HW 9, matlab solution

Nathan Barton

November 6, 1999

1 Problem 3

The `matlab` files for the solution are given below. First is the output from the driver file, followed by the driver file itself and then the `m`-file `fan_solve.m` which does most of the work that is new in this part of the problem. The `m`-files are also available for download from the web page.

As in previous solutions, all angles are measured counterclockwise from the horizontal axis, and the `matlab` functions used in the solution take advantage of the systematic way in which the data structure for the problem has been created. Much of the code in this solution was given in the solution to the previous part of the fan problem.

In `fan_solve.m`, the force equations in each link are set up according to figure 1. Point S is the "start" of the link and point E is the "end" of the link. Force balance gives:

$$F_i^E\hat{\imath} + F_j^E\hat{\jmath} - F_i^S\hat{\imath} - F_j^S\hat{\jmath} = m(\overline{a}_g - \overline{g})$$

And moment balance about point S gives:

$$r\overline{n} \wedge \overline{F}^E + \text{(concentrated moments)} + mr_g\overline{n}_g \wedge \overline{g} = I_\alpha \alpha\hat{k} + mr_g\overline{n}_g \wedge \overline{a}_g$$

The system of equations set up in `fan_solve.m` follows this convention for all of the links.

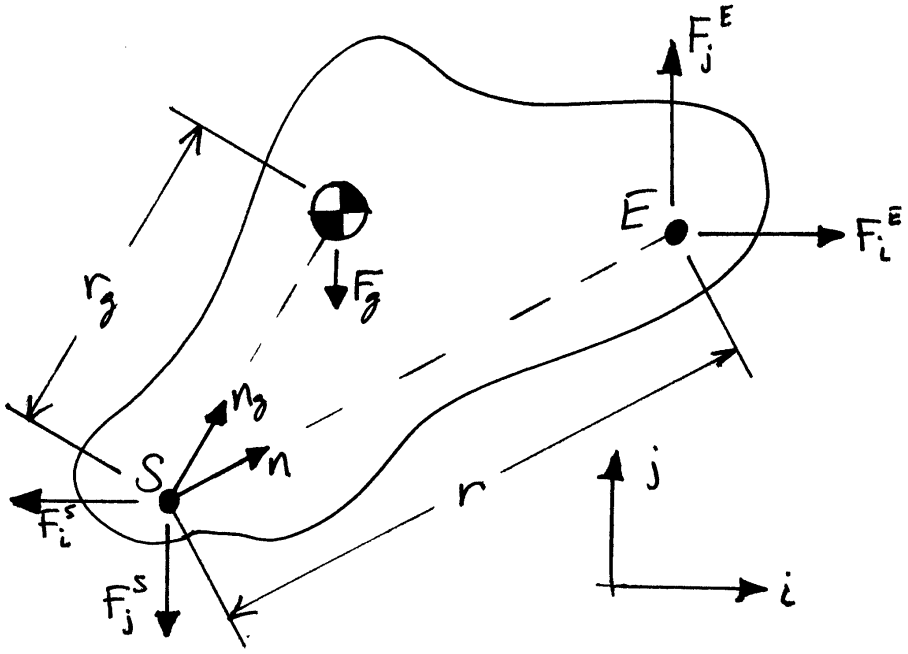

Figure 1: diagram for a general link

As stated in the problem description, there is an interesting value for the moment at the initial time. Because the links are assumed to be massless, the link connecting points B and C is a two-force body. The force in this body therefore acts along the line connecting points B and C. At the initial time, the link BC and the link AB align perfectly and thus at this instant the force at point B acting on link AB is aligned with link AB. That force then exerts no moment about point A. Doing moment balance about point A for link AB, we see that the only contribution comes from the moment at the motor, which must therefore be zero. The moment at the motor is *not* generally zero, and indeed you will calculate the value of the moment over a cycle as part of the next assignment.

