

```

1  !S9Font.f95      Group ID: #9      Stereo-3D Simulation Environment Vsn:1.00
2  !2025.05.24.1840cdt- 3D text.
3
4  !               Author- Jeffrey M. Setterholm, Lakeville,MN 55044 USA
5  !               IP Status- Free source code (e.g.: post copyright)
6  !
7  !               Computer- "T3"/Dell Precision T3500/Intel i5 E5520/win10Pro-21H2
8  !                   ^name ^Mfgr.Id      ^chipset      ^OS
9  !                   /Absoft Pro Fortran 21.0.2/GeForce GTX 1050/f90gl~Glut3.7
10 !                   ^compiler ~Fortran 95      ^graphics card ^graphics
11
12 !       f90gl bindings- public domain; see "https://math.nist.gov/f90gl/"
13
14 !Disclaimer:
15 !*****
16 !*****      Individual cognition is always flawed,      *****
17 !*****      including yours and mine.      *****
18 !*****      - So: -      *****
19 !*****      Use this code at your own risk.      *****
20 !*****
21
22 !Table of Contents: ...use to search...
23 !Module CharDef
24 ! ~an English character vector font, & more.
25 !End Module CharDef
26 !Subroutine AlphaJS(cLabelL,PosLLCq,RpyDq,SizeHq,iCol,fLinewidth,iP)
27 !Subroutine Model7DoF(Xyzh,Rpyh,S,Model7h,iP)
28
29 !-----7-9
30
31 Module CharDef
32 !2018.08.16.2120cdt JMS- Transferred the C implementation to Fortran.
33 !2018.07.17.      JMS- Grid coordinates defining characters are now square.
34 !               vertical & horizontal lines can have the same width.
35 !2018.04.11.1400cdt JMS- Enlarged the Free symbol char(224)
36 !               - The identical character data is found in:
37 !               http://ftp.setterholm.com/3DenvC in file 3DVecText.h
38 !               - Traveler2/Athlon64/winXPPro-32/APF9.0-32
39 !use CharDef      !2018.03.21
40 implicit none
41 !---
42
43 !ASCII character starting locations in CharLoc (if >0) :
44 integer(2),public:: CharLoc(0:255) = (/
45 ! 0   1   2   3   4   5   6   7   8   9   a   b   c   d   e   f   &
46 ! 0, 0, 0, 0, 0, 0, 0, BEL, BS, HT, LF, VT, FF, CR, 0, 0,&
47 ! 0, 0, 0, 0, 0, 0, 0, 1, 67, 136, 169, 196, 220, 253, 0, 0,&
48 ! 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,&
49 ! 385, 388, 412, 427, 454, 505, 568, 634, 643, 670, 697, 718, 733, 754, 763, 781,&
50 ! 0 1 2 3 4 5 6 7 8 9 : ; < = > ? 3
51 790, 838, 856, 889, 931, 946, 979, 1018, 1030, 1081, 1120, 1153, 1180, 1192, 1207, 1219,&
52 ! @ A B C D E F G H I J K L M N O 4
53 1276, 1336, 1354, 1393, 1426, 1450, 1471, 1489, 1522, 1543, 1564, 1585, 1609, 1621, 1639, 1654,&
54 ! P Q R S T U V W X Y Z [ \ ] ^ _ 5
55 1684, 1708, 1744, 1777, 1822, 1837, 1858, 1870, 1888, 1903, 1921, 1936, 1951, 1960, 1975, 1987,&
56 ! ` a b c d e f g h i j k l m n o 6
57 1996, 2005, 2038, 2071, 2098, 2131, 2164, 2188, 2224, 2248, 2272, 2305, 2326, 2338, 2377, 2401,&
58 ! p q r s t u v w x y z { | } ~ 7
59 2431, 2464, 2506, 2527, 2566, 2581, 2605, 2617, 2635, 2650, 2683, 2698, 2734, 2743, 2779, 2812,&
60 !
61 !
62 !

```

```

63      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,&
64      !      • box grid      ™      9
65      0, 0, 0, 0, 0, 2860, 0, 0, 0, 2884, 0, 0, 0, 0, 0, 0,&
66      !      ©      ®      a
67      0, 0, 0, 0, 0, 0, 0, 0, 0, 2938, 0, 0, 0, 0, 2992, 0,&
68      !      ±      b
69      3049,3091, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,&
70      !Special use in CEnv3D- Colors (black screen background):
71      !mov wht lRed Red Mag Yel lGrn Grn Cyn lBlu Blu LGry Gry Brn Prp Blk c
72      3112,3184,3238,3316,3385,3448,3493,3574,3646,3700,3775,3841,3925,4000,4078,4150,&
73      !Special use in CEnv3D- Shape modifiers:
74      !toff w1. w2. w3. w4. w5. ita und sub sup      show      d
75      4219,4300,4327,4369,4414,4447,4486,4534,4588,4675, 0, 0, 0,4747, 0, 0,&
76      !Free © ® ™ ©x      box gride
77      4831,4906,4960,5017,5071, 0, 0, 0, 0, 0, 0, 0, 0, 0, 5122,5140,&
78      !      f
79      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0/)&
80      ! 0 1 2 3 4 5 6 7 8 9 a b c d e f
81      !CharLoc(256) defined above.
