Universal method of kinematic analysis of spatial and planar link mechanisms

Alexey Ivanov
stitch_design@rambler.ru

This application demonstrates a universal method for calculating link mechanisms. It is based on the Draghilev's method for solving systems of nonlinear equations.
When calculating link mechanisms we can use geometrical relationships to produce their mathematical models without specifying the "input link". The new method allows us to specify the "input link", any link of mechanism.

Example: Three-bar Mechanism

The system of equations linkages in this mechanism is as follows:

```
f1 := x1^2+(x2+1)^2+(x3-.5)^2 - r^2;
f2 := x1 -.5*x2+.5*x3;
f3 := (x1-x4)^2+(x2-x5)^2+(x3-x6)^2 - 19;
f4 := sin(x4)-x5;
f5 := sin(2*x4)-x6;
```

Coordinates green point x'i', i = 1..3, the coordinates of red point x'i', i = 4..6.
f1, f2 trajectory of the green point.
f4, f5 trajectory of the red point.
f3 is the distance between these points.
Set of x0'i', i = 1..6 searched arbitrarily, is the solution of the system of equations and is the initial point for the solution of the ODE system. The solution of ODE system is the solution of system of equations linkages for concrete assembly linkage.

The example shown below is for one mechanism, but the code can be modified for two cases. In 1 case, the "input link" is the blue-green, 2 case the "input link" is the red-green. For the case of 2 you need to remove the "#" in the areas of the program highlighted in blue. After the calculation trajectories of points, we can always find the values of other variables, for example, the angles.

Detailed description of the method and two examples can be found in "Calculation method of linkages.pdf", available as part of this download.

Code

```
> restart : with(LinearAlgebra) : with(plots) :

  r := 3 : h := 0.025 : N := 120 : smin := 0 : smax := 2·3.05·r :
```

```
#r:=3: h:=0.025: N:=120: smin := 0: smax := 2.07*r:


f1 := (x1)^2 + (x2 + 1)^2 + (x3 - 0.5)^2 - r^2 :
f2 := x1 - 0.5 x2 + 0.5 x3 :
f3 := (x1 - x4)^2 + (x2 - x5)^2 + (x3 - x6)^2 - 19 :
f4 := sin(x4) - x5 :
f5 := sin(2 x4) - x6 :

x01 := 1.18325;
 x02 := 1.18325;
  x03 := -1.18325;
   x04 := -2.07809;
    x05 := -0.87406;
     x06 := 0.84926;

  n := 5 :
 x := cat(x, 1 ..n + 1) :
 x0 := cat(x0, 1 ..n + 1) :
 F := [seq(unapply(eval(cat('f', i)), [x]), i = 1 .. n)] :
A := MTM:-jacobian(F(x), [x]) :
A := MTM:-subs(A, [x], [x(s)]) :

 for i to n do  b[i] := simplify(Determinant(DeleteColumn(ColumnOperation(A, [i, n + 1]),
     n + 1))) od:
            b[n + 1] := simplify(-Determinant(DeleteColumn(A, n + 1))) :
           deqs := seq(diff(x[i](s), s) = b[i] / (b[1]^2 + b[2]^2 + b[3]^2)^0.5, i = 1..n + 1):
          # deqs := seq(diff(x[i](s), s) = b[i] / (b[4]^2 + b[5]^2 + b[6]^2)^0.5, i = 1..n + 1):

ics := seq(x[i](0) = x0[i], i = 1 .. n + 1) :
sln := dsolve([deqs, ics], numeric, method = rkf45, abserr = h/100, maxfun = 10000, range
    = smin .. smax) :
 a1 := sln(smin + smax*nn/N)[2] :
   a2 := sln(smin + smax*nn/N)[3] :
    a3 := sln(smin + smax*nn/N)[4] :
      a4 := sln(smin + smax*nn/N)[5] :
        a5 := sln(smin + smax*nn/N)[6] :
          a6 := sln(smin + smax*nn/N)[7] :


 AB := seq(plottools[line]([rhs(a1), rhs(a2), rhs(a3)], [rhs(a4), rhs(a5), rhs(a6)],
      color = black, thickness = 2), nn = 0 .. N) :
 AO := seq(plottools[line]([rhs(a1), rhs(a2), rhs(a3)], [-0.5, -0.75, 0.25], color
     = black, thickness = 2), nn = 0 .. N) :
LL := seq(pointplot3d([rhs(a1), rhs(a2), rhs(a3)], symbolsize = 16, symbol = solidsphere ,
      color = green), nn = 0 .. N) :
```

