Hinweise & Videos zu den MATLAB-Beispielen

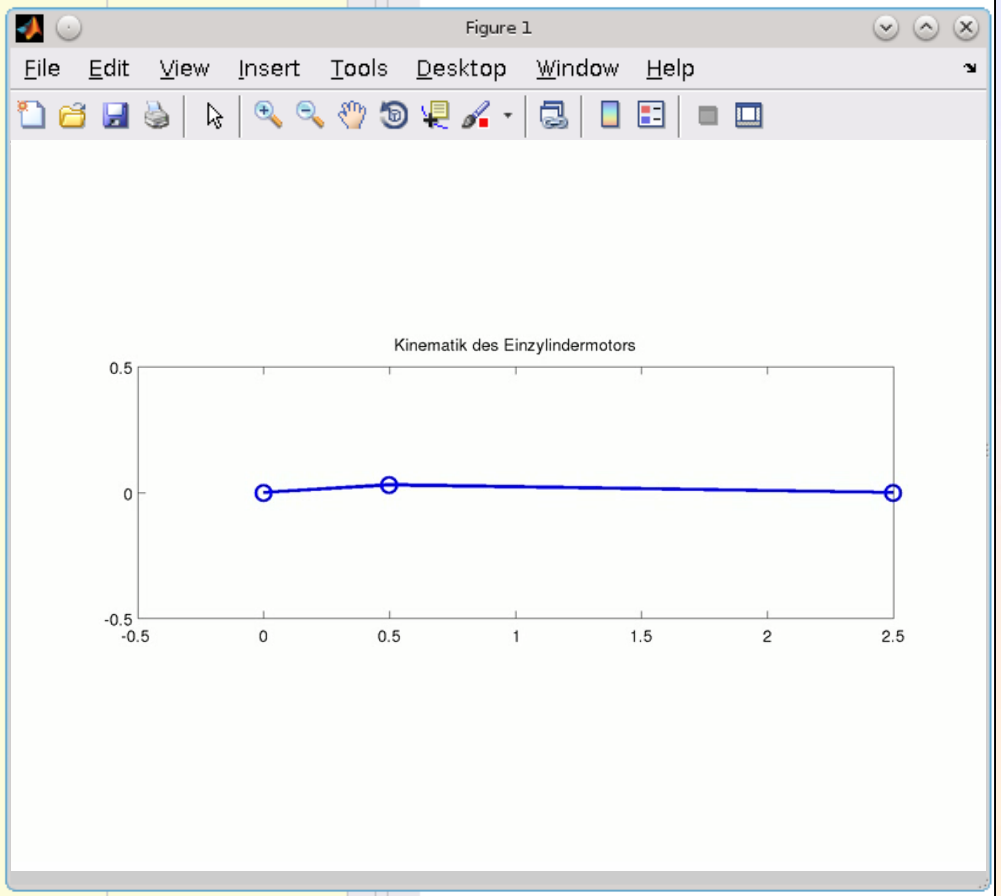
Wie löst man die Aufgaben der Arbeitsblätter mit MATLAB?

Übersicht

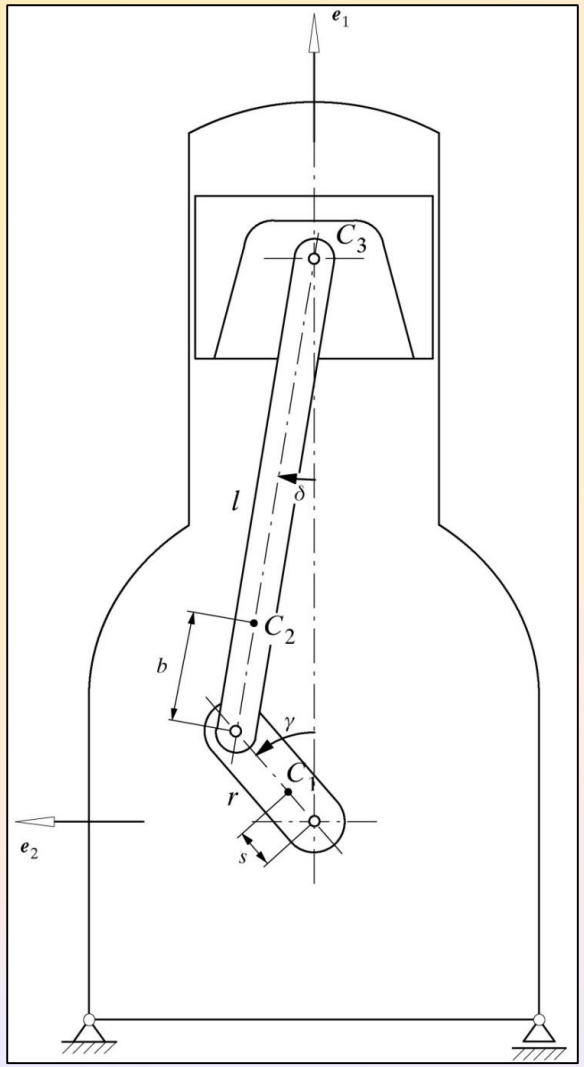
1. Kinematik Animation (Aufgabe B1)
   http://info.itm.uni-stuttgart.de/courses/madyn/Uebungen/mFiles/Einzylindermotor_Kinematik_Anim.m
2. Bewegungsgleichungen und Reaktionsgleichungen des Einzylindermotors
   (Aufgaben B2 und Aufgabe A12)
   http://info.itm.uni-stuttgart.de/courses/madyn/Uebungen/mFiles/Einzylindermotor_Gln.m
3. Zeit-Integration der Bewegungsgleichungen des Einzylindermotors & Animation
   http://info.itm.uni-stuttgart.de/courses/madyn/Uebungen/mFiles/Einzylindermotor_ZeitSim_files.zip
4. Bewegungsgleichungen des Doppelpendels und Zeit-Integration der Bewegungsgleichungen des Doppelpendels & Animation
   http://info.itm.uni-stuttgart.de/courses/madyn/Uebungen/mFiles/Doppelpendel_ZeitSim_files.zip

