■ A simulation of Delta Robot

Willy Svenningsson 2010-09-23
willy.svenningsson@f ager .st

■ Init

In[1]:=
```
enVertCircel[where_, radie_] :=
 Module[{},
  Line@Table[{where[[1]] + radie * Cos[v],
     where[[2]], where[[3]] + radie * Sin[v]}, {v, 0, 2 Pi, Pi / 12}]
 ]
```

In[2]:=
```
showP[pkts_List] :=
 Map[(
    showP[#[[1]], #[[2]]]
   ) &, pkts]
showP[name_, where_] :=
 Module[{},
  {Point[where], Text[name, where, {-1, 0}]}
 ]
```

In[4]:=
```
vridPunkt[vinkel_] := (Cos@vinkel  -Sin@vinkel
                       Sin@vinkel   Cos@vinkel )
```

In[5]:=
```
kalkSnittPunkter[radieCentrum_, cPunkten_, radieCPunkten_] :=
 Module[{dist, x, y, vridVinkel, cx, cy, x1, y1, x2, y2},
  {cx, cy} = cPunkten;
  dist = EuclideanDistance[{0, 0}, cPunkten] // N;
  x = (dist^2 - radieCPunkten^2 + radieCentrum^2) / (2 dist) // N;
  y =
   1/(2 dist) Sqrt[4 dist^2 radieCentrum^2 - (dist^2 - radieCPunkten^2 + radieCentrum^2)^2] // N;
  vridVinkel = ArcTan[cx, cy];
  {x1, y1} = {x, y}.vridPunkt[-vridVinkel] // N;
  {x2, y2} = {x, -y}.vridPunkt[-vridVinkel] // N;
  {{x1, y1}, {x2, y2}}
 ]
```

In[6]:=

```
figAsBox[start_, dims_] :=
 Module[{len, width, height, dx, dy,
   dz, radie, offset, endx, endy, endz, name, endoffset},
  {len, width, height, {dx, dy, dz}, radie, {endx, endy, endz}, name} = dims;
  offset = {dx, dy, dz};
  endoffset = {endx, endy, endz};
  Cuboid[start + offset + {0, -width / 2, -height / 2},
   start + {len, width / 2, height / 2} + endoffset]
 ]
```

In[7]:=

```
figAsCyl[start_, dims_] :=
 Module[{len, width, height, dx, dy,
   dz, radie, offset, endx, endy, endz, name, endoffset},
  {len, width, height, {dx, dy, dz}, radie, {endx, endy, endz}, name} = dims;
  offset = {dx, dy, dz};
  endoffset = {endx, endy, endz};
  Cylinder[{start + offset, start + {0, len, 0} + endoffset}, radie]
 ]
```