1.1 output from driver file

```
>> fan_driver

th_init =

    0.6435
    0.6435
   -1.5708

forces at pins (points A,B,C,D in that order);
forces at pins are applied with the sign given below to the next link
and with opposite sign to the previous link, where the "next" link
is the one closer to the fan and the "previous" one is the link
closer to the motor
 horizontal component |   vertical component
          1.19979e+02 |          8.99841e+01
          1.19979e+02 |          8.99841e+01
          1.19979e+02 |          8.99841e+01
          1.19979e+02 |          2.23598e+02
the moment acting on the motor is          0.00000e+00
>>
```

1.2 Driver file

```
fan_driver.m


% some of the values that are set in this file are not used until later
% parts of the problem

global r r2 rg lthg;
global iwk iwu1 iwu2 ium;
global w_data;
global gc g_by_gc;
global m ai;

% figure counter
ifig = 0;
```

```
% some of geometry of linkage
%
% lengths of bars in linkage (meters)
r = 0.01*[10.0 40.0 30.0 40.0]';
r2 = r*r';
% angle (relative to local coordinates on link) to the center of mass;
% see m-file linkage_ag.m
lthg = [0.0 0.0 pi/6]';
% distance to centers of mass from starting point of link
rg = [r(1)*0.5 r(2)*0.5 r(3)/cos(lthg(3))]';

% masses, moments of inertia
m = [0.0 0.0 10.0]'; % in kg
% moment of inertia for uniform distribution of mass over links
ai = zeros(3,1);
ai(1) = m(1)*r2(1,1)/12.0;
ai(2) = m(2)*r2(2,2)/12.0;
diam_fan = 1.0;
ai(3) = m(3)*diam_fan*diam_fan/16.0;

% gravitation
g       = [0.0 -9.8]'; % direction 2 is down, in m/s^2
gc      = 1.0; % metric, trust mass to be in kg
g_by_gc = g/gc;

th1_init = atan2(3,4);
th_init = fan_angles(r,r2,th1_init)

% data for initial angular velocities
w_data = [2.0*pi 0.0 0.0]';
% indices for known (iwk) and unknown (iwu*) angular velocities
% and index for unknown moment (ium)
iwk = 1; iwu1 = 2; iwu2 = 3; ium = 1;

tstart = 0.0; tstop = 2.0*pi/abs(w_data(iwk)); nt = 101;
tspan = linspace(tstart, tstop, nt);
dtime = (tstop - tstart)/(nt-1);


options = odeset('AbsTol',1e-10);
[t, th_num] = ode45('fan_solve',tspan,th_init,options);



% obtain accelerations over a full cycle

ag_fan = zeros(nt,2);   mag_ag_fan = zeros(nt,1);
```

```
thddot_fan = zeros(nt); mag_thddot_fan = zeros(nt,1);
for it = 1:nt;
   [thdot,thddot,ag] = fan_solve(t(it), th_num(it,:));
   % store results at time it
   ag_fan(it,:) = ag(:,3)';
   mag_ag_fan(it) = norm(ag(:,3));
   thddot_fan(it) = thddot(3);
   mag_thddot_fan(it) = abs(thddot(3));
end;


[thdot,thddot,ag,f] = fan_solve(t(1), th_num(1,:));

disp('forces at pins (points A,B,C,D in that order);');
disp('forces at pins are applied with the sign given below to the next link');
disp('and with opposite sign to the previous link, where the "next" link ');
disp('is the one closer to the fan and the "previous" one is the link ');
disp('closer to the motor');
disp(sprintf(' %20s | %20s ','horizontal component','vertical component'));
if_base = 0;
for ipin = 1:length(r);
      disp(sprintf(' %20.5e | %20.5e ',f(if_base+1),f(if_base+2)));
      if_base = if_base + 2;
end;
disp(sprintf('the moment acting on the motor is %20.5e',f(if_base+1)));
```

