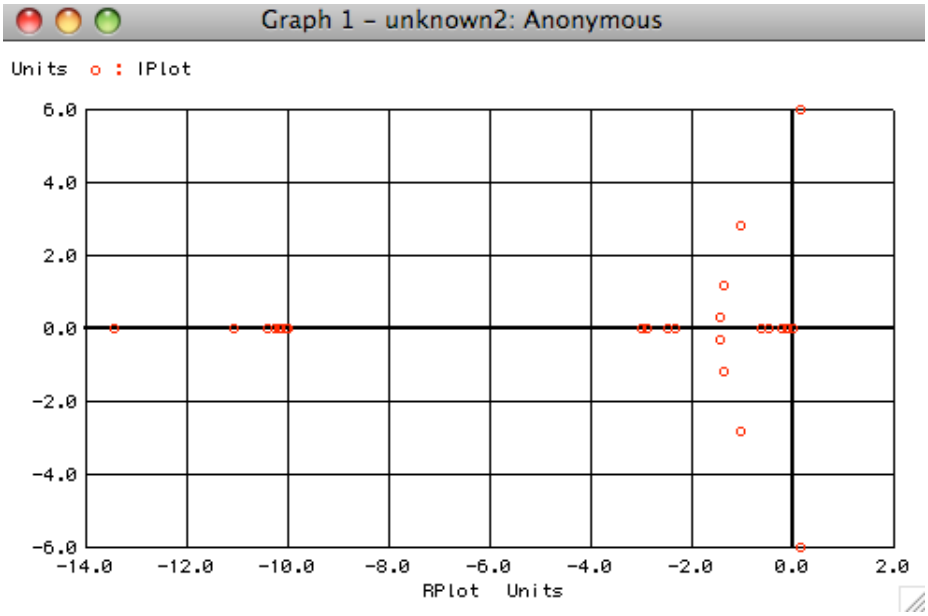
# ==========RootLocus==========

**SPICE PROVIDES A POLE/ZERO EXTRACTION FEATURE. THIS FEATURE CAN BE CROSS CHECKED TO A ROOT LOCUS PLOT OF AN ANALOG SYSTEM IN NEGATIVE FEEDBACK.**



