graphics3d_E

COLLABORATORS			
	TITLE : graphics3d_E		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		August 22, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

graphics3d_E

1.1 graphics3d_E.doc

graphics3d.library

GD_addobjpoly()	GD_addobjvertex()	GD_ambientlight()
GD_aspectratio()	GD_cascene()	GD_cattpoly()
GD_changeviewmode()	GD_changeviewmodeobj()	GD_clipbox()
GD_clipmode()	GD_close_display3d()	GD_createlightsource()
GD_deleteobject()	GD_dfl2fix()	GD_display3d()
GD_fix2dfl()	GD_fix2int()	GD_fix2sfl()
GD_frustum()	GD_getobj()	GD_int2fix()
GD_moveforward()	GD_newobj()	GD_newview()
GD_over()	GD_paintframe()	GD_pickobj()
GD_positioncamera()	GD_positionobject()	GD_recalcobj()
GD_rgb4()	GD_rotateobject()	GD_scaleobject()
GD_setobj()	GD_sfl2fix()	GD_switch_rp()
GD_touchpalette()	GD_translateobject()	GD_viewangle()

1.2 graphics3d.library/GD_display3d()

```
NAME
     GD_display3d  --  To initialize all ambient for the library.
SYNOPSIS
     ambient3d=GD_display3d(win, x0, y0, scrw, scrh, vdist)
                            A0   D0  D1  D2    D3    D4
     struct ambient3d *GD_display3d(struct Window*,LONG,LONG,LONG,LONG,LONG);

FUNCTION
     create and initialized the ambient3d structure that is the describer  ↩
                of the 3d scene and is used us input from all other functions.

INPUTS
     win   = pointer to Window structure of the window where you want
             viewer the 3d scene.
     x0,y0 = coordinates of upper left corner of the box that define
             the visualizations limits of the scene.
     scrw  = width of this box.
```

```
     scrh  = height of this box.
     vdist = distance from observer and proiection plane ,is expressed
             as integer.
RESULT
     ambient3d = pointer to ambient3d structure created, if it is equal
                 to 0 than there is an error and the inizalization
                 is aborted.

BUGS
     anyone note, if you find and tell me.

NOTES
     This function it must be use BEFORE all the other.
     It can be used more than one time on the same program than storing
     separately the pointer returned, it possible to work simultaneously
     and independently an all the scenes so definited.
     In the future perhaps I will make possible generate more scene of
     the same 3D space (for example to do more view) but all with the
     same memories areas for the objects definition .Now the library ever
     reallocate all areas and if the objects are much or complex it use
     very more memory.

SEE ALSO
     GD_close_display3d
```

1.3 graphics3d.library/GD_close_display3d()

```
NAME
     GD_close_display3d  --  erase all over the 3d scene viewing.

SYNOPSIS
     GD_close_display3d(in)
                        A0
     void GD_close_display3d(struct ambient3d *);

FUNCTION
     erase all that it was open and defined with GD_display3d included
     all objects of this scene.

INPUTS
     in = pointer to a ambient3d structure that you want delete.
          If this pointer is 0 than it do nothing.

RESULT

BUGS
     anyone note, if you find any tell me.

NOTES
     use it tipically at the end of the programm to erase all that is in
     relation with this library.

SEE ALSO
     GD_display3d
```

1.4 graphics3d.library/GD_changeviewmode()

```
NAME
     GD_changeviewmode  --  change the view mode of all objects

SYNOPSIS
     esi=GD_changeviewmode(in, modo, b_col)
                           A0  D0    D1
     LONG GD_changeviewmode(struct ambient3d *,LONG,LONG);

FUNCTION
     change simultaneously the view mode of ALL objects defined in the
     scene3d, for now is possible only this three view modes :
     wire frame, solid shading and flat shading.

INPUTS
     in    = pointer to ambient3d structure of the 3d scene over