Dynamics of Two Cylinders with Three Springs

Initialization Code (optional)

Manipulate

```
Manipulate[
 tick;
 Module[{sol, t, θ1sol, θ2sol, v1sol, v2sol, eq1, eq2, ic, θ1, θ2, result},

  If[state == "running" || state == "step" || state == "initial",
   eq1 = 3/2 r^2 m1 θ1''[t] + r^2 ((k1 + k2) θ1[t] - k2 θ2[t]) == 0;
   eq2 = 3/2 r^2 m2 θ2''[t] + r^2 (-k2 θ1[t] + (k2 + k3) θ2[t]) == 0;
   ic = {θ1[0] == θ1current, θ2[0] == θ2current, θ1'[0] == v1current, θ2'[0] == v2current};
   sol = First@NDSolve[{eq1, eq2, ic}, {θ1[t], θ2[t], θ1'[t], θ2'[t]}, {t, 0, stepSize}]
  ];

  If[state == "running" || state == "step",
   θ1current = θ1[t] /. sol /. t → stepSize;
   θ2current = θ2[t] /. sol /. t → stepSize;
   v1current = θ1'[t] /. sol /. t → stepSize;
   v2current = θ2'[t] /. sol /. t → stepSize;
   x1 = springRelaxedL + r θ1[t] /. sol /. t → 0;
   x2 = x1 + springRelaxedL + r θ2[t] /. sol /. t → 0;
   currentTime = currentTime + stepSize;
   If[state == "running",
    tick += del
   ];
  ];

  Text@drawSystem[currentTime, r, x1, x2,
    θ1current, θ2current, v1current, v2current, m1, m2, k1, k2, k3, showGrid]
 ],

 Text@Grid[{
    {
     Grid[{
       {
        Button[Text[Style["run", 11]], state = "running"; tick += del, ImageSize → {50, 35}],
        Button[Text[Style["pause", 11]], state = "paused"; tick += del, ImageSize → {50, 35}]
       },
       {Button[Text[Style["step", 11]], state = "step"; tick += del, ImageSize → {50, 35}],
        Button[Text[Style["reset", 11]], state = "initial";
         tick = 0; θ10 = 0; θ1current = 0; θ20 = 0; θ2current = 0; v1current = 40.0 * (2 Pi) / 60; v10 = 40.0;
         v2current = 0; v20 = 0; currentTime = 0; k1 = 100; k2 = 200;
         k3 = 300; m1 = 6; m2 = 1; x1 = springRelaxedL; x2 = 2 springRelaxedL; tick += del,
         ImageSize → {50, 35}
        ]
```

```
      }
     }, Spacings → {0.4, .2}, Alignment → Center
    ]
   },
   {
    Grid[{
      {
       "step size", Manipulator[Dynamic[stepSize, {stepSize = #} &], {0.001, 0.02, 0.001},
        ImageSize → Tiny, ContinuousAction → True], Dynamic[padIt2[stepSize, {4, 3}]], "sec"
      },
      {Row[{"θ"1 ', "(0)"}], Manipulator[Dynamic[v10, {v10 = #; v1current = v10 * 2 Pi / 60} &], {0, 50, .1},
        ImageSize → Tiny, ContinuousAction → True], Dynamic[padIt2[v10, {2, 1}]], "rpm"
      },
      {Row[{"θ"2 ', "(0)"}], Manipulator[Dynamic[v20 , {v20 = #; v2current = v20 * 2 Pi / 60} &], {0, 50, .1},
        ImageSize → Tiny, ContinuousAction → True], Dynamic[padIt2[v20, {2, 1}]], "rpm"
      },
      {Text@TraditionalForm@Style[k1, 11], Manipulator[Dynamic[k1 , {k1 = #; tick += del} &],
        {100, 500, 1}, ImageSize → Tiny, ContinuousAction → True], Dynamic[padIt2[k1, 3]], "N/m"
      },
      {TraditionalForm@Style[k2, 11], Manipulator[Dynamic[k2 , {k2 = #; tick += del} &],
        {100, 500, 1}, ImageSize → Tiny, ContinuousAction → True], Dynamic[padIt2[k2, 3]], "N/m"
      },
      {TraditionalForm@Style[k3, 11], Manipulator[Dynamic[k3 , {k3 = #; tick += del} &],
        {100, 500, 1}, ImageSize → Tiny, ContinuousAction → True], Dynamic[padIt2[k3, 3]], "N/m"
      },
      {TraditionalForm@Style[m1, 11], Manipulator[Dynamic[m1 , {m1 = #; tick += del} &],
        {.1, 10, .1}, ImageSize → Tiny, ContinuousAction → True], Dynamic[padIt2[m1, {3, 1}]], "kg"
      },
      {TraditionalForm@Style[m2, 11], Manipulator[Dynamic[m2 , {m2 = #; tick += del} &],
        {.1, 10, .1}, ImageSize → Tiny, ContinuousAction → True], Dynamic[padIt2[m2, {3, 1}]], "kg"
      }