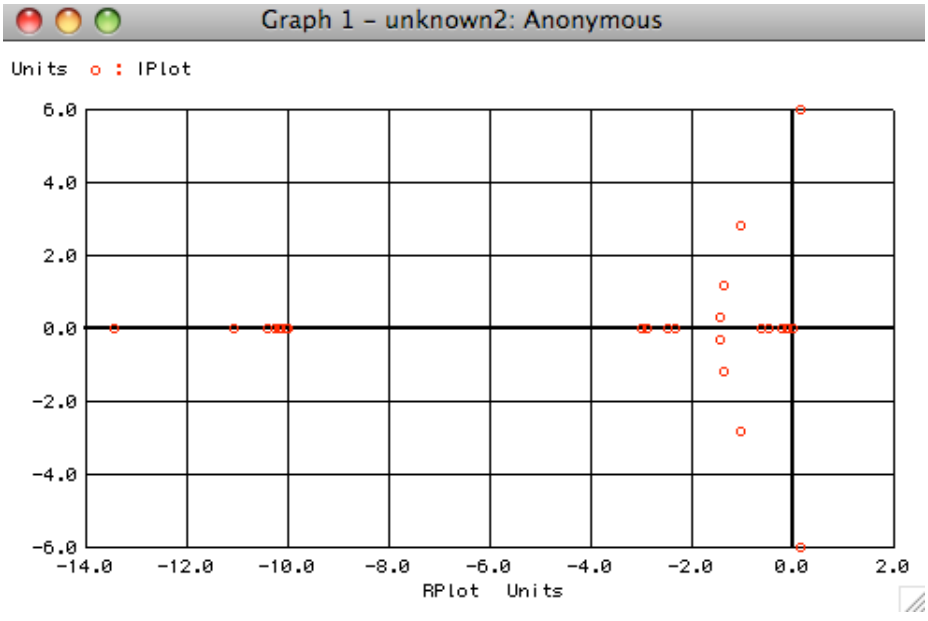

**RootLocus**

```
*
*  VIN       V00       R0      V0       R1       V1        R2      V2
*
*  P1 = .1/(s+.1)  s is in jw (2PI*F) format
*  P2 =  3/(s+3)
*  P3 =  10/(s+10)
*
*  Transfer_V00->V2 = 3/((s+.1)*(s+.3)*(s+10)
*  Root Locus needs this format   K*(s+z0)(s+z1)../(s+p0)(s+p1)
*  K_4_rootlocus = 3*Egain

VIN         VIN     0       1.00     AC      1
EGain       V00     0       VIN      V2      1

R0          V00     V0      1.0
C0          V0      0       10.0
EGN0        VEG0    0       V0       0       1
R1          VEG0    V1      1.0
C1          V1      0       .33
EGN1        VEG1    0       V1       0       1
R2          VEG1    V2      1.0
C2          V2      0       0.1


.control
pz                  vin     0       v2      0       vol     pol
setplot             new
set NameList =      ( RPlot IPlot   )
```

```
compose             GVals values   (-1)  0 .5 .6 .8 1 1.5 2.2
settype current     GVals
let NoOfG =         length(GVals)
begin
unset               interrupt

* ============Loop_K_Gain=============
set thisName =     $NameList[1]
let RPlot =        0*vector(40)
set thisName =     $NameList[2]
let IPlot =        0*vector(40)

let k =            1
while              ( k <= NoOfG )
let gainn =        10^GVals[k-1]
alter              egain      gain  = $&gainn
let K_4RL =        3*gainn
print              K_4RL
pz                 vin    0      v2     0      vol     pol
print              pz.pole(1)
let pr =           real(pz.pole(1))
let pi =           imag(pz.pole(1))
let                unknown.RPlot[unknown.k-1] = pr
let                unknown.IPlot[unknown.k-1] = pi
echo              "Preal =  $&pr Pimag =  $&pi   "


let  offset =      10
print              pz.pole(2)
let pr =           real(pz.pole(2))
let pi =           imag(pz.pole(2))
let                unknown.RPlot[offset + unknown.k-1] = pr
let                unknown.IPlot[offset + unknown.k-1] = pi
echo              "Preal =  $&pr Pimag =  $&pi   "


let                offset = 20
print              pz.pole(3)
let pr =           real(pz.pole(3))
let pi =           imag(pz.pole(3))
let                unknown.RPlot[offset + unknown.k-1] = pr
let                unknown.IPlot[offset + unknown.k-1] = pi
echo              "Preal =  $&pr Pimag =  $&pi   "

destroy
let k =            k + 1

if                 ($?interrupt)
goto               bail
endif
endwhile

settype            notype IPlot
settype            notype RPlot
setscale           GVals
set                pensize = 1
plot               IPlot vs RPlot pointplot

label              bail
echo              "Done."
end
.endc

.end

==========End================

Circuit: RootLocus*

k_4rl = 3.000000e-01
```

```
pz.pole(1) = -1.00044e+01,0.000000e+00
pz.pole(2) = -3.01542e+00,0.000000e+00
pz.pole(3) = -1.10494e-01,0.000000e+00
k_4rl = 3.000000e+00
pz.pole(1) = -1.00435e+01,0.000000e+00
pz.pole(2) = -2.87711e+00,0.000000e+00
pz.pole(3) = -2.09738e-01,0.000000e+00
k_4rl = 9.486833e+00
pz.pole(1) = -1.01344e+01,0.000000e+00
pz.pole(2) = -2.49756e+00,0.000000e+00
pz.pole(3) = -4.98312e-01,0.000000e+00
k_4rl = 1.194322e+01
pz.pole(1) = -1.01679e+01,0.000000e+00
pz.pole(2) = -2.32353e+00,0.000000e+00
pz.pole(3) = -6.38897e-01,0.000000e+00
k_4rl = 1.892872e+01
pz.pole(1) = -1.02603e+01,0.000000e+00
pz.pole(2) = -1.43501e+00,3.155547e-01
pz.pole(3) = -1.43501e+00,-3.15555e-01
k_4rl = 3.000000e+01
pz.pole(1) = -1.03993e+01,0.000000e+00
pz.pole(2) = -1.36551e+00,1.157896e+00
pz.pole(3) = -1.36551e+00,-1.15790e+00
k_4rl = 9.486833e+01
pz.pole(1) = -1.10834e+01,0.000000e+00
pz.pole(2) = -1.02345e+00,2.805694e+00
pz.pole(3) = -1.02345e+00,-2.80569e+00
k_4rl = 4.754680e+02
pz.pole(1) = -1.34516e+01,0.000000e+00
pz.pole(2) = 1.606726e-01,5.991900e+00
pz.pole(3) = 1.606726e-01,-5.99190e+00
Done.
```