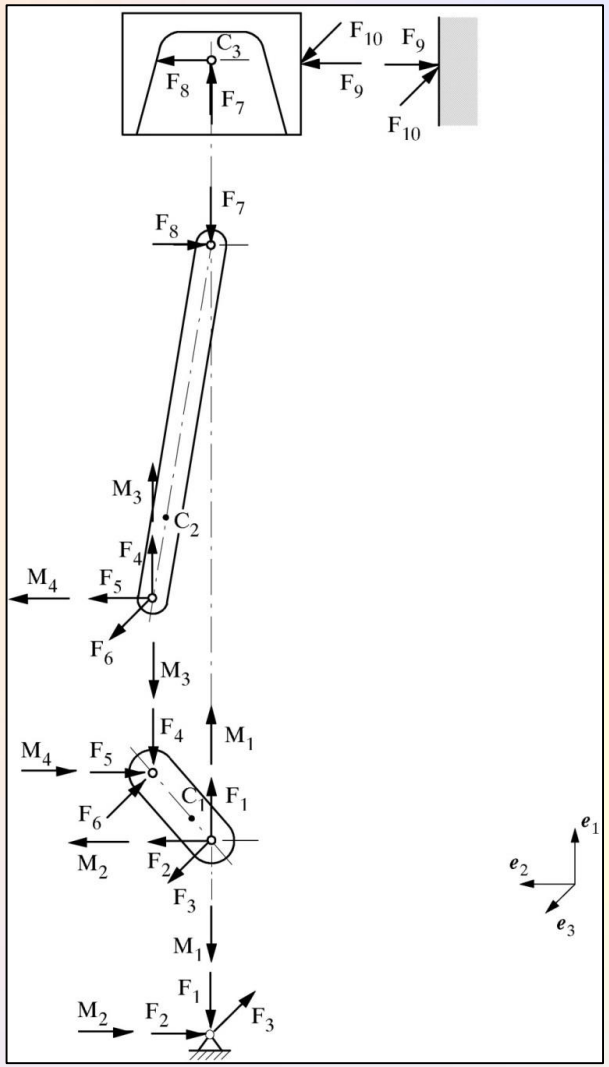
Kinematik Animation in MATLAB

```m
%% Kinematik des Einzylindermotors A8: Animationsbeispiel
syms r l gamma delta          % symbolische Variablen definieren
% Ortsvektoren
r1 = [r*cos(gamma) r*sin(gamma)];
r2 = [r*cos(gamma)+l*cos(delta) 0];

% Schleifen-Zwangsbedingung / Berechnung der Hilfsvariablen
sol = solve(r*sin(gamma)-l*sin(delta) == 0, delta);
delta_sol = sol(1);

% Beispielwerte
r = 0.5;
l = 2;
gamma_ = [1:200]*2*2*pi/200;  % 2 Umdrehungen in 100 Schritten

% Animation
h_fig = figure(1);
for idx = 1:length(gamma_)    % Zeitschritte durchiterieren
    % Werte einsetzen
    gamma = gamma_(idx);
    delta = eval(delta_sol);
    r1_ = eval(r1);
    r2_ = eval(r2);

    % Bild neu zeichnen
    figure(h_fig);cla
    plot([0 r1_(1) r2_(1)],[0 r1_(2) r2_(2)],'-o','linewidth',2,...
        'markersize',10)
    title('Kinematik des Einzylindermotors')
    axis equal
    xlim([-r r+l])
    ylim([-r r])

    pause(0.03)               % kurz halten und zeigen
end
```