■ Run

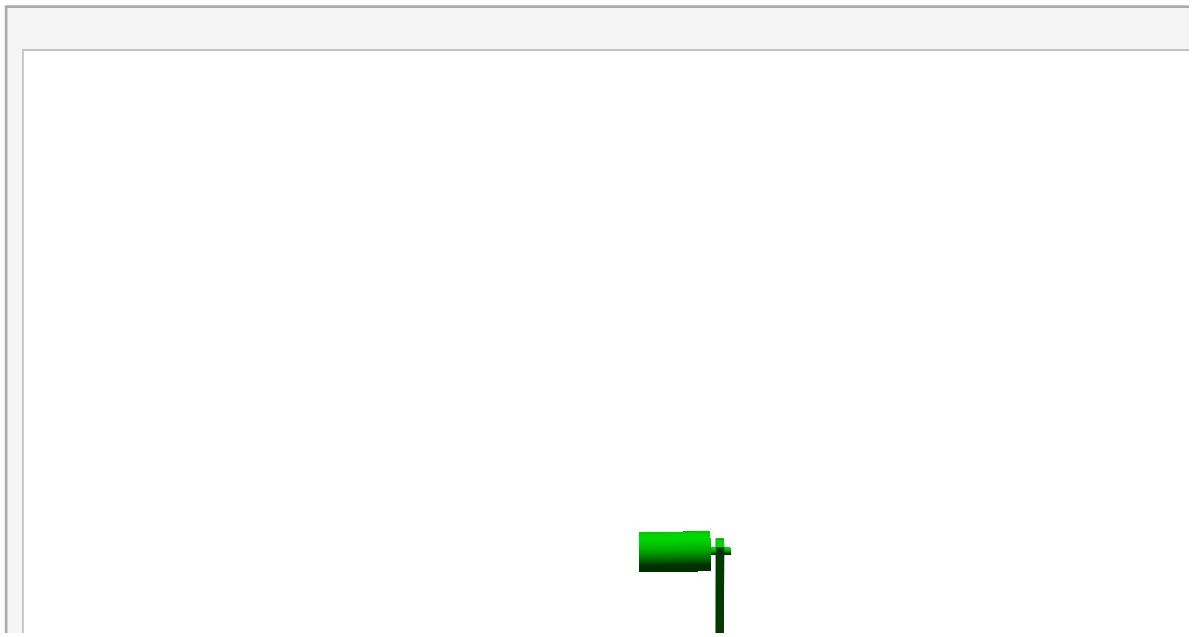
In[8]:=

```
Module[
 {jobbRadie, radieArm, tjockLillaArm,
  p0,
  golvetZ, gGolvet,
  dimfig1, gFig1,
  dimfig2, gFig2,
  dimfig3, gFig3,
  dimfig4, gFig4,
  gFig51, gFig52, gFig61, gFig62, gFig7, gFig81, gFig82,
  gFig3Static, gFig4Static, gFig5Static,

  gArm,
  p1, p2, p2x, p2y, p2z, p3, p3x, p3y, p3z,
  dt, triangel, gTriangel, movedTriangel,
  radieSnittCirkel, circSnitt, px11, pz11, px12, pz12, p11, p12,

  toShow, machineNumber, allMachines, allAlfa, finalPicture,
  oldpx, oldpy, oldpz, oldlenLongArm, oldradieArm,
  demoPoints, demoPointNow, figDemoPoints
 },
 demoPoints =
  Table[{r Sin[t], r Cos[t], r / 1 Sin[t] - 1.5}, {t, 0, 2 Pi, Pi / 32}] /. r -> .5;
 figDemoPoints = Line@demoPoints;
 demoPointNow = 1;
 golvetZ = -1.7;
 jobbRadie = 0.99;
 p0 = {0, 0, 0};
 (*    box len,    width, height, offset origo, radie, end offset*)
 dimfig1 = {0.4,      .2,    .2,   {0, .05, 0},       0.1, {0, 0, 0}, "motor"};
 dimfig2 =
  {.2,      .2,    .2,   {0, -.05, 0},       0.02, {0, 0, 0}, "motoraxel"};
 dimfig3 = {-0.20,    .04,  .07,   {-.05, 0, 0},
   0.02, {.05, 0, 0}, "motorarm"};
 dimfig4 = {0.20,     .04,  .07,   {0, -0.20 / 2, 0},
   0.01, {0, -0.20 / 2, 0}, "lagerarm"};


 tjockLillaArm = .02;


 gFig1 = figAsCyl[p0, dimfig1];
 gFig2 = figAsCyl[p0, dimfig2];

 gFig3Static = figAsBox[p0, dimfig3];
 gFig4Static = figAsCyl[p0, dimfig4];
 gFig5Static = Sphere[p0, 3 * dimfig4[[5]]];

 dt = .2;
 triangel =
  {{dt, 0, 0}, {-dt * 1 / 2, dt * Sqrt[3] / 2, 0}, {-dt * 1 / 2, -dt * Sqrt[3] / 2, 0}} // N;
```