**It is nice to see that the pole zero feature of spice can be sanity checked against a typical root locus plot for an Op Amp. The plot below is done in a point plot format which make it easier to pole value to a K value.**

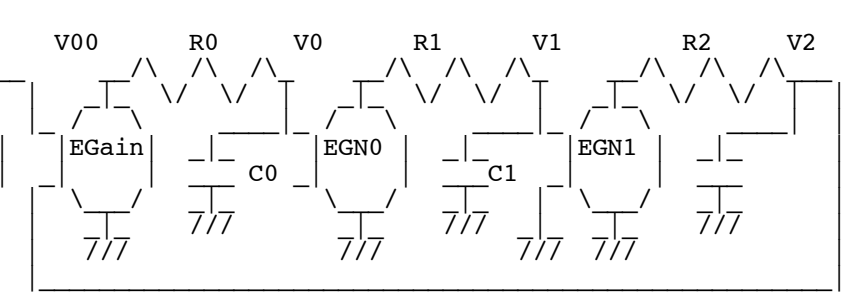



**Two details need to be watched out for. First the poles do not come out in frequency format. Second, the transfer function of the H(s) needs**

**to be in this format..**

**K_4_rootlocus*(s+z0)(s+z1)../(s+p0)(s+p1)**

**In the circuit below the low pass filters have constant terms that need to be taken account of.**

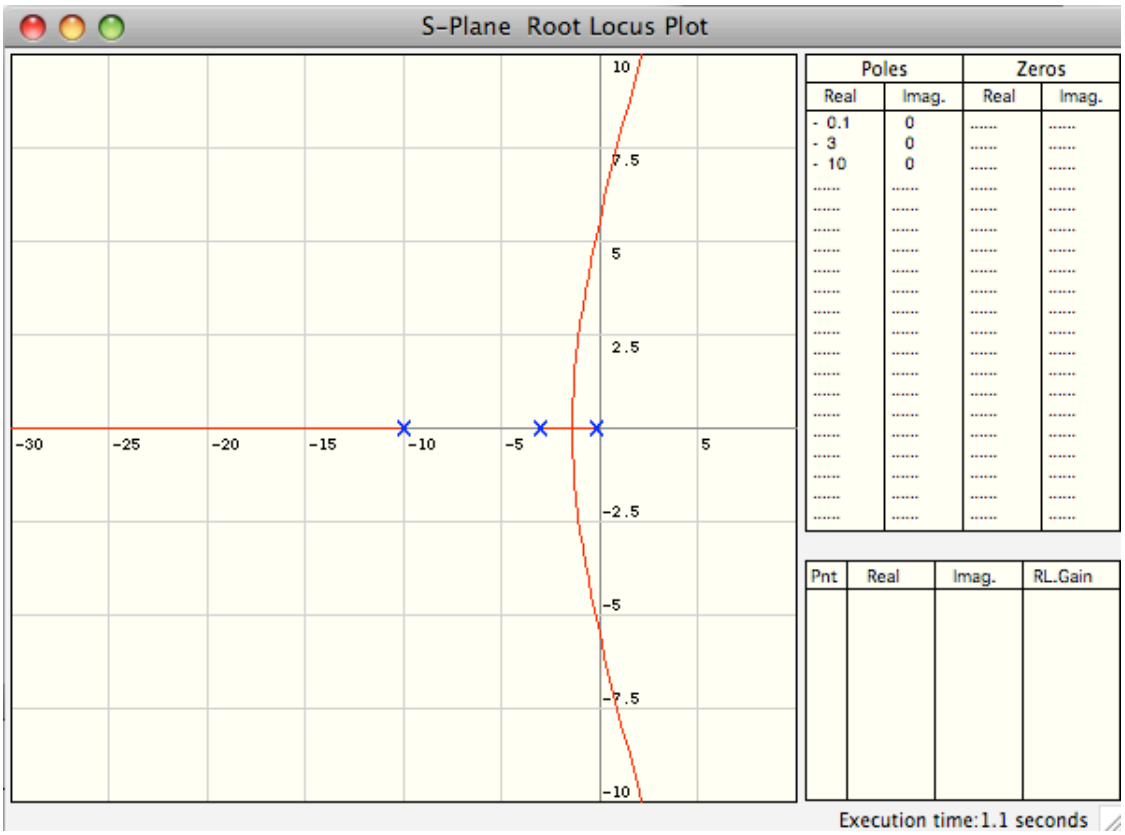


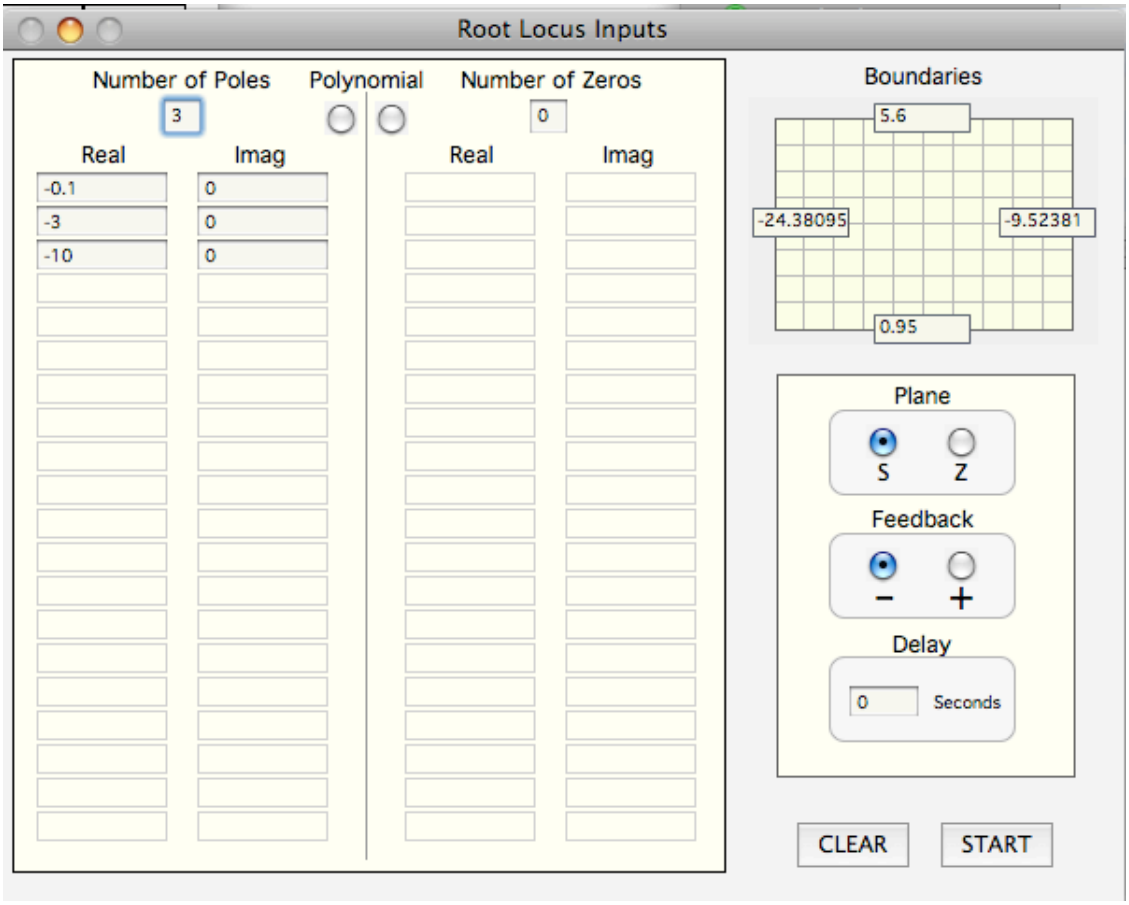

P1 = .1/(s+.1)  **s is in jw (2PI*F) format**
P2 =  3/(s+3)
P3 =  10/(s+10)

Transfer_V00->V2 = 3/((s+.1)*(s+.3)*(s+10)
Root Locus needs this format   K*(s+z0)(s+z1)../(s+p0)(s+p1)
**K_4_rootlocus = 3*Egain**

**The exact same root locus can be run off a mac or PC using a program from the following site.**

http://uk.geocities.com/a.a.robinson@btinternet.com/


S-Plane Root Locus Plot
10
7.5
5
2.5
-2.5
-5
-7.5
-10
-30
-25
-20
-15
-10
-5
5
Poles
Zeros
Real
Imag.
Real
Imag.
- 0.1
0
- 3
0
- 10
0
Pnt
Real
Imag.
RL.Gain
Execution time:1.1 seconds


Root Locus Inputs
Number of Poles
3
Polynomial
Number of Zeros
0
Real
Imag
Real
Imag
-0.1
0
-3
0
-10
0
Boundaries
5.6
-24.38095
-9.52381
0.95
Plane
S
Z
Feedback
−
+
Delay
0
Seconds
CLEAR
START