82
83      integer(2),public:: PointLoc(5199)      !8 triples-per-line.
84      data PointLoc( 1: 600) / & !~ASCII characters 7- 38 decimal
85      0, 7, 21,1,-20, 30,2,-10, 30,2, -8, 27,2, -8, 21,2,-12, 18,2,-20, 18,1,-12, 18,&
86      2, -8, 15,2, -8, 9,2,-10, 6,2,-20, 6,2,-20, 30,1, 6, 12,2, -6, 12,2, -6,-12,&
87      2, 6,-12,1, -6, 0,2, 2, 0,1, 8, -6,2, 8,-30,2, 20,-30,0, 8, 22,1,-16, 24,&
88      2, -6, 24,2, -2, 21,2, -2, 15,2, -8, 12,2,-16, 12,1, -8, 12,2, -2, 9,2, -2, 3,&
89      2, -6, 0,2,-16, 0,2,-16, 24,1, 16, -3,2, 12, 0,2, 6, 0,2, 2, -3,2, 2, -9,&
90      2, 16,-15,2, 16,-21,2, 12,-24,2, 6,-24,2, 2,-21,0, 9, 10,1,-16, 24,2,-16, 0,&
91      1,-16, 12,2, -2, 12,1, -2, 24,2, -2, 0,1, 2, 0,2, 18, 0,1, 10, 0,2, 10,-24,&
92      0, 10, 8,1,-16, 24,2,-16, 0,2, -2, 0,1, 2,-24,2, 2, 0,2, 16, 0,1, 2,-12,&
93      2, 14,-12,0, 11, 7,1,-16, 24,2, -8, 0,2, -2, 24,1, 2, 0,2, 18, 0,1, 10, 0,&
94      2, 10,-24,0, 12, 10,1,-16, 0,2,-16, 24,2, -2, 24,1,-16, 12,2, -4, 12,1, 2,-24,&
95      2, 2, 0,2, 16, 0,1, 2,-12,2, 14,-12,0, 13, 17,1, -2, 18,2, -6, 24,2,-12, 24,&
96      2,-16, 18,2,-16, 6,2,-12, 0,2, -6, 0,2, -2, 6,1, 2,-24,2, 2, 0,2, 12, 0,&
97      2, 16, -3,2, 16, -9,2, 12,-12,2, 2,-12,1, 8,-12,2, 16,-24,0, 27, 25,1, -8, 30,&
98      2,-20, 30,2,-20, 6,2, -8, 6,1,-20, 18,2,-12, 18,1, 6, 9,2, 4, 12,2, -4, 12,&
99      2, -6, 9,2, -6, 3,2, 6, -3,2, 6, -9,2, 4,-12,2, -4,-12,2, -6, -9,1, 20, -9,&
100     1, 20,-12,2, 16, -6,2, 12, -6,2, 8,-12,2, 8,-24,2, 12,-30,2, 16,-30,2, 20,-24,&
101     0, 32, 0,0, 33, 7,1, 0, 30,2, 0,-18,1, 0,-27,2, 2,-30,2, 0,-33,2, -2,-30,&
102     2, 0,-27,0, 34, 4,1, -8, 30,2, -8, 15,1, 8, 30,2, 8, 15,0, 35, 8,1, -8, 18,&
103     2,-14,-24,1, 12, 18,2, 6,-24,1,-16, 6,2, 16, 6,1,-18,-12,2, 14,-12,0, 36, 16,&
104     1, 14, 12,2, 6, 21,2, -6, 21,2,-14, 12,2,-14, 9,2, -6, 0,2, 6, 0,2, 14, -9,&
105     2, 14,-12,2, 6,-21,2, -6,-21,2,-14,-12,1, -4, 27,2, -4,-27,1, 4, 27,2, 4,-27,&
106     0, 37, 20,1,-10, 30,2, -6, 27,2, -6, 18,2,-10, 15,2,-16, 15,2,-20, 18,2,-20, 27,&
107     2,-16, 30,2,-10, 30,1, 20, 30,2,-20,-30,1, 10,-15,2, 16,-15,2, 20,-18,2, 20,-27,&
108     2, 16,-30,2, 10,-30,2, 6,-27,2, 6,-18,2, 10,-15,0, 38, 21,1, 20,-30,2,-12, 9,&
109     2,-14, 15,2,-14, 18,2,-10, 27,2, -4, 30,2, 2, 30,2, 8, 27,2, 12, 18,2, 12, 15 &
110     /
111     data PointLoc( 601:1200) / & !~ASCII characters 39- 61
112     2, 8, 9,2,-12, -6,2,-18, -9,2,-20,-15,2,-20,-18,2,-18,-24,2,-14,-27,2, -6,-30,&
113     2, 6,-30,2, 10,-27,2, 20,-15,0, 39, 2,1, 0, 30,2, 0, 15,0, 40, 8,1, 6, 30,&
114     2, 0, 27,2, -4, 18,2, -6, 6,2, -6, -6,2, -4,-18,2, 0,-27,2, 6,-30,0, 41, 8,&
115     1, -6, 30,2, 0, 27,2, 4, 18,2, 6, 6,2, 6, -6,2, 4,-18,2, 0,-27,2, -6,-30,&
116     0, 42, 6,1, 10, 24,2,-10,-24,1, 20, 0,2,-20, 0,1, 10,-24,2,-10, 24,0, 43, 4,&
117     1, 0, 18,2, 0,-18,1, 20, 0,2,-20, 0,0, 44, 6,1, 2,-33,2, -2,-33,2, -2,-30,&