Bewegungsgleichungen und Reaktionsgleichungen des Einzylindermotors





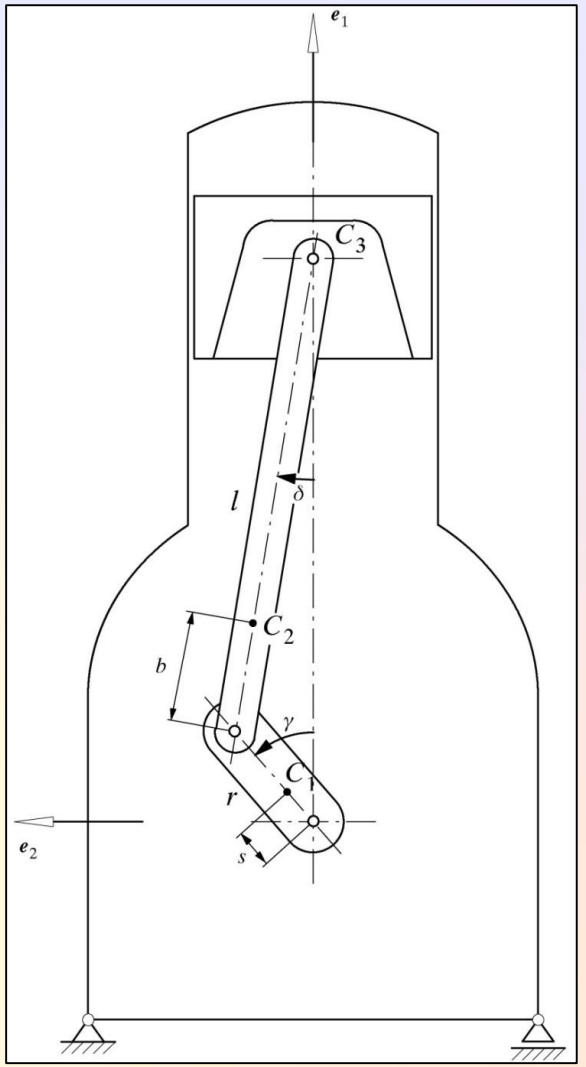
Bewegungsgleichungen und Reaktionsgleichungen des Einzylindermotors

Editor - Einzylindermotor_Gln.m

```
syms r s b l lambda gamma Dgamma D2gamma m1 m2 m3 g Kr I1 I2 MA FG
syms delta deltaStr delta2Str
syms F1 F2 F3 F4 F5 F6 F7 F8 F9 F10 M1 M2 M3 M4

%% Bewegungsgleichung aus haendischer Herleitung (A8, A15, A16)
% Kinematik
% ---------
% Kurbel
J_T1 = [-s*sin(gamma)
         s*cos(gamma)
         0];
a1q = [-s*cos(gamma)
       -s*sin(gamma)
        0]*Dgamma^2;
J_R1 = [0
        0
        1];
alpha1q = [0
           0
           0];

% Pleuel
J_T2 = [-r*sin(gamma)-b*deltaStr*sin(delta)
         r*cos(gamma)-b*deltaStr*cos(delta)
         0];
a2q = [-b*delta2Str*lambda*sin(gamma)-(r+b*deltaStr*lambda)*cos(gamma)
       (r-b*lambda)*sin(gamma)
        0]*Dgamma^2;
J_R2 = [ 0
         0
        -deltaStr];
alpha2q = [ 0
            0
           -delta2Str*gamma];

% Kolben (Punktmasse)
J_T3 = [-r*sin(gamma)-l*deltaStr*sin(delta)
         0
         0];
a3q = [-l*delta2Str*lambda*sin(gamma)-(r+l*deltaStr*lambda)*cos(gamma)
       0
       0]*Dgamma^2;

% globale Jacobi-Matrix J
J = [J_T1;
```

Command Window

```
>> Einzylindermotor_Gln

M =

I1 + m2*(r*cos(gamma) - b*deltaStr*cos(delta))^2 + m2*(r*sin(gamma) + b

k =

Dgamma^2*m2*(r*sin(gamma) + b*deltaStr*sin(delta))*(cos(gamma)*(r + b*d

q =

(r*sin(gamma) + deltaStr*l*sin(delta))*(FG + g*m3) - MA + g*m2*(r*sin(g

test =

[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

>>
```