1.3 m-files

```
fan_solve.m
```

```
function [thdot,thddot,ag,f] = fan_solve(t, th)
%
% given angles of components, find:
% thdot  -- angular velocities
% thddot -- angular accelerations
% ag     -- linear acceleration of mass centers
% f      -- forces at pins and unknown moment
%
global r r2 rg lthg;
global iwk iwu1 iwu2 ium;
global w_data;
global gc g_by_gc;
global m ai;
%
nlink = 3; % number of moving links
if (length(th) ~= nlink);
      disp('error, th of wrong length');
      return;
end;
%
% get unit vectors:
% vectors nvec are along the links and vectors tvec are perpendicular to
% the links and in the direction of motion of the end of the link when
% the angular velocity is positive
%
[nvec, tvec] = linkage_vec(th);
%
% solve for angular velocitiess
A = [r(iwu1)*tvec(:,iwu1) r(iwu2)*tvec(:,iwu2)];
y = [w_data(iwk)*r(iwk)*tvec(:,iwk)];
x = -A\y;
%
% set values in thdot
thdot = zeros(3,1);
thdot(iwk)  = w_data(iwk);
thdot(iwu1) = x(1);
thdot(iwu2) = x(2);
%
%
if (nargout > 1);
% find angular accelerations
%
% get squares of angular velocities
thdot2 = thdot.*thdot;
%
A = [r(iwu1)*tvec(:,iwu1) r(iwu2)*tvec(:,iwu2)];
y = [-thdot2(iwk)* r(iwk)* nvec(:,iwk)  + ...
     -thdot2(iwu1)*r(iwu1)*nvec(:,iwu1) + ...
     -thdot2(iwu2)*r(iwu2)*nvec(:,iwu2)];
x = -A\y;
%
% set values in thddot
thddot = zeros(3,1);
thddot(iwk)  = 0.0;
thddot(iwu1) = x(1);
thddot(iwu2) = x(2);
%
end; % end of calculation of angular accelerations
%
%
if (nargout > 2);
% find linear acceleration of mass centers
%
[ag,ng] = linkage_ag(thdot2,thddot,r,rg,lthg,nvec,tvec);
```

```
%
end; % end calculation of linear acceleration of mass centers
%
%
if (nargout > 3);
%
% find forces and moments
%
% code below works for a sequence of links with forces defined in the
% negative sense at the beginning of a link (connection to previous link)
% and in the position sense at the end of a link (connection to next link)
%
% two equations from force balance and one from moment balance for each link
neq_per_link = 2+1;
neq = nlink*neq_per_link;
%
% allow only one unknown moment; its index is imd
imd = (nlink+1)*2+1;
%
% zero necessary arrays
A = zeros(neq,neq);
b = zeros(neq,1);
f = zeros(neq,1);
%
% initialize equation counter
eq_num = 0;
%
for i_link = 1:nlink;
 %
 % equation numbers
 ieq = eq_num+1; jeq = eq_num+2; meq = eq_num+3;
 % increment equation counter
 eq_num = eq_num + neq_per_link;
 %
 % force indices for ends of links
 if_base = (i_link-1)*2;
 % forces at start of link
 ifsi = 1 + if_base; ifsj = 2 + if_base;
 if_base = if_base + 2;
 % forces at end of link
 ifei = 1 + if_base; ifej = 2 + if_base;
 %
 % force equations
 A(ieq,[ifsi, ifei]) = [-1.0 1.0];
 A(jeq,[ifsj, ifej]) = [-1.0 1.0];
 b([ieq,jeq]) = m(i_link)*(ag(:,i_link)/gc - g_by_gc(:));
 %
 % moment equation, about beginning of link
 ntemp = cross_2d_ij(nvec(:,i_link));
 A(meq,[ifei,ifej]) = r(i_link)*ntemp';
 if (ium == i_link);
   %this is link on which unknown moment is applied
   A(meq, imd) = 1.0;
 end;
 b(meq) = ai(i_link)*thddot(i_link)/gc + ...
+m(i_link)*rg(i_link)*cross_2d(ng(:,i_link),ag(:,i_link)) + ...
-m(i_link)*rg(i_link)*cross_2d(ng(:,i_link),g_by_gc);
 %
end; % end of loop over links
%
% finally, ready to solve
%
f = A\b;
%
end; % end of force and moment calculations
```

`cross_2d_ij.m`

```
function val=cross_2d_ij(n)
%
% if m = n ^ f .dot. k (where ^ denotes cross product and .dot. k means dot
%                       with the unit vector out of the plane)
% then val is set such that m = val(1)*f(1) + val(2)*f(2)
%
val = [-n(2) n(1)]';
```

For `fan_angles.m`, `linkage_vec.m`, `law_cos_len.m`, `linkage_ag.m` and `law_cos_ang.m`, see previous solutions.