118     2, 2,-30,2, 2,-36,2, -2,-45,0, 45, 2,1,-20, 0,2, 20, 0,0, 46, 5,1, 0,-27,&
119     2, 2,-30,2, 0,-33,2, -2,-30,2, 0,-27,0, 47, 2,1,-20,-30,2, 20, 30,0, 48, 15,&
120     1, 0,-30,2,-10,-30,2,-20,-15,2,-20, 0,2,-20, 15,2,-10, 30,2, 0, 30,2, 10, 30,&
121     2, 20, 15,2, 20, 0,2, 20,-15,2, 10,-30,2, 0,-30,1,-20,-30,2, 20, 30,0, 49, 5,&
122     1,-10, 6,2, 0, 30,2, 0,-30,1,-10,-30,2, 10,-30,0, 50, 10,1,-20, 18,2,-12, 24,&
123     2, 0, 30,2, 12, 24,2, 20, 15,2, 12, 6,2, 0, 0,2,-12, -6,2,-20,-30,2, 20,-30,&
124     0, 51, 13,1,-20, 18,2,-12, 30,2, 12, 30,2, 20, 18,2, 20, 6,2, 12, 0,2, -4, 0,&
125     2, 12, 0,2, 20, 6,2, 20, 18,2, 12, 20,2, 20, 18,0, 52, 4,1, 20, 0,2

```

3

```

187 0,107, 6,1,-18, 24,2,-18,-30,1,-18, -6,2, 10, 12,1,-14, -3,2, 18,-30,0,108, 3,&
188 1, 0, 24,2, 0,-30,2, 4,-30,0,109, 12,1,-18,-30,2,-18, 0,1,-18, -9,2,-12, 0,&
189 2, -6, 0,2, 0, -9,2, 0,-24,1, 0, -9,2, 6, 0,2, 12, 0,2, 18, -9,2, 18,-30,&
190 0,110, 7,1,-18,-30,2,-18, 0,1,-18, -9,2, -8, 0,2, 8, 0,2, 18, -9,2, 18,-30,&
191 /
192 data PointLoc(2401:3000) / & !~ASCII characters 111-174
193 0,111, 9,1, 8, 0,2, 18, -9,2, 18,-21,2, 8,-30,2, -8,-30,2,-18,-21,2,-18, -9,&
194 2, -8, 0,2, 8, 0,0,112, 10,1,-18,-45,2,-18, 0,1,-18, -9,2, -8, 0,2, 8, 0,&
195 2, 18, -9,2, 18,-21,2, 8,-30,2, -8,-30,2,-18,-21,0,113, 13,1, 18,-21,2, 8,-30,&
196 2, -8,-30,2,-18,-21,2,-18, -9,2, -8, 0,2, 8, 0,2, 18, -9,2, 18,-21,2, 12,-42,&
197 2, 14,-45,2, 18,-45,2, 20,-42,0,114, 6,1,-18,-30,2,-18, 0,1,-18, -9,2, -8, 0,&
198 2, 8, 0,2, 20, -9,0,115, 12,1, 18, -6,2, 8, 0,2, -8, 0,2,-18, -6,2,-18, -9,&
199 2, -8,-15,2, 8,-15,2, 18,-21,2, 18,-24,2, 8,-30,2, -8,-30,2,-18,-24,0,116, 4,&
200 1, 0, 24,2, 0,-30,1,-18, 12,2, 18, 12,0,117, 7,1,-18, 0,2,-18,-21,2, -8,-30,&
201 2, 8,-30,2, 18,-21,2, 18, 0,2, 18,-30,0,118, 3,1,-18, 0,2, 0,-30,2, 18, 0,&
202 0,119, 5,1,-18, 0,2, -8,-30,2, 0, -6,2, 8,-30,2, 18, 0,0,120, 4,1,-18, 0,&
203 2, 18,-30,1,-18,-30,2, 18, 0,0,121, 10,1,-18, 0,2,-18,-21,2,-10,-30,2, 8,-30,&
204 2, 18,-21,1, 18, 0,2, 18,-36,2, 8,-45,2,-10,-45,2,-18,-42,0,122, 4,1,-18, 0,&
205 2, 18, 0,2,-18,-30,2, 18,-30,0,123, 11,1, 6, 30,2, 0, 27,2, -2, 21,2, 0, 9,&
206 2, -2, 3,2, -6, 0,2, -2, -3,2, 0, -9,2, -2,-21,2, 0,-27,2, 6,-30,0,124, 2,&
207 1, 0, 30,2, 0,-30,0,125, 11,1, -6, 30,2, 0, 27,2, 2, 21,2, 0, 9,2, 2, 3,&
208 2, 6, 0,2, 2, -3,2, 0, -9,2, 2,-21,2, 0,-27,2, -6,-30,0,126, 10,1,-20, -3,&
209 2,-18, 3,2,-12, 6,2, -8, 6,2, -2, 3,2, 2, -3,2, 8, -6,2, 12, -6,2, 18, -3,&
210 2, 20, 3,0,127, 15,1,-14,-30,2,-14,-12,2,-16,-15,1, -6,-15,2, -2,-12,2, 2,-12,&
211 2, 6,-15,2, 6,-21,2, 2,-24,2, -4,-24,2, -6,-30,2, 6,-30,1, 8,-12,2, 20,-12,&
212 2, 14,-30,0,149, 7,1, 2, 3,2, 4, 0,2, 2, -3,2, -2, -3,2, -4, 0,2, -2, 3,&
213 2, 2, 3,0,153, 17,1,-18, 24,2,-18, 27,2, -2, 27,2, -2, 24,1,-10, 27,2,-10, 3,&