Bewegungsgleichungen und Reaktionsgleichungen des Einzylindermotors

1. Integration der **Bewegungsgleichungen**:

$$\text{Trafo: } \mathbf{x} = \begin{bmatrix} \mathbf{y} \\ \dot{\mathbf{y}} \end{bmatrix} \xrightarrow{\text{ODE}} \underbrace{\begin{bmatrix} \dot{\mathbf{y}} \\ \ddot{\mathbf{y}} \end{bmatrix}}_{\dot{\mathbf{x}}} = \underbrace{\begin{bmatrix} \dot{\mathbf{y}} \\ \mathbf{M}^{-1} \cdot (\mathbf{q} - \mathbf{k}) \end{bmatrix}}_{\mathbf{f}(\mathbf{x},\mathbf{u},t)} \xrightarrow{\int \dots \, dt} [\mathbf{t}, \mathbf{x}] \longrightarrow \mathbf{y}, \dot{\mathbf{y}}, t$$

2. Berechnung der Reaktionskräfte aus den Reaktionsgleichungen

$$\overbrace{\overline{\mathbf{Q}}^{T} \cdot \bar{\bar{\mathbf{M}}}^{-1} \cdot \overline{\mathbf{q}}^{c}}^{\hat{\mathbf{k}}} = \overbrace{\overline{\mathbf{Q}}^{T} \cdot \bar{\bar{\mathbf{M}}}^{-1} \cdot \overline{\mathbf{q}}^{e}}^{\hat{\mathbf{q}}} - \overbrace{\overline{\mathbf{Q}}^{T} \cdot \bar{\bar{\mathbf{M}}}^{-1} \cdot \overline{\mathbf{Q}}}^{\mathbf{N}} \cdot \mathbf{g}$$

$$\Rightarrow \mathbf{g} = \mathbf{N}(\mathbf{y}, t)^{-1} \cdot \left(\hat{\mathbf{k}}(\mathbf{y}, \dot{\mathbf{y}}, t) - \hat{\mathbf{q}}(\mathbf{y}, \dot{\mathbf{y}}, t) \right)$$

Zeit-Integration der Bewegungsgleichungen & Animation in MATLAB

Editor - /home/itm/institut/Lehre/Maschinendynamik/Sonstiges/SimulationenMatlab/Einzylindermotor/Einzy...

Einzylindermotor_ZeitSim.m | beweggl_funktion.m | gaskraft_funktion.m | reaktionskraefte_funktion.m

This file can be published to a formatted document. For more information, see the publishing video or help.

```
1   %% Einzylindermotor-Zeitsimulation & Animation (A8, A15, A16)
2   % Simulation: Loesung der nichtlinearen Zustandsdifferentialgleichung
3
4   % Anfangsbedingenungen
5   x0 = [5*pi/180; 3*360*pi/180]; % [rad rad/s]
6
7   % Randbedingungen
8   tspan = 0:0.0025:1.5;
9
10  % Integration der Zustandsgleichung
11  [t, x] = ode45('beweggl_funktion',tspan,x0);
12
13  figure(1); clf;
14  gammaV  = x(:,1);    % gamma Integrationsergebnis
15  DgammaV = x(:,2);    % Dgamma Integrationsergebnis
16  plot(t,gammaV*180/pi,'b',t,DgammaV*180/pi,'r')
17  legend({'gamma','Dgamma'})
18  title('Trajektorien')
19
20  % Berechnung der zugehoerigen Lagerreaktionskraefte & Gaskraft
21  [g_, FG_] = arrayfun(@reaktionskraefte_funktion,gammaV,DgammaV,'UniformOutput',false);
22  g_V = vertcat(g_{:});
23  FG_V = vertcat(FG_{:});
24
25  figure(2);clf;hold on
26  plot(t,g_V(:,[1 2 4 5 6 8 9]))
27  plot(t,FG_V,'linewidth',2)
28  legend({'F1','F2','F4','F5','F6','F8','F9','FG_V'})
29  title('Reaktionskraefte und Gaskraft')
30
31  %% Interpolation und Animation der Ergebnisse
32  dt      = 0.003; % Zeitschritt
33  t_interp = 0:dt:max(t);
34  gammaV_interp = interp1(t,gammaV,t_interp');
35  g_V_interp = interp1(t,g_V,t_interp');
36  FG_V_interp = interp1(t,FG_V,t_interp');
37
38  % Plot-Werte
39  LWidth_ = 2;    % Line-Width
40  JSize   = 8;    % Joint-Size
41  ASize   = 7;    % Arrow-Size
42  ADist   = 0.08; % Arrow-Distance
43  FScale = 2.5e-4;% Force-Scale
```