     }, Frame → True, FrameStyle -> Directive[Thickness[.001], Gray], Alignment → Center, Spacings → {.4, .1}
    ]
   }
  }],

Row[{"show grid", Spacer[4], Checkbox[Dynamic[showGrid, {showGrid = #; tick += del} &]]}],

{{showGrid, True}, None},
{{springRelaxedL, 2}, None},
{{x1, 2}, None}, (*keep same as springRelaxedL*)
{{x2, 4}, None}, (*keep same as 2*springRelaxedL*)
{{r, .5}, None},
{{state, "initial"}, None},
{{stepSize, 0.015}, None},
{{m1, 6}, None},
{{m2, 1}, None},
{{k1, 100}, None},
{{k3, 300}, None},
{{k2, 200}, None},
{{θ10, 0}, None}, (*initial radial position m1*)
{{θ20, 0}, None}, (*initial radial position m2*)
{{v10, 40}, None}, (*initial angular position RPM, m2*)
{{v20, 0}, None}, (*initial angula position RPM, m2*)

{{θ1current, 0}, None}, (*current radial position m1*)
{{θ2current, 0}, None}, (*current radial position m2*)
{{v1current, 40 (2 Pi / 60)}, None}, (*current angular position, rad/sec m2*)
{{v2current, 0 (2 Pi / 60)}, None}, (*current angula position rad/sec m2*)

{{currentTime, 0}, None}, (*simulation time*)
{{del, $MachineEpsilon}, None},
{{tick, 0}, None},
```

```
ControlPlacement → Left,
SynchronousUpdating → False,
SynchronousInitialization → False,
ContinuousAction → False,
Alignment -> Center,
ImageMargins → 0,
FrameMargins → 0,
Paneled → True,
Frame → False,
AutorunSequencing → {1},

TrackedSymbols :→ {tick},
Initialization :→
 {
  integerStrictPositive = (IntegerQ[#] && # > 0 &);
  integerPositive = (IntegerQ[#] && # ≥ 0 &);
  numericStrictPositive = (Element[#, Reals] && # > 0 &);
  numericPositive = (Element[#, Reals] && # ≥ 0 &);
  numericStrictNegative = (Element[#, Reals] && # < 0 &);
  numericNegative = (Element[#, Reals] && # ≤ 0 &);
  bool = (Element[#, Booleans] &);
  numeric = (Element[#, Reals] &);
  integer = (Element[#, Integers] &);

  (*--------------------------------------------*)
  (* helper function for formatting             *)
  (*--------------------------------------------*)
  padIt1[v_?numeric, f_List] :=
   AccountingForm[Chop[v] , f, NumberSigns → {"-", "+"}, NumberPadding → {"0", "0"}, SignPadding → True];
  (*--------------------------------------------*)
  (* helper function for formatting             *)
  (*--------------------------------------------*)
  padIt2[v_?numeric, f_List] :=
   AccountingForm[Chop[v] , f, NumberSigns → {"", ""}, NumberPadding → {"0", "0"}, SignPadding → True];
  padIt2[v_?numeric, f_Integer] := AccountingForm[Chop[v] , f, NumberSigns → {"", ""},
    NumberPadding → {"0", "0"}, SignPadding → True];

  (*--------------------------------------------*)
  getKE[m1_, m2_, v1_, v2_] :=
   Module[{i1 = 1/2 m1 r^2, i2 = 1/2 m2 r^2}, 1/2 i1 (v1)^2 + 1/2 m1 (r v1)^2 + 1/2 i2 (v2)^2 + 1/2 m2 (r v2)^2];
  getPE[θ1_, θ2_, k1_, k2_, k3_] := 1/2 k1 (r θ1)^2 + 1/2 k2 (r θ2 - r θ1)^2 + 1/2 k3 (r θ2)^2;
  (*--------------------------------------------*)
  drawSystem[currentTime_,
    r_(*radius of sphere*),
    x1_(*distance of center of 1st cylinder from left wall*),
    x2_(*distance of center of 2nd cylinder from left wall*),
    θ1sol_(*angle of rotation of first cyliner*),
    θ2sol_(*angle of rotation of second cyliner*),
    v1sol_(*anglular velocity of first cyliner*),
    v2sol_(*anglular velocity of first cyliner*),
    m1_, m2_, k1_, k2_, k3_, showGrid_] := Module[{ke, pe, h1},
    ke = getKE[m1, m2, v1sol, v2sol];
    pe = getPE[θ1sol, θ2sol, k1, k2, k3];

    h1 = Text@Style[Grid[{
         {"time (sec)",
          "P.E. (J)",
          "K.E. (J)",
          "energy (J)",
```