214 1,-12, 3,2, -8, 3,1, 0, 3,2, 4, 3,1, 2, 3,2, 2, 27,2, 10, 15,2, 18, 27,&
215 2, 18, 3,1, 16, 3,2, 20, 3,0,169, 17,1, 10, 21,2, 20, 12,2, 20,-12,2, 10,-21,&
216 2,-10,-21,2,-20,-12,2,-20, 12,2,-10, 21,2, 10, 21,1, 12, 6,2, 6, 12,2, -6, 12,&
217 2,-12, 6,2,-12, -6,2, -6,-12,2, 8,-12,2, 12, -6,0,174, 18,1, -8, -3,2, -8, 21,&
218 /
219 data PointLoc(3001:3600) / & !~ASCII characters 176-199
220 2, 4, 21,2, 8, 18,2, 8, 12,2, 4, 9,2, -8, 9,1, 2, 9,2, 8, -3,1, 8, 30,&
221 2, 20, 21,2, 20, -3,2, 8,-12,2, -8,-12,2,-20, -3,2,-20, 21,2, -8, 30,2, 8, 30,&
222 0,176, 13,1, 2, 30,2, 6, 27,2, 8, 24,2, 8, 21,2, 6, 18,2, 2, 15,2, -2, 15,&
223 2, -6, 18,2, -8, 21,2, -8, 24,2, -6, 27,2, -2, 30,2, 2, 30,0,177, 6,1,-20, 0,&
224 2, 20, 0,1, 0, 15,2, 0,-15,1,-20,-18,2, 20,-18,0,192, 23,1,-20, 15,2,-20, 27,&
225 2,-16, 21,2,-12, 27,2,-12, 15,1, -4, 27,2, -2, 24,2, -2, 18,2, -4, 15,2, -6, 15,&
226 2, -8, 18,2, -8, 24,2, -6, 27,2, -4, 27,1, 2, 27,2, 6, 15,2, 10, 27,1, 20, 27,&
227 2, 14, 27,2, 14, 15,2, 20, 15,1, 14, 21,2, 20, 21,0,193, 17,1,-20, 24,2,-16, 6,&
228 2,-12, 24,2, -8, 6,2, -4, 24,1, 0, 24,2, 0, 6,1, 0, 15,2, 10, 15,1, 10, 24,&
229 2, 10, 6,1, 18, 24,2, 18, 6,1, 16, 24,2, 20, 24,1, 16, 6,2, 20, 6,0,194, 25,&
230 1,-20, 18,2,-20, 6,2,-16, 6,1,-14, 6,2,-14, 24,2, -6, 24,2, -4, 21,2, -4, 18,&
231 2, -6, 15,2,-14, 15,1,-10, 15,2, -4, 6,1, 8, 24,2, 0, 24,2, 0, 6,2, 8, 6,&
232 1, 0, 15,2, 6, 15,1, 12, 24,2, 18, 24,2, 20, 18,2, 20, 12,2, 18, 6,2, 12, 6,&
233 2, 12, 24,0,195, 22,1,-20, 6,2,-20, 24,2,-12, 24,2, -8, 21,2, -8, 18,2,-12, 15,&
234 2,-20, 15,1,-14, 15,2, -8, 6,1, 6, 24,2, -4, 24,2, -4, 6,2, 6, 6,1, -4, 15,&
235 2, 4, 15,1, 10, 24,2, 16, 24,2, 20, 18,2, 20, 12,2, 16, 6,2, 10, 6,2, 10, 24,&
236 0,196, 20,1,-20, 6,2,-20, 24,2,-14, 15,2, -8, 24,2, -8, 6,1, -4, 6,2, 0, 24,&
237 2, 4, 6,1, -2, 15,2, 2, 15,1, 20, 21,2, 16, 24,2, 12, 24,2, 8, 21,2, 8, 9,&
238 2, 12, 6,2, 16, 6,2, 20, 9,2, 20, 15,2, 14, 15,0,197, 14,1,-20, 24,2,-14, 15,&
239 2, -8, 24,1,-14, 15,2,-14, 6,1, 6, 24,2, -4, 24,2, -4, 6,2, 6, 6,1, -4, 15,&
240 2, 2, 15,1, 10, 24,2, 10, 6,2, 20, 6,0,198, 26,1,-20, 18,2,-20, 6,2,-16, 6,&
241 1, -4, 21,2, -8, 24,2,-10, 24,2,-14, 21,2,-14, 9,2,-10, 6,2, -8, 6,2, -4, 9,&
242 2, -4, 15,2, -8, 15,1, 0, 6,2, 0, 24,2, 6, 24,2, 8, 21,2, 8, 18,2, 6, 15,&
243 2, 0, 15,1, 2, 15,2, 8, 6,1, 12, 6,2, 12, 24,2, 20, 6,2, 20, 24,0,199, 23,&
244 1,-10, 21,2,-14, 24,2,-16, 24,2,-20, 21,2,-20, 9,2,-16, 6,2,-14, 6,2,-10, 9,&
245 /
246 data PointLoc(3601:4200) / & !~ASCII characters 200-207
247 2,-10, 15,2,-14, 15,1, -6, 6,2, -6, 24,2, 0, 24,2, 4, 21,2, 4, 18,2, 0, 15,&
248 2, -6, 15,1, -2, 15,2, 4, 6,1, 8, 6,2, 8, 24,2, 20, 6,2, 20, 24,0,200, 17,&
249 1, 8, 21,2, 12, 24,2, 16, 24,2, 20, 21,2, 12, 6,2, 12, 6,2, 8, 0,2,

```

5

```

311 2, -8, 3,1, 0, 3,2, 4, 3,1, 2, 3,2, 2, 27,2, 10, 15,2, 18, 27,2, 18, 3,&
