Title: AutoLISP Routine "pltfun"

Author(s): James R. Coe

Date: September 16, 1987

DISTRIBUTION:

GB Library
R. Norrod
R. Weimer
D. Schiebel
E. Childers
R. Lacasse
C. Brockway
J. Coe
G. Behrens
S. White
S. Srikanth
R. Fisher
F. Crews
B. Peery
AutoLISP Routine "pltfun"

If your Autocad system has the optional ADE-3 package you can define variables, perform arithmetic operations, and create new commands. Using these features, a routine has been written in AutoLISP which draws functions of the form y = f(x) on Autocad drawings. Rectangular or polar graphs of a given function can be constructed. The routine is stored in the file \acad\dwg\pltfun.lsp.

To load the program from within Autocad, type (load "pltfun") in response to the prompt Command:. The new commands RECTPLT or POLARPLT can then be used to draw the function.

Using POLARPLT the Array Factor for a two element array fed by a Butler beamforming network was plotted as shown in Figure 1. This is the function described by lines 13 thru 17 in the routine listed in Figure 2.

To plot other functions replace this function with the desired one. You can edit the pltfun.lsp routine from within Autocad by typing edit after Command: is displayed. This editor is the one line EDLIN editor described in the MS-DOS manual. Also, the EMACS or PE editors can be used to modify the .lsp files, but to use these editors you must exit from the Autocad.

See the AUTOLISP Programmer's Reference Manual for a detailed description of the lisp construction of arithmetic functions.
FIG. 1 ARRAY FACTOR

4/7 LAMBDA

90 DEGREE HYBRID

0 DEG POWER CMBNR
FIGURE 2

"pltfun.lsp" Listing

1: ; get Xmin, Xmax and Xincrement
2:
3: (defun xrange ()
4: (setq xmin (getreal "Xmin "))
5: (setq xmax (getreal "Xmax "))
6: (setq xin (getreal "Xincrement "))
7: (setq x xmin)
8: )
9:
10:
11: ; define function
12: (defun function ()
13: (setq ds (* 8.0 (/ pi 7.0)))
14: (setq anl (* ds (cos x)))
15: (setq an (* 0.5 (+ anl (/ pi 2.0))))
16: (setq bn (* 0.5 (- anl (/ pi 2.0))))
17: (setq y (* 2.0 (+ (cos an) (cos bn))))
18: )
19:
20: ; draw rectangular plot of function
21: (defun drawr ()
22: (while (<= x xmax)
23: (function)
24: (setq ptl (list x y))
25: (setq x (+ x xin x))
26: (function)
27: (setq pt2 (list x y))
28: (command "pline" ptl pt2)
29: (command "")
30: )
31: )
32: ; rectangular plot command
33:
34: (defun C:RECTPLT ()
35: (xrange)
36: (blipechoff)
37: (drawr)
38: (blipechoon)
39: 30,57p
40: )
41:
42:
43: ; draw polar plot
44:
45: (defun C:POLARPLT ()
46: ; initialize
47: (setq xp 0.0)
48: (setq yp 0.0)

Continued --
FIGURE 2 (continued):

49:  (setq theta 0.0)
50:  (blipechoff)
51:    (while (< theta (* 2 pi))
52:      (setq x theta)
53:      (function)
54:      (setq xp (* y (cos theta)))
55:      (setq yp (* y (sin theta)))
56:      (setq pt3 (list xp yp))
57:      (setq theta (+ (/ pi 180) theta))
58:      (setq x theta)
59:      (function)
60:      (setq xp (* y (cos theta)))
61:      (setq yp (* y (sin theta)))
62:      (setq pt4 (list xp yp))
63:      (command "pline" pt3 pt4)
64:      (command "")
65:   )
66:  (blipechoon)
67: )
68: (defun blipechoff ()
69:   (setq sblip (Qetvar "blipmode"))
70:   (setq scmde (getvar "cmdecho"))
71:   (setvar "blipmode" 0)
72:   (setvar "cmdecho" 0)
73: )
74:
75: (defun blipechoon()
76:   (setvar "blipmode" sblip)
77:   (setvar "cmdecho" scmde)
78: )