Documentation for `doublePendulum`

doublePendulum v0.6 written by Alexander Erlich (`mailto:alexander.erlich@gmail.com`)
Last update submitted to MATLAB File Exchange on October 5th, 2010.

Lagrangian and differential equations

The program `double_pendulum.m` animates the double pendulum's (mostly) chaotic behavior.

The Lagrangian of the double pendulum as depicted in figure 1 is

$$\begin{aligned}\mathcal{L} &= \frac{m_1+m_2}{2}\, l_1^2\, \dot\varphi_1^2 + \frac{m_2}{2}\, l_2^2\, \dot\varphi_2^2 + m_2\, l_1\, l_2\, \dot\varphi_1\, \dot\varphi_2\, \cos(\varphi_1-\varphi_2) + \\ &\quad + (m_1+m_2)\, g\, l_1\, \cos\varphi_1 + m_2\, g\, l_2\, \cos\varphi_2\end{aligned}$$

The equations of motion can be derived using the Euler-Lagrange equations

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L}{\partial \dot\varphi_i} - \frac{\partial L}{\partial \varphi_i}\,, \qquad i = 1,\, 2$$

One obtains two ordinary differential equations of second order:

$$\begin{aligned}\ddot\varphi_1 + \frac{g}{l_1}\, \sin\varphi_1 + \frac{m_2}{m_1+m_2}\, \frac{l_2}{l_1}\, \left[\cos(\varphi_2-\varphi_1)\, \ddot\varphi_2 - \sin(\varphi_2-\varphi_1)\, \dot\varphi_2^2\right] &= 0 \\ \ddot\varphi_2 + \frac{g}{l_2}\, \sin\varphi_2 + \frac{l_1}{l_2}\, \left[\cos(\varphi_2-\varphi_1)\, \ddot\varphi_1 + \sin(\varphi_2-\varphi_1)\, \dot\varphi_1^2\right] &= 0\end{aligned}$$

It is now possible to rewrite this system of two second order ODEs into a system of four first order ODEs. Defining e.g. $x_1 = \varphi_1$ and $x_2 = \dot\varphi_1 = \dot x_1$ as well as $x_3 = \varphi_2$ and $x_4 = \dot\varphi_2 = \dot x_3$ and introducing a vector $\boldsymbol{x} = (x_1,\, x_2,\, x_3,\, x_4)^T$ one obtains a system $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x})$ of first order ODEs. The results of such a manipulation are presented in `double_pendulum_ODE.m`. A *Mathematica* notebook `double_pendulum_ODE_deduction.nb` containing all manipulations is also provided.

Running `double_pendulum`

The most simple way to run the program is `>> double_pendulum_init`. The parameters φ_1, $\dot\varphi_1$, φ_2, $\dot\varphi_2$, g, m_1, m_2, l_1 and l_2 can be adapted in the `double_pendulum_init.m` file. The parameters *duration*, *fps* and *movie* define the duration and framerate of the animation and whether the animation is supposed to be shown in realtime or rendered into a movie (.avi file).
Further and more technical information is given in the comments preceding the source code of the `m`-files.

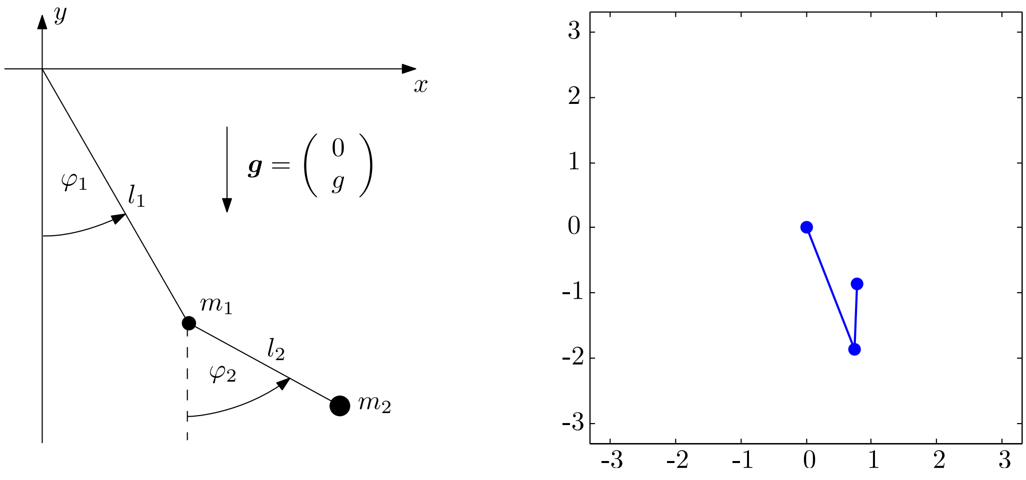

Figure 1: Scheme drawing (left) and Matlab figure (right)