312 1, 16, 3,2, 20, 3,0,228, 16,1, 12, 6,2, 6, 12,2, -6, 12,2,-12, 6,2,-12, -6,&
313 2, -6,-12,2, 8,-12,2, 12, -6,1, 20, 21,2, 20,-12,2, 10,-21,2,-20,-21,2,-20, 12,&
314 2,-10, 21,2, 20, 21,2,-20,-21,0,238, 5,1, 30, 30,2, 30,-30,2,-30,-30,2,-30, 30,&
315 2, 30, 30,0,239, 19,1, 30, 30,2, 30,-45,2,-30,-45,2,-30, 30,2, 30, 30,1,-30, 15,&
316 2, 30, 15,1,-30, 0,2, 30, 0,1,-30,-15,2, 30,-15,1,-30,-30,2, 30,-30,1,-14, 30,&
317 2,-14,-45,1, 0, 30,2, 0,-45,1, 14, 30,2, 14,-45 /
318 !PointLoc(5199) defined above. JMS 2018.08.16.2120
319 !-----7 9
320 !contains
321 !-----7 9
322 End Module CharDef
323 !-----7 9
324
325 !-----7 9
326 Subroutine AlphaJS(cLabelL,PosLLCq,RpyDq,SizeHq,iCol,fLinewidth,iP)
327 !2021.10.03.0715cdt JMS- Adapted to print to screen.
328 !2018.09.12.1035cdt JMS- Modified for TweakEngine to output to a .3dv file.
329 ! PosLLCq,RpyDq, & SizeHq are quad precision.
330 !2018.08.16.2210cdt JMS- Square grid.
331 !2018.04.06.1620cdt JMS- 7DoF vector-based characters;
332 ! the individual character shapes are specified.
333 ! - Traveler2/Athlon64/WinXPPro-32/APF9.0-32
334
335 !This supports both: font modification and extension -and-
336 ! exporting characters as 3D lines.
337 !2016.06.24.1600cdt JMS- worked in C - but - color of first character was
338 ! "kluged-to-correctness".
339
340 ! Unrotated characters are written parallel-to the +X axis,
341 ! readable from the +Y direction
342 ! in righthanded "Flight Simulation" (abbrev. 'Fs') coordinates.
343 !
344 ! Function 'HindSight'- with S.ThreePhase=3: uses Fs-based screen coordinates
345 ! "2D/3D" for the various projection matrices
346 ! +X:forward,+Y:right,+Z:down.
347 ! - with S.ThreePhase=2: uses glOrtho (which is non-FS).
348 ! "2D ortho" +X:right,+Y:up, +Z:forward,
349 ! a left-handed coordinate frame.
350 !
351 ! Use 'PrntOrtho' to write stationary color 2D text on the screen.
352
353 !Function 'Colors3D' defines the colors
354
355 !VecText.h has the following c* & d* characters encoded,
356 ! ... however, they presently have no useful effect.
357 !The objective is to enrich the formatting within a text file
358 ! by repurposing some relatively unused characters.
359 !The key question is: How to easily insert hex code characters in text?
360
361 !Custom character usages:
362 !My colors: [c][0] 'move without draw' - a no-op in this context
363 ! (clarifies & simplifies the content of .3dv files.)
364 ! white [c][1]
365 ! Red [2]
366 ! !Green= Magenta [3] <- ! = "not"
367 ! !Blue = Yellow [4]
368 ! light Green [5]
369 ! Green [6]
370 ! !Red = Cyan [7]
371 ! light Blue [8]
372 ! Blue [9]

```

```

373 !      light Gray      [a]
374 !          Gray      [b]
375 !          Brown     [c]
376 !          Purple    [d]
377 !      !White Black   [e]
378 !          user-defined [f]
379
380 ! Shape modifiers [d][0] toggles-> off
381 !     Line width  [ ][1] =1.