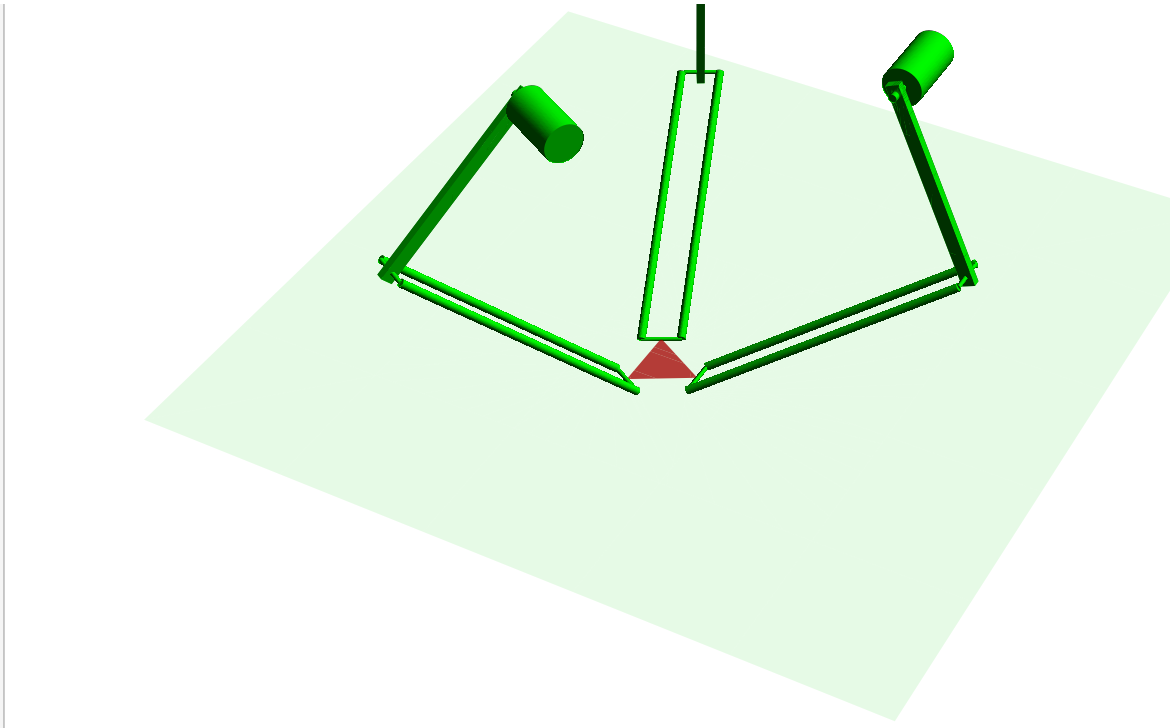
```
allMachines = {{}, {}, {}};
allAlfa = {{}, {}, {}};
gGolvet = Polygon[
  {{2, 2, golvetZ}, {2, -2, golvetZ}, {-2, -2, golvetZ}, {-2, 2, golvetZ}}];

{oldpx, oldpy, oldpz} = {Infinity, Infinity, Infinity};
oldlenLongArm = Infinity;
oldradieArm = Infinity;
oldlenLongArm = Infinity;
Manipulate
 [
 If[v1Scenario,
  If[demoPointNow == Length@demoPoints - 1, demoPointNow = 0];
  ++demoPointNow;
  {px, py, pz} = demoPoints[[demoPointNow]];
  (*px += .05;*)
  ,
  demoPointNow = 1];
 If[{oldpx, oldpy, oldpz} != {px, py, pz},
  movedTriangel = Map[(# + {px, py, pz}) &, triangel];
  {oldpx, oldpy, oldpz} = {px, py, pz};
  triangelMoved = True,
  triangelMoved = False
 ];
 radieArm = rArm;
 (*dimfig3[[1]]=rArm - dimfig3[[4,1]];*)
 If[oldradieArm != radieArm,
  dimfig3[[1]] = rArm ;
  gFig3Static = figAsBox[p0, dimfig3];
  oldradieArm = rArm;
 ];


 Do[
  (*If[triangelMoved,*)
  {p2x, p2y, p2z} = movedTriangel[[machineNumber]];
  transform = TranslationTransform[{-jobbRadie, 0, 0}];

  Switch[machineNumber,
   1,
   {p2x, p2y, p2z} = transform[{p2x, p2y, p2z}];
   p2 = {p2x, p2y, p2z},
   2,
   rotate = RotationTransform[-120 Degree // N, {0, 0, 1}];
   {p2x, p2y, p2z} = rotate[{p2x, p2y, p2z}];
   {p2x, p2y, p2z} = transform[{p2x, p2y, p2z}];
   p2 = {p2x, p2y, p2z},
   3,
   rotate = RotationTransform[+120 Degree // N, {0, 0, 1}];
   {p2x, p2y, p2z} = rotate[{p2x, p2y, p2z}];
   {p2x, p2y, p2z} = transform[{p2x, p2y, p2z}];
   p2 = {p2x, p2y, p2z}
  ];
  (*];*)
```

```
(*p2={p2x,p2y,p2z};*)

{p3x, p3y, p3z} = p2;
p3y = 0;
p3 = {p3x, p3y, p3z};

radieSnittCirkel = Sqrt[lenLongArm^2 - p2y^2] // N;
radieArm = rArm;


{{px11, pz11}, {px12, pz12}} =
 kalkSnittPunkter[radieArm, {p3x, p3z}, radieSnittCirkel];
p11 = {px11, 0, pz11}; p12 = {px12, 0, pz12};

allAlfa[[machineNumber]] = -ArcTan[px11, pz11] // N;
p1 = {Cos[allAlfa[[machineNumber]]] * radieArm,
    0, -Sin[allAlfa[[machineNumber]]] * radieArm} // N;

gFig3 = Rotate[gFig3Static, allAlfa[[machineNumber]], {0, 1, 0}];
gFig4 = Translate[gFig4Static, p1];
gFig51 = Translate[gFig5Static, p1 + {0, -dimfig4[[1]] / 2, 0}];
gFig52 = Translate[gFig5Static, p1 + {0, dimfig4[[1]] / 2, 0}];
gFig61 = Cylinder[{p1 + {0, -dimfig4[[1]] / 2, 0},
    p2 + {0, -dimfig4[[1]] / 2, 0}}, tjockLillaArm];