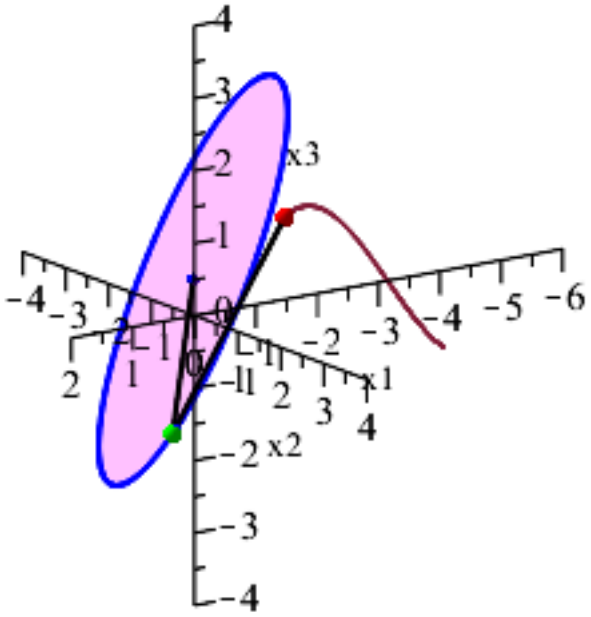
```
LLL := seq(pointplot3d([rhs(a4), rhs(a5), rhs(a6)], symbolsize = 16, symbol
    = solidsphere , color = red), nn = 0 .. N) :


P := pointplot3d([-0.5, -0.75 , 0.25], symbolsize = 12, symbol = solidsphere, color = blue) :


B := rhs~(Matrix(sln~([seq(i, i = smin .. smax + 20 h, h)]))[ .. , 1 .. n + 2]) :
# B := rhs~(Matrix(sln~([seq(i, i = smin .. smax + h, 0.1 · h)]))[ .. , 1 .. n + 2]) :

m := ArrayNumElems(B[ .., 1]); m;
  k := 1 :
for j to n + 1 do L[j] := B[ .., k + j] : od:
  S := [seq(seq(L[j][i], j = 1 ..3), i = 1 .. m)] :
 SS := [seq(seq(L[j][i], j = 4 ..6), i = 1 .. m)] :
FACE := plottools[polygon]([seq([L[1][i], L[2][i], L[3][i]], i = 1 ..m)], color = RGB(12,
     0.1, 5), transparency = 0.7, style = patchnogrid) :
 MS := pointplot3d(S, color = blue, style = line, thickness = 2, axes = normal, labels = ["x1",
    " x2", " x3"]) :
 MSS := pointplot3d(SS, color = RGB(0.5, 0.1, 0.21), style = line, thickness = 1) :
E := seq(display(MSS, MS, AB[k], AO[k], P, FACE, LL[k], LLL[k]), k = 1 .. N) :
display(E, insequence = true, view = [-6 ..2, -4 ..4, -4 ..4], axes = normal);
```

$$x01 := 1.18325$$
$$x02 := 1.18325$$
$$x03 := -1.18325$$
$$x04 := -2.07809$$
$$x05 := -0.87406$$
$$x06 := 0.84926$$
$$m := 753$$
$$753$$


4
3
2
x3
1
-4
-3
2
1
0
-1
2
3
4
-2
-3
-4
-5
-6
x1
-2
x2
-3
-4