382 !                 [ ][2] =2.
383 !                 [ ][3] =3.
384 !                 [ ][4] =4.
385 !                 [ ][5] =5.
386 !     Italics      [ ][6] toggle on/off
387 !     Underlined   [ ][7]
388 !     Subscript    [ ][8]
389 !     Superscript  [ ][9]
390 !                 [ ][a]
391 !                 [ ][b]
392 !                 [ ][c]
393 !     Show controls [ ][d]
394 !                 [ ][e]
395 !                 [ ][f]
396
397 !-----
398 use OpenGLRec,only: & !Ref: OpenGL GL/GLU/GLUT docs
399     glVertex3dv , glShadeModel, GL_SMOOTH , GL_FLAT, glBegin &
400     ,GL_LINE_STRIP, GL_LINES , glLineWidth ,glFlush , glEnd, glFlush
401
402 use CharDef
403 !use Tweakrec, only: TL2 !Used in calling Draw3dJTL(TL2,0) to draw every line
404 !     segment of each character.
405 !use opengl_gl
406 implicit none
407 character:: cLabelL*80 !Label- text
408 real(8) :: PosLLCq(3) ! - position of the lower left corner (not homog.)
409 real(8) :: RpyDq(3) ! - Roll,pitch, & yaw of the label (not homog.)
410 real(8) :: SizeHq ! - size
411 integer(4) :: iCol ! - color
412 real(4) :: fLineWidth ! - line width
413 integer(4) :: iP !Print flag. Prints when iP>5 to unit# IP.
414 !--
415 real(8) :: PosLLC(3) ! - position of the lower left corner (not homog.)
416 real(8) :: RpyD(3) ! - Roll,pitch, & yaw of the label (not homog.)
417 real(8) :: SizeH ! - size
418
419 integer(4) :: i,iPass,j,j1,j2,k,m,n,nCh,nChars,nPts,nPu
420 character:: cA*80, cAi(80)*1 ;equivalence(cA,cAi)
421 character:: cB*1 ;integer(4) :: iB ;equivalence(cB,iB )
422 !--
423 real(8) :: Pin(3)
424 real(8) :: Pout(3)
425
426 real(8) :: pXYZ(4) !Homog.i/o- pos.(X,Y,Z,1.) - per FSCoords [meters]
427 real(8) :: aRpy(4) ! - att.(Roll,Pitch,Yaw)- [degrees]
428 real(8) :: hXR(4,4) ! - trans & rot matrix- i/o [i/o:h-meters]
429 !integer(4) :: Mode !=1:p&a->h & hinv
430 real(8) :: VecBehind(3)
431 integer(4) :: iColor !DOS color
432 !---
433
434 !-- Compute the text's translation & rotation offset H-matrix:

```

```

435 PosLLC   = dble(PosLLCq)
436 RpyD      = dble(RpyDq )
437 SizeH     = dble(SizeHq )
438
439 pXYZ(1:3)=PosLLC; pXYZ(4)=1.d0
440 aRpy(1:3)=RpyD;   aRpy(4)=1.d0
441
442 call Model7DoF(pXYZ,aRpy,1.d0,hXR,iP)
443
444 VecBehind=-hXR(1:3,2)*SizeH/50.d0
445 !----- Render the text
446 nChars=len_trim(cLabelL)
447 cA=Char(0)
448 cA(1:nChars)=cLabelL(1:nChars)
449
450 !if(iP.gt.5) write(iP, "('AlphaJS: ',i2,1x,a80)") nChars,cLabelL(1:nChars)
451 !if(iP.gt.5) write(iP, "('AlphaJS: ',i2,1x,80i4)") nChars &
452               ,(ichar( cLabelL(i:i) ),i=1,nChars)
453
454 m= 0 !Zero the line counter. 0 is the index of the first line.
455 n=-1 !Set the character offset to 'before the first character' of the line
456 !Process the incoming label... character by character... starting at (1):
457 do i=1,nChars; cB=cAi(i); iB=iand(iB,255); nCh=iB
458   if(iB==224) call BreakHere(iP,"S9Font@L444")
459   if(iB.le.0) exit
460   if(iP.gt.5) write(iP, "('i= ',i2,1x,a1,1x,i3)") i,cB,nCh
461
462   ! Recognize & utilise some of the traditional control characters
463   ! to easily write paragraphs in 7dof space:
464   if((nCh.ge.0).and.(nCh.lt.32)) then
465     select case(nCh)
466       case( 8); n= n-1           !BackSpace           '\b'
467       case( 9); n=(n+1)/8       !Horizontal Tab- 8 wide  '\t'
468         n=(n+1)*8-1
469       case(11); m=(m )/5        !Vertical   Tab- 5 high  '\v'
470         m=(m+1)*5
471       case(10); m= m+1          !Line Feed           '\n'
472       case(13); n=  -1         !Carriage Return     '\r'
473       case(12); m= m+80        !Form Feed           '\f'
474     end select !nCh
475     cycle
476   endif !0 <= nCh <32
477   n=n+1; !move right by one character.