Source code of `double_pendulum.m`

```
function double_pendulum(ivp, duration, fps, movie)
% DOUBLE_PENDULUM Animates the double pendulum's (mostly) chaotic behavior.
%
%   author:  Alexander Erlich (alexander.erlich@gmail.com)
%
%   parameters:
%
%   ivp=[phi1; dtphi1; phi2; dtphi2; g; m1; m2; l1; l2]
%
%                               Initial value problem. phi1 and dtphi1 are
%                               the initial angle and anglular velocity. g
%                               is gravity, m1 and l1 mass and rod length.
%                               For an explaining picture, see
%                               documentation file in same folder.
%
%   duration                    The time interval on which the ode is
%                               solved spans from 0 to duration (in sec).
%
%   fps                         Frames Per Second. The framerate is
%                               relevant both for normal (realtime)
%                               animation and movie recording.
%
%   movie                       If false, a normal realtime animation of
%                               the motion of the double pendulum (the
%                               framerate being fps) is shown.
%                               If true, a movie (.avi) is recorded. The
%                               filename is 'doublePendulumAnimation.avi'
%                               and the folder into which it is saved is
%                               the current working directory.
%
%   This function calls double_pendulum_ODE and is, in turn, called by
%   double_pendulum_init.
%
%   Example call:    >> double_pendulum([pi;0;pi;5;9.81;1;1;2;1],100,10,false)
%   Or, simply call  >> double_pendulum_init
%
%   -------------------------------------------------------------------

clear All; clf;

nframes=duration*fps;
sol=ode45(@double_pendulum_ODE,[0 duration], ivp);
t = linspace(0,duration,nframes);
y=deval(sol,t);

phi1=y(1,:)'; dtphi1=y(2,:)';
phi2=y(3,:)'; dtphi2=y(4,:)';
l1=ivp(8); l2=ivp(9);
% phi1=x(:,1); dtphi1=x(:,2);
% phi2=x(:,3); dtphi2=x(:,4);
% l1=ivp(8); l2=ivp(9);

h=plot(0,0,'MarkerSize',30,'Marker','.','LineWidth',2);
range=1.1*(l1+l2); axis([-range range -range range]); axis square;
set(gca,'nextplot','replacechildren');

    for i=1:length(phi1)-1
        if (ishandle(h)==1)
            Xcoord=[0,l1*sin(phi1(i)),l1*sin(phi1(i))+l2*sin(phi2(i))];
            Ycoord=[0,-l1*cos(phi1(i)),-l1*cos(phi1(i))-l2*cos(phi2(i))];
            set(h,'XData',Xcoord,'YData',Ycoord);
            drawnow;
```

```
        F(i) = getframe;
        if movie==false
            pause(t(i+1)-t(i));
        end
    end
end
if movie==true
    movie2avi(F,'doublePendulumAnimation.avi','compression','Cinepak','fps',fps)
end
```

Source code of double_pendulum_ODE.m

```
function xdot = double_pendulum_ODE(t,x)
% DOUBLE_PENDULUM_ODE Ordinary differential equations for double pendulum.
%
%   author:  Alexander Erlich (alexander.erlich@gmail.com)
%
%   parameters:
%
%   t       Column vector of time points
%   xdot    Solution array. Each row in xdot corresponds to the solution at a
%           time returned in the corresponding row of t.
%
%   This function calls is called by double_pendulum.
%
%   --------------------------------------------------------------------

g=x(5); m1=x(6); m2=x(7); l1=x(8); l2=x(9);

xdot=zeros(9,1);

xdot(1)=x(2);

xdot(2)=-((g*(2*m1+m2)*sin(x(1))+m2*(g*sin(x(1)-2*x(3))+2*(l2*x(4)^2+...
    l1*x(2)^2*cos(x(1)-x(3)))*sin(x(1)-x(3))))/...
    (2*l1*(m1+m2-m2*cos(x(1)-x(3))^2)));

xdot(3)=x(4);

xdot(4)=(((m1+m2)*(l1*x(2)^2+g*cos(x(1)))+l2*m2*x(4)^2*cos(x(1)-x(3)))*...
    sin(x(1)-x(3)))/(l2*(m1+m2-m2*cos(x(1)-x(3))^2));
```

Source code of double_pendulum_init.m

```
% Simply call
%
%       >> double_pendulum_init
%
% to run the double pendulum simulation with the below parameters. This
% script calls double_pendulum.
%
%   --------------------------------------------------------------------

phi1                = pi;
dtphi1              = 0;
phi2                = pi;
dtphi2              = 5;
g                   = 9.81;
m1                  = 1;
m2                  = 1;
l1                  = 2;
l2                  = 1;
duration            = 100;
fps                 = 10;
movie               = true;

clc; figure;

interval=[0, duration];
ivp=[phi1; dtphi1; phi2; dtphi2; g; m1; m2; l1; l2];

double_pendulum(ivp, duration, fps, movie);
```