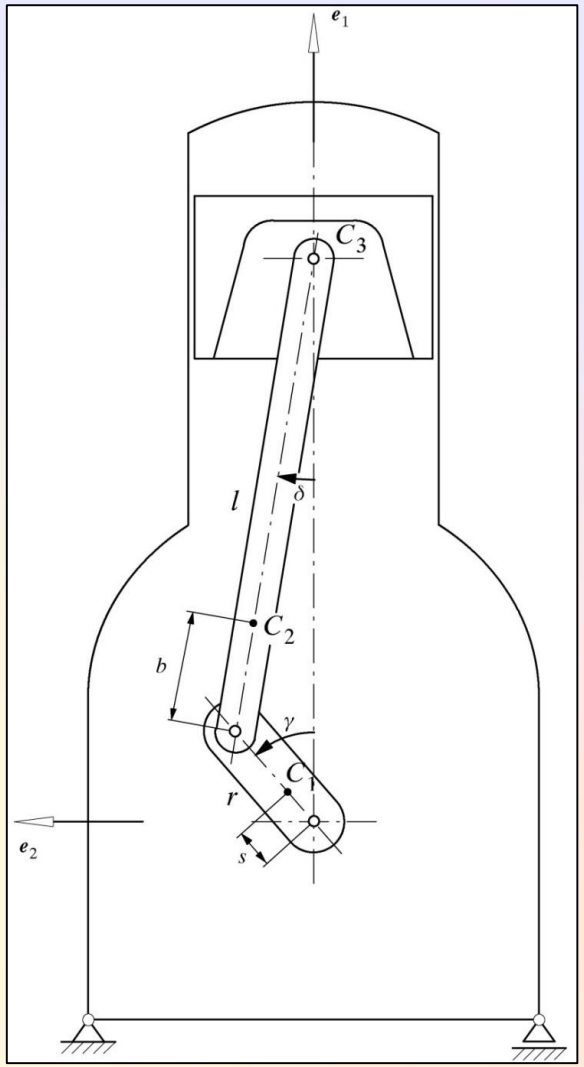
pulsierende Gaskraft

gaskraft_funktion

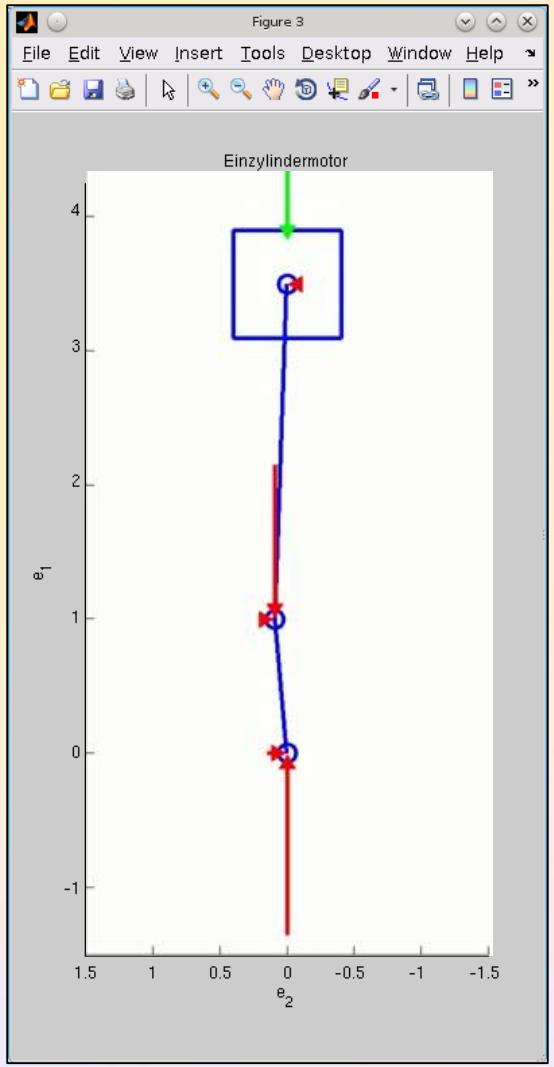
Command Window