478   if(nCh.lt. 32) nCh= 32
479   if(nCh.gt.255) nCh=255
480   j1 = CharLoc(nCh)
481   if((iP>5).and.(nCh==224)) &
482     write(iP, "('S9Font@L466 Free Symbol:224 requested. j1 = ',i6,i4)") &
483       j1,PointLoc(j1+2)
484   if(j1.eq.0) cycle !if the character isn't defined- skip to the next one.
485   j1 = j1 !This is the beginning of the character's information
486   nPts = PointLoc(j1+2)
487   if(nPts.eq.0) cycle !if the charcter has no points- go to the next one.
488   j2 = j1+(1+nPts)*3 !This is the beginning of the following character
489
490   nPu= 0; k = 0
491   if(iP.gt.5) write(iP, "('i= ',i2,1x,a1,1x,i3,i3,i2)") i,cB,nCh,nPts,iPass
492 ! Draw the intended character
493   iColor=iCol
494
495   call colors3D(iColor)
496   if(fLinewidth>0.) call glLinewidth(fLinewidth)

```



```

497 call glShadeModel(GL_FLAT)
498 if(iCol>0) then;call glBegin(GL_LINES);call glEnd;call glFlush !Quirk fix?
499 endif !iCol>0)
500
501 call glBegin(GL_LINE_STRIP)
502 do j=j1+3,j2-1,3; nPu=nPu+1 !Step through the character's points
503 if(PointLoc(j).gt.2) stop "Huge problem!"
504 if(PointLoc(j).eq.1) k = 0
505 k = k + 1
506 !Draw (or move): Resize first, the 7th dof, and increment:
507 Pin(1) = ( (PointLoc(j+1)+30.d0) /60.d0+ n ) *SizeH
508 Pin(2) = 0.d0
509 Pin(3) = ( -(PointLoc(j+2)+30.d0) /60.d0 +m*1.5d0) *SizeH !/20.
510 !if(nCh==224) Pin = 1.5d0 * Pin
511 !Then translate & rotate... the other 6 dof's:
512 Pout(1)=hXR(1,1)*Pin(1)+hXR(1,3)*Pin(3)+hXR(1,4)
513 Pout(2)=hXR(2,1)*Pin(1)+hXR(2,3)*Pin(3)+hXR(2,4)
514 Pout(3)=hXR(3,1)*Pin(1)+hXR(3,3)*Pin(3)+hXR(3,4)
515 if((k==1).and.(j>j1+3)) then
516 call glEnd; call glFlush
517 call glBegin(GL_LINE_STRIP)
518 endif
519 call glVertex3dv(Pout)
520 enddo!j
521 call glEnd !Step through the character's points
522
523 enddo!i
524
525 End Subroutine AlphaJS
526 !-----7 9
527
528 Subroutine Model7DoF(Xyzh,Rpyh,S,Model7h,iP) !version: 2018.08.03
529 ! "Model 7DoF" ... "DoF" = "Degrees of Freedom"
530 !Translation is 3DoF, Rotation is 3Dof, & Scale is 1Dof => 7DoF
531 !The general transform for inserting rigid 3D objects into 3D views.
532 !This sub routine concatenates (~mashes) all seven effects into a single 4x4
533 ! matrix; the pre-concatinated symbolic sub matrices are:
534 !The inverse sequence is:
535 !1. S = Scale (3D Magnify) the object: unTranslate
536 ! +S , 0 , 0 , 0 +1 , 0 , 0 , -X
537 ! 0 ,+S , 0 , 0 0 ,+1 , 0 , -Y
538 ! 0 , 0 ,+S , 0 0 , 0 ,+1 , -Z
539 ! 0 , 0 , 0 ,+1 0 , 0 , 0 ,+1
540 ! (The reciprocals of a diagonal scaling matrix are the inverse.)
541 !2. Roll: sR=sine(Roll), cR=cosine(Roll) unYaw
542 ! +1 , 0 , 0 , 0 +cY ,+sY , 0 , 0
543 ! 0 ,+cR,-sR, 0 -sY ,+cY , 0 , 0
544 ! 0 ,+sR,+cR, 0 0 , 0 ,+1 , 0
545 ! 0 , 0 , 0 ,+1 0 , 0 , 0 ,+1
546 !3. Pitch: sP=sine(Pitch), cP=cosine(Pitch) unPitch
547 ! +cP, 0 ,+sP, 0 +cP , 0 , -sP , 0
548 ! 0 ,+1 , 0 , 0 0 ,+1 , 0 , 0
549 ! -sP, 0 ,+cP, 0 +sP , 0 ,+cP , 0
550 ! 0 , 0 , 0 ,+1 0 , 0 , 0 ,+1
551 !4. Yaw: sY=sine(Yaw) ,cY=cosine(Yaw) unRoll
552 ! +cY,-sY, 0 , 0 +1 , 0 , 0 , 0
553 ! +sY,+cY, 0 , 0 0 ,+cR ,+sR , 0
554 ! 0 , 0 ,+1 , 0 0 , -sR ,+cR , 0
555 ! 0 , 0 , 0 ,+1 0 , 0 , 0 ,+1
556 ! ( The transpose of a pure-rotation matrix is the inverse;
557 ! this remains true even after [Yaw]<-[Pitch]<-[Roll] concatenation. )
558 !5. Translate to (X,Y,Z): unSScale

```

```

559 !      +1 , 0 , 0 ,+X      +1/S, 0 , 0 , 0
560 !      0 ,+1 , 0 ,+Y      0 ,+1/S, 0 , 0
561 !      0 , 0 ,+1 ,+Z      0 , 0 ,+1/S, 0
562 !      0 , 0 , 0 ,+1      0 , 0 , 0 ,+1
563 !      (The negatives of a pure-translation matrix are the inverse.)