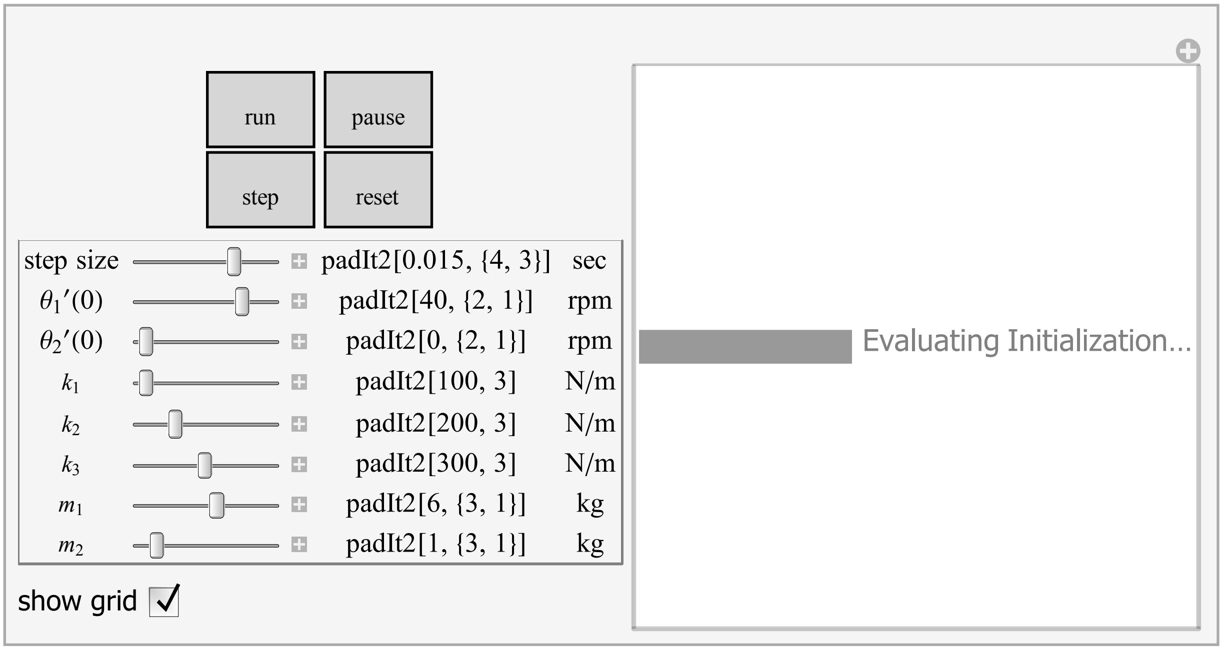
```
      Row[{"θ"1', " (rpm)"}],
      Row[{"θ"2', " (rpm)"}]
     },
     {padIt2[currentTime, {6, 2}],
      padIt2[pe, {6, 3}],
      padIt2[ke, {6, 3}],
      padIt2[ke + pe, {6, 3}],
      padIt2[v1sol * 60 / (2 Pi), {6, 3}],
      padIt2[v2sol * 60 / (2 Pi), {6, 3}]}
    },
    Frame → All,
    FrameStyle -> Directive[Thickness[.001], Gray],
    Spacings → {1, 1.2},
    Alignment → Center]
   , 12];

Grid[{
  {
   h1
  },
  {
   Graphics[{
     {AbsoluteThickness[k1/250], Line@makeSpring[0, r, x1 - r, r, r / 2]},
     {EdgeForm[Black], Opacity[m1 / 10], Red, Disk[{x1, r}, r]},
     {Black, Line[{{x1, r}, {x1 + r Sin[θ1sol], r + r Cos[θ1sol]}}]},
     {AbsoluteThickness[k2/250], Line@makeSpring[x1 + r, r, x2 - r, r, r / 2]},
     {EdgeForm[Black], Opacity[m2 / 10], Red, Disk[{x2, r}, r]},
     {Black, Line[{{x2, r}, {x2 + r Sin[θ2sol], r + r Cos[θ2sol]}}]},
     {AbsoluteThickness[k3/250], Line@makeSpring[x2 + r, r, 3 springRelaxedL, r, r / 2]}
    },
    PlotRange → {{0, 3 springRelaxedL}, {-0.5 r, 3 r}},
    ImageSize → 400,
    Axes → True,
    Frame → True,
    AxesOrigin → {0, 0},
    AspectRatio → Automatic,
    GridLines → If[showGrid, {Range[0, 3 springRelaxedL, .5], Automatic}, None],
    GridLinesStyle → LightGray
   ]
  }
 }, Alignment → Center, Spacings → {.1, .2}
]
];
(*---------------------------------------------*)
makeSpring[xFirst_?numeric, yFirst_?numeric, xEnd_?numeric, yEnd_?numeric, szel_?numeric] :=
 Module[{hx, veghossz, hossz, hy, dh, tbl},
  hx = xEnd - xFirst;
  If[Abs[hx] ≤ $MachineEpsilon, hx = 10^-6];
  hy = yEnd - yFirst;
  If[Abs[hy] ≤ $MachineEpsilon, hy = 10^-6];

  veghossz = 0.03;
  hossz = Sqrt[hx^2 + hy^2];
  dh = (hossz - 2 * veghossz) / 20;