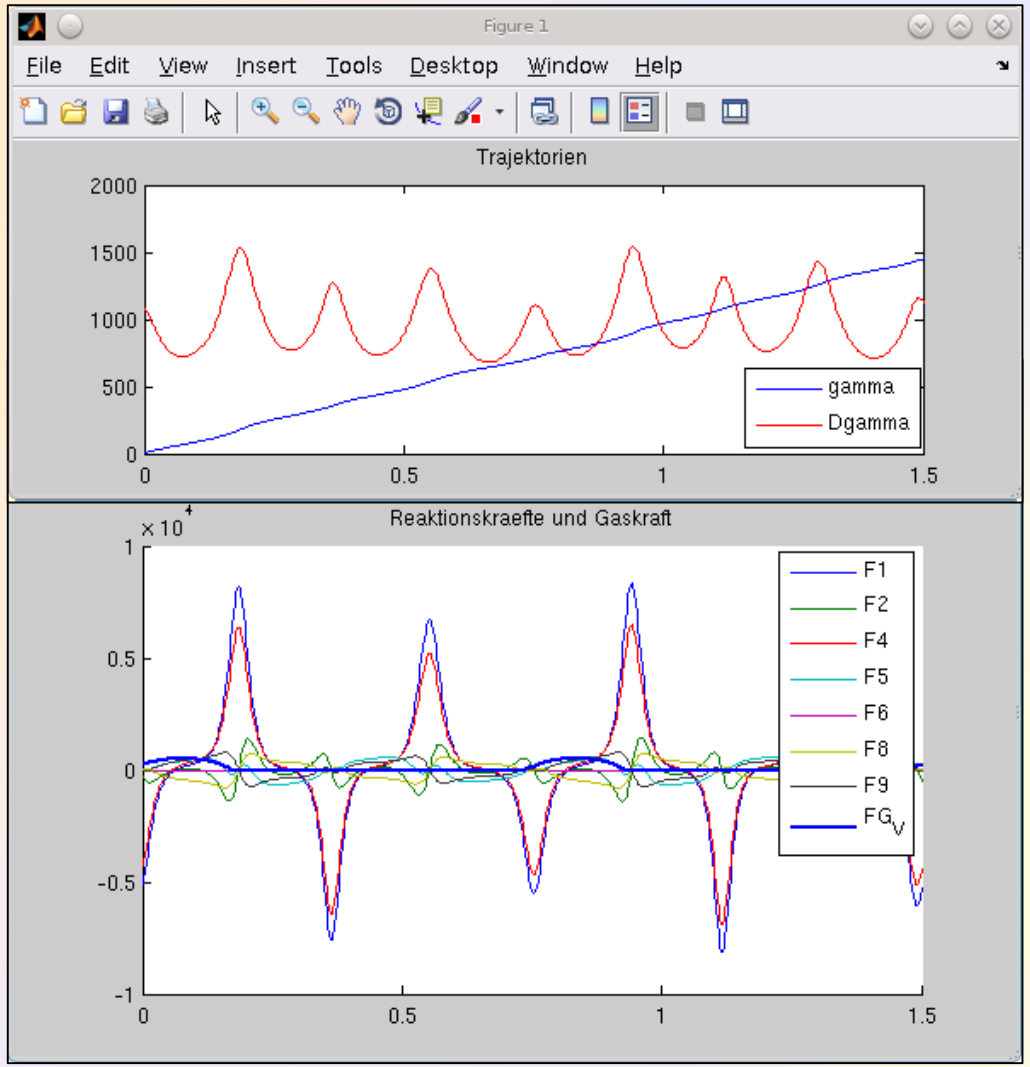
New to MATLAB? Watch this Video, see Examples,

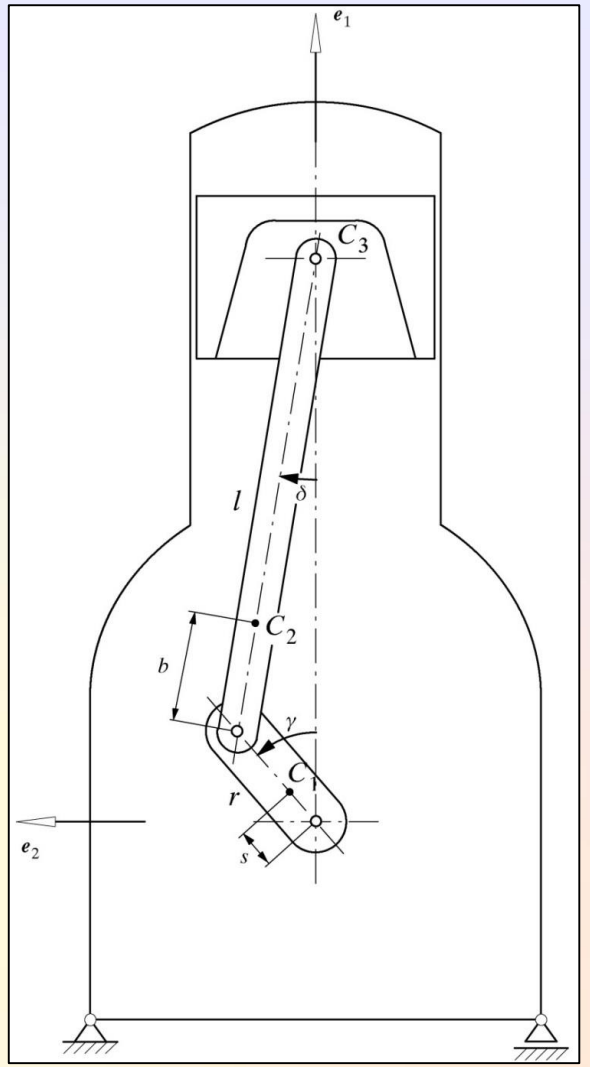
```
>> Einzylindermotor_ZeitSim
>>
>>
```



