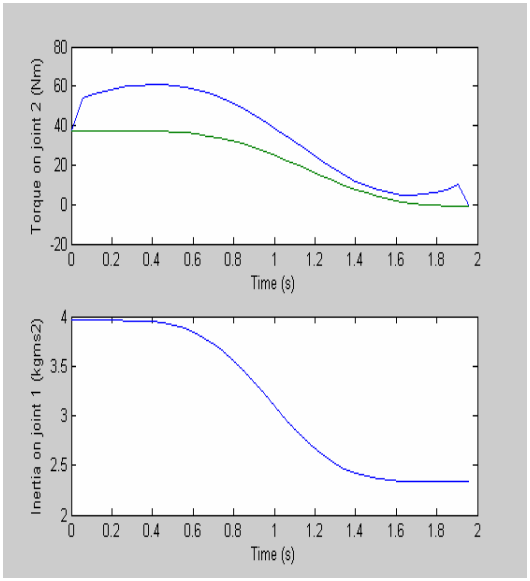
```
function tau = rne(robot, a1, a2, a3, a4, a5)
    if robot.mdh ~= 0,
            error('Jacobian only valid for standard D&H parameters')
    end
    z0 = [0;0;1];
    grav = robot.gravity;
    fext = zeros(6, 1);
    n = robot.n;
    if numcols(a1) == 3*n,
            Q = a1(:,1:n);
            Qd = a1(:,n+1:2*n);
            Qdd = a1(:,2*n+1:3*n);
            np = numrows(Q);
            if nargin >= 3,
                    grav = a2;
            end
```

```
        if nargin == 4,
                fext = a3;
        end
  else
        np = numrows(a1);
        Q = a1;
        Qd = a2;
        Qdd = a3;
        if numcols(a1) ~= n | numcols(Qd) ~= n | numcols(Qdd) ~= n | ...
                numrows(Qd) ~= np | numrows(Qdd) ~= np,
                error('bad data');
        end
        if nargin >= 5,
                grav = a4;
        end
        if nargin == 6,
                fext = a5;
        end
  end
  tau = zeros(np,n);
  for p=1:np,
        q = Q(p,:)';
        qd = Qd(p,:)';
        qdd = Qdd(p,:)';
        Fm = [];
        Nm = [];
        pstarm = [];
        Rm = [];
        w = zeros(3,1);
        wd = zeros(3,1);
        v = zeros(3,1);
        vd = grav;
for j=1:n,
        link = robot.link{j};
        Tj = link(q(j));
        Rm{j} = tr2rot(Tj);
        if link.RP == 'R',
                D = link.D;
        else
                D = q(j);
        end
        alpha = link.alpha;
        pstarm(:,j) = [link.A; D*sin(alpha); D*cos(alpha)];
  end

  for j=1:n,
        link = robot.link{j};
        R = Rm{j}';
        pstar = pstarm(:,j);
        r = link.r;
        if link.RP == 'R',

  % revolute axis
wd = R*(wd + z0*qdd(j) + ...
cross(w,z0*qd(j)));
w = R*(w + z0*qd(j));
%v = cross(w,pstar) + R*v;
vd = cross(wd,pstar) + ...
cross(w, cross(w,pstar)) +R*vd;
else

% prismatic axis
w = R*w;
wd = R*wd;
vd = R*(z0*qdd(j)+vd) + ...
cross(wd,pstar) + ...
2*cross(w,R*z0*qd(j)) +...
cross(w, cross(w,pstar));
end
vhat = cross(wd,r) + ...
cross(w,cross(w,r)) + vd;

  F = link.m*vhat;
  N = link.I*wd + cross(w,link.I*w);
  Fm = [Fm F];
  Nm = [Nm N];
end

f = fext(1:3); % Gaya/momen
  nn = fext(4:6);
  for j=n:-1:1,
        link = robot.link{j};
        pstar = pstarm(:,j);
        if j == n,
        R = eye(3,3);
        else
        R = Rm{j+1};
  end
        r = link.r;
        nn = R*(nn + cross(R'*pstar,f)) + ...
```