  tbl = Table[If[OddQ[i], {xFirst + hx * (i * dh + veghossz) / hossz + hy * szel / hossz,
```

```
        yFirst + hy * (i * dh + veghossz) / hossz - hx * szel / hossz}, {xFirst + hx * (i * dh + veghossz) / hossz -
         hy * szel / hossz, yFirst + hy * (i * dh + veghossz) / hossz + hx * szel / hossz}], {i, 2, 18}];
    {{xFirst, yFirst}} ~ Join ~ {{xFirst + hx * (dh + veghossz) / hossz, yFirst + hy * (dh + veghossz) / hossz}} ~
     Join ~ tbl ~ Join ~ {{xFirst + hx * (19 * dh + veghossz) / hossz, yFirst + hy * (19 * dh + veghossz) / hossz}} ~
     Join ~ {{xEnd, yEnd}}
   ];

  }
]
```



Caption

This Demonstration determines the solution of a discrete system of two cylinders connected by three springs. You can adjust the mass of each cylinder and the stiffness of each spring to see the effect on the dynamics of the system. The left and right cylinders have masses m_1 and m_2, respectively. The spring stiffness values are k_1, k_2, and k_3 from left to right. You can adjust the initial angular velocity of either cylinder. The initial angular position is assumed to be zero. The equations of motion are derived using the Lagrangian method. The author
s report contains the mathematical derivation of the equations of motion. Cylinders roll without slipping and the system has two degrees of freedom $\theta_1(t)$ and $\theta_2(t)$. The system potential energy, kinetic energy, and total energy are displayed in the table at the top. Since zero friction and damping are assumed, the total energy remains constant.

- [ ] Create and Delete Locators
- [x] Slider Zoom
- [x] Gamepad Controls
- [x] Automatic Animation
- [ ] Bookmark Animation

Search Terms (optional)

mechanics
vibration
Lagrangian method

Related Links (optional)

Simple Harmonic Motion

Authoring Information

Contributed by: Nasser M. Abbasi