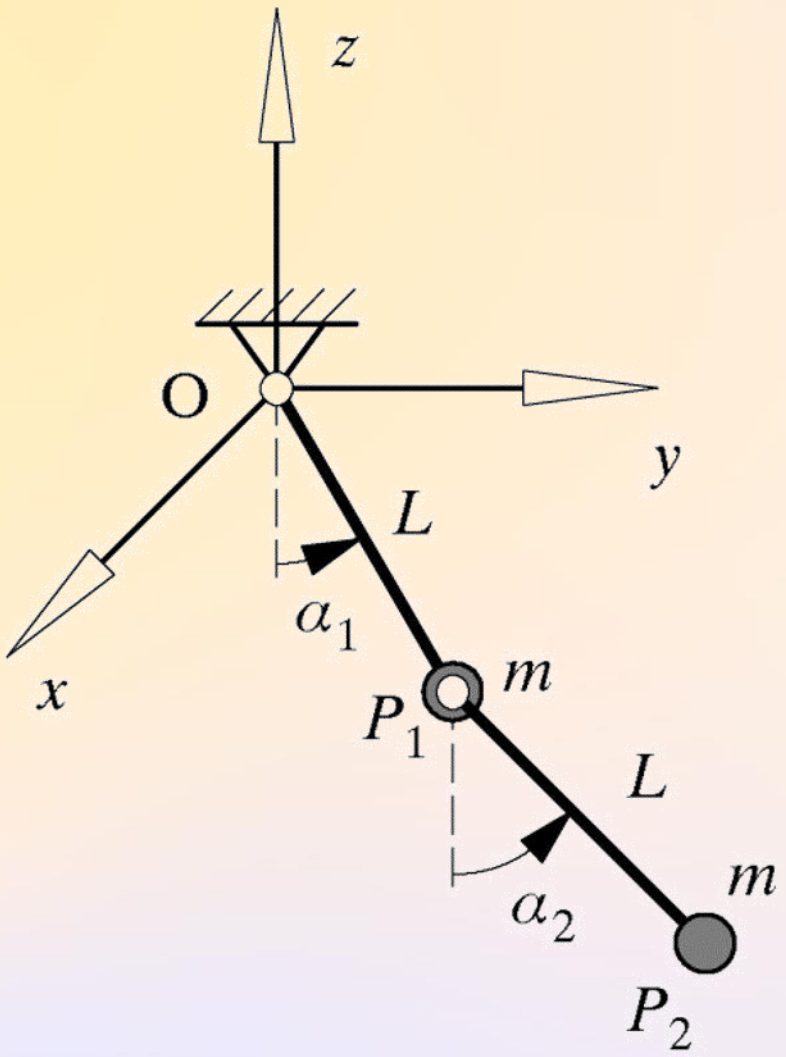
Zeit-Integration der Bewegungsgleichungen & Animation in MATLAB







Bewegungsgleichungen des Doppelpendels, A14, Herleitung in MATLAB



Editor - Doppelpendel_Herleitung_BewegGl.m | Variables - g_{1, 1}

Doppelpendel_ZeitSim.m | Doppelpendel_Herleitung_BewegGl.m

This file can be published to a formatted document. For more information, see the publishing video or help.

```
44 -    a1 = subs(a1_,t_,0);
45 -    a2 = subs(a2_,t_,0);
46
47      % Jacobi-Matrizen berechnen
48 -    Jt1 = jacobian(r1,y);
49 -    Jt2 = jacobian(r2,y);
50
51      % lokale Beschleunigungen ausrechnen: ai = Jti*ydd+aiq --> aiq = ai-Jti*ydd
52 -    a1q = simple(a1-Jt1*ydd);
53 -    a2q = simple(a2-Jt2*ydd);
54
55      % globale Massenmatrix  M_querquer
56 -    Mqq = diag([m1 m1 m1 m2 m2 m2]);
57
58      % Globale Jacobi-Matrix J_quer
59 -    Jq  = [Jt1;
60             Jt2];
61
62      % Newton-Euler-Massen-Matrix M_quer berechnen
63 -    Mq = Mqq*Jq;
64
65      % Vektor der Coriolis-, Zertrifugal- und Kreisel-Kraefte definieren
66 -    qc = [m1*a1q;
67            m2*a2q];
68
69      % Die eingepraegten Kraefte festlegen
70 -    qe = [ 0;
71             0;
72            -m1*g;
73             0;
74             0;
75            -m2*g];
76
77      %% D'Alembertsches Prinzip auf Newton-Eulergleichung anwenden
78      %    Mq*ydd + qc = qe + Qq*g      -->    M*ydd + k = q
79
80      % Massenmatrix M resultiert aus Linksmultipli. mit Jq^T & Vereinfachung
81 -    M = simple(Jq.'*Mq)
82
83      % Vektor der verallgemeinerten Coriolis- Zertrifugal- Kreisel-Kraefte
84 -    k = simple(Jq.'*qc)
85
86      % Vektor der verallgemeinerten eingepraegten-Kraefte
87 -    q = simple(Jq.'*qe)
88
```

Command Window

New to MATLAB? Watch this Video, see Examples, or read Getting Started.

```
>>
```

Run

....