564
565 !Matrix concatenation sequence:
566 !      [5:translate]<-[4:Scale]<-[3:Yaw]<-[2:Pitch]<-[ 1:Roll ]
567 ! & for the inverse:
568 !      [ unRoll ]<-[unPitch]<-[unYaw]<-[unScale]<-[untranslate]
569
570 !The resulting (4,4) matrix is implemented below.
571
572 !-----
573 implicit none                                     !arguments
574
575 real(8) ::xyzh(4)      !The seven "Degrees of Freedom"(DoF) are:
576 real(8) ::Rpyh(4)      !"translation": X , Y ,Z {,h} 1,2,3
577 real(8) ::S            !" rotation ": Roll,Pitch,Yaw {,h} 4,5,6
578                                     !" scale ": scale 7
579 real(8) ::Model7h(4,4) !Resulting homogeneous 7DoF matrix output.
580
581 integer(4)::iP      !Write enable>5: write(iP,...)
582 !---
583 real(8) :: X , Y , Z      !Translations
584 real(8) ::Roll , Pitch, Yaw !Rotations
585 real(8) ::sr,cr, sp,cp, sy,cy !Sines & Cosines
586 real(8) :: H16(16)
587 integer(4)::i,j,k
588 !-----
589 !Extract non-homogeneous values and compute the rotation sines & cosines:
590 X =xyzh(1)/xyzh(4)
591 Y =xyzh(2)/xyzh(4)
592 Z =xyzh(3)/xyzh(4)
593 Roll =Rpyh(1)/Rpyh(4); sr=dsind(Roll ); cr=dcosd(Roll )
594 Pitch=Rpyh(2)/Rpyh(4); sp=dsind(Pitch); cp=dcosd(Pitch)
595 Yaw =Rpyh(3)/Rpyh(4); sy=dsind(Yaw ); cy=dcosd(Yaw )
596
597 ! Translate(X,Y,Z) <- Scale(S) <- Rotate(Roll,Pitch,Yaw)... concatenated.
598 !Model 7DoF homogeneous transform: ----- 2017.05.23.0900
599 H16= (/ S*CY*CP , S*(CY*SP*SR-SY*CR) , S*(CY*SP*CR+SY*SR) , X &
600 , S*SY*CP , S*(SY*SP*SR+CY*CR) , S*(SY*SP*CR-CY*SR) , Y &
601 , -S*SP , S*CP*SR , S*CP*CR , Z &
602 , 0.d0 , 0.d0 , 0.d0 , +1.d0 /)
603 ! ...pack the results in their (4,4) array:
604 k=0; do i=1,4; do j=1,4; k=k+1; Model7h(i,j)=H16(k); enddo; enddo!i
605
606 !Concatenated Symbolic Inverse:
607 !      CY*CP /S, SY*CP /S, -SP /S , ( -X* CY*CP /S
608 !      -Y* SY*CP /S +Z*SP /S )
609 ! (CY*SP*SR-SY*CR)/S, (SY*SP*SR+CY*CR)/S, CP*SR/S , ( -X*(CY*SP*SR-SY*CR)/S
610 ! -Y*(SY*SP*SR+CY*CR)/S -Z*CP*SR/S )
611 ! (CY*SP*CR+SY*SR)/S, (SY*SP*CR-CY*SR)/S, CP*CR/S , ( -X*(CY*SP*CR+SY*SR)/S
612 ! -Y*(SY*SP*CR-CY*SR)/S -Z*CP*CR/S )
613 ! 0 , 0 , 0 , (+1)
614
615 !Fact N : Numeric matrix inversion sidesteps solving for symbolic inverses.
616
617 if(iP>5) then
618 write(iP, "('sub:Model7DoF: ',45('-'))")
619 write(iP, "(9x,'user units',13x,'degrees',11x,'multiplier')")
620 write(iP, "(' X =',f12.6,' Roll=',f12.6,' Scale=',f12.6)") &

```

```
621      X      Roll      S
622      write(iP,"(' Y      =',f12.6,' Pitch=',f12.6)") Y,Pitch
623      write(iP,"(' Z      =',f12.6,' Yaw=',f12.6)") Z,Yaw
624      write(iP,"(' Model7h:'      )")
625      do i=1,4
626          write(iP,"(7x,4f12.6)") (Model7h(i,j),j=1,4)
627      enddo!i
628      endif !iP>5
629
630      return
631 End Subroutine Model7DoF
632 !-----7 9
633
634
```