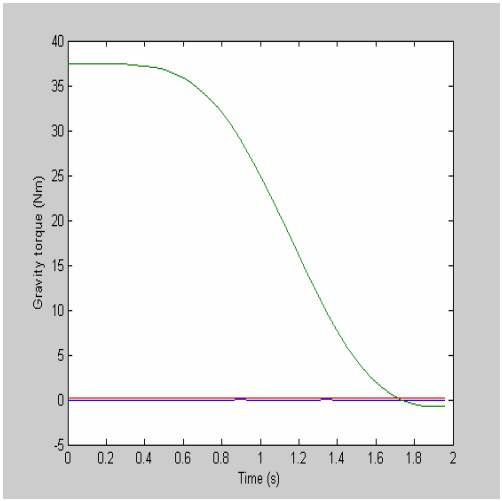
```
            cross(pstar+r,Fm(:,j)) + ...
                     Nm(:,j);
                     f = R*f + Fm(:,j);
                     R = Rm{j};
                     if link.RP == 'R',

                     % revolute
                     tau(p,j) = nn'*(R'*z0) + ...
                     link.G^2 * ( link.Jm*qdd(j) + ...

   friction(link, qd(j)) ...
        );
   else

   % prismatik
   tau(p,j) = f'*(R'*z0) + ...
   link.G^2 * ( link.Jm*qdd(j) + ...
               friction(link, qd(j)) ...
               );
               end
        end
end
```



Gambar 5. Perubahan torsi terhadap waktu



Gambar 6. Perubahan torsi terhadap waktu



Gambar 7. Perubahan gravitasi torsi terhadap waktu

Pada akhirnya kita mendapatkan inersia dari salah satu sendi yaitu sendi pertama dengan membandingkan nilai maksimum dan nilai minimum.

```
M = inertia(p560, q);
    M11 = squeeze(M(1,1,:));
    plot(t, M11);
    xlabel('Time (s)');
    ylabel('Inertia on joint 1 (kgms2)')

max(M11)/min(M11)
ans =
    1.6947
```

5. Kesimpulan

Kinerja optimal dari suatu manipulator robot dapat diperoleh bila menggunakan strategi kontrol. Tetapi pengontrolan yang tepat pada kecepatan tinggi membutuhkan model dinamika yang realistis dari lengan (*arm*). Analisa dinamika manipulator merupakan tahapan yang harus dilakukan sebelum studi pengaturan guna memperoleh transfer fungsi sendi-ruas manipulator. Pergerakan manipulator sebenarnya adalah jatuh dibawah gaya tarik. Tetapi kecepatan putaran dari atas ke bawah dipengaruhi oleh gaya sentrifugal dan korolois.

6. Batasan Penelitian

Pemodelan ini hanya dilakukan pada sebuah manipulator dengan 6 derajat kebebasan. Untuk itu kiranya para peneliti lain yang berkeinginan meneruskan penelitian ini dapat dilakukan pada sebuah manipulator dengan derajat kebebasan banyak (Multiple *degree of freedom*). Sehingga perancangan struktur mekanisnya dapat dilihat dari nilai-nilai maksimum dan nilai minimum yang mempengaruhi pergerakkannya.

Daftar Pustaka

Kreyszig, E. (1999). Advanced Engineering Mathematics. New York: Wiley.

K.S. Fu, Rc. Gonzales, dan C.S.G. Lee.(1987). Robotics: *Control, Sensing, Vision, and Intelligence*. McGraw-Hill.

Ogata, K. (1994). Solving Control Engineering Problems with Matlab. New Jersey: Prentice Hall.

Sciavicco, L, Siciliano, B. (1996). Modeling and Control of Robot Manipulators. McGraw-Hill.