Bewegungsgleichungen des Doppelpendels in der Zustandsform

Transformation:

$$\mathbf{x} = \begin{bmatrix} \mathbf{y} \\ \dot{\mathbf{y}} \end{bmatrix} \xrightarrow{\text{ODE}} \underbrace{\begin{bmatrix} \dot{\mathbf{y}} \\ \ddot{\mathbf{y}} \end{bmatrix}}_{\dot{\mathbf{x}}} = \underbrace{\begin{bmatrix} \dot{\mathbf{y}} \\ \mathbf{M}^{-1} \cdot (\mathbf{q} - \mathbf{k}) \end{bmatrix}}_{\mathbf{f}(\mathbf{x},\mathbf{u},t)}$$

$$\int \dots \, dt$$

$$\mathbf{y}, \dot{\mathbf{y}}, t \longleftarrow [\mathbf{t}, \mathbf{x}]$$

Integration der ODE in MATLAB:

```
% Integration der Zustandsgleichung:
[t, x] = ode45('beweggl_funktion',tspan,x0);
```

```
function dx = beweggl_funktion(t,x)
%% Bewegungsgleichungen (ODE) integrieren
% Beispielhafte Zahlenwerte definieren
l1 = 1;
l2 = 1;
m1 = 1;
m2 = 1;
g  = 9.81;

% Anfangsbedingungen / Bedingungen des letzten Integrationsschritts lesen
alpha1  = x(1);
alpha2  = x(2);
Dalpha1 = x(3);
Dalpha2 = x(4);

% hier muessen die zu integrierenden Bewegungsgleichungen stehen
% (copy+paste vom Ergbinis aus dem Skript "Doppelpendel_Herleitung_BewegGl.m")
M = ...
    [            m1*l1^2 + m2*l2^2, l1*l2*m2*cos(alpha1 - alpha2)
     l1*l2*m2*cos(alpha1 - alpha2),                        l1^2*m2];


k = ...
    [  Dalpha2^2*l1*l2*m2*sin(alpha1 - alpha2)
      -Dalpha1^2*l1*l2*m2*sin(alpha1 - alpha2)];


q = ...
    [ -g*sin(alpha1)*(l1*m1 + l2*m2)
                -g*l1*m2*sin(alpha2)];

% Nichtlineare Zustandsgleichung formulieren
dx = [Dalpha1; Dalpha2; inv(M)*(q-k)];

end % function
```

Zeit-Integration der Bewegungsgleichungen & Animation in MATLAB

Editor - Doppelpendel_ZeitSim.m

Doppelpendel_ZeitSim.m | Doppelpendel_Herleitung_BewegGl.m

This file can be published to a formatted document. For more information, see the publishing video or help.

```
%% Doppelpendel-Zeitsimulation & Animation
% Simulation: Loesung der nichtlinearen Zustandsdifferentialgleichung

% Anfangsbedingungen:
alpha1_0 = 0*pi/180;
alpha2_0 = 90*pi/180;
x0 = [alpha1_0; alpha2_0; 0; 0];

% Randbedingungen:
tspan = [0,10];

% Integration der Zustandgleichung:
[t, x] = ode45('beweggl_funktion',tspan,x0);

% Integrationsergebnisse lesen
alpha1_res = x(:,1);
alpha2_res = x(:,2);

% Ergebnisse (Trajektorien) zeichnen
figure(1);
plot(t,[alpha1_res alpha2_res]*180/pi);
title('Trajektorien')
xlabel('Zeit t /sec')
ylabel('Winkel \alpha_1 & \alpha_2 / Grad')

%% Animation der Ergebnisse
h_fig = figure(2); clf;

% Werte nochmals definieren
l1 = 1;
l2 = 1;

% Ortsvektoren
r1 = [ 0;
       l1*sin(alpha1_0);
      -l1*cos(alpha1_0)];
r2 = [ 0;
       l1*sin(alpha1_0)+l1*sin(alpha2_0);
      -l1*cos(alpha1_0)-l1*cos(alpha2_0)];

% Anfangskonfiguration zeichnen
plot([0 r1(2) r2(2)],[0 r1(3) r2(3)],'-o',...
    'linewidth',2,'markersize',10)
```

Command Window

New to MATLAB? Watch this

```
>> Doppelpendel_ZeitSi
>>
```