gFig62 = Cylinder[{p1 + {0, +dimfig4[[1]] / 2, 0},
    p2 + {0, +dimfig4[[1]] / 2, 0}}, tjockLillaArm];
gFig7 = Translate[gFig4Static, p2];
gFig81 = Translate[gFig5Static, p2 + {0, -dimfig4[[1]] / 2, 0}];
gFig82 = Translate[gFig5Static, p2 + {0, dimfig4[[1]] / 2, 0}];

toShow = {gFig1, gFig2, gFig3, gFig4,
  gFig51, gFig52, gFig61, gFig62, gFig7, gFig81, gFig82};

toShow = {gFig1, gFig2, gFig3, gFig4, gFig61, gFig62, gFig7};
(* Sphere och transform is really heavy *)
(*toShow={gFig51,gFig52,gFig81,gFig82};*)
Switch[machineNumber,
 1,
 alfa1 = allAlfa[[1]];
 allMachines[[1]] = Map[Translate[#, {jobbRadie, 0, 0}] &, toShow],
 2,
 alfa2 = allAlfa[[2]];
 allMachines[[2]] = Map[(Rotate[#, 120 Degree, {0, 0, 1}]) &,
   Map[Translate[#, {jobbRadie, 0, 0}] &, toShow]],
 3,
 alfa3 = allAlfa[[3]];
 allMachines[[3]] = Map[(Rotate[#, -120 Degree, {0, 0, 1}]) &,
   Map[Translate[#, {jobbRadie, 0, 0}] &, toShow]];
];
, {machineNumber, 1, 3}];

finalPicture = Map[(Graphics3D@#) &, allMachines];

Show[
 {
```

```
    Graphics3D@Green,
    finalPicture,
    Graphics3D@Opacity[0.1],
    Graphics3D@gGolvet,
    Graphics3D@Red,
    Graphics3D@Opacity[0.76],
    Graphics3D@{Polygon[movedTriangel]}
    , If[v1Scenario, Graphics3D@figDemoPoints, Graphics3D@Red]
   },
   ImageSize -> {500, 500},
   Axes -> False, Boxed -> False,
   AxesOrigin -> {0, 0, 0}, SphericalRegion -> True,
   PlotRange -> {{-2, 2}, {-2, 2}, {-2, 1}}
  ],

  {{p1, p2, p3}, ControlType -> None},
  {{rArm, 1, "Lenght upper joint"}, 0, 2},
  {{px, 0, "Desired x"}, -1, 1},
  {{py, 0, "Desired y"}, -1, 1},
  {{pz, -1.7, "Desired z"}, -2, 0},
  {{lenLongArm, 1.5, "Length of lower joint"}, 0.5, 2},
  {{alfa1, 30 Degree, "Engine axis 1 rotation (Read only)"},
   -360 Degree, 360 Degree},
  {{alfa2, 30 Degree, "Engine axis 2 rotation(Read only)"},
   -360 Degree, 360 Degree},
  {{alfa3, 30 Degree, "Engine axis 3 rotation(Read only)"},
   -360 Degree, 360 Degree},
  Delimiter,
  {{v1Scenario, False, "Run scenario"}, {True, False}},
  TrackedSymbols -> {rArm, px, py, pz, lenLongArm, v1Scenario},
  SynchronousUpdating -> True,

  ControlPlacement → Bottom
 ]
]
```



Out[8]=



Lenght upper joint

Desired x

Desired y

Desired z

Length of lower joint

Engine axis 1 rotation (Read only)

Engine axis 2 rotation(Read only)

Engine axis 3 rotation(Read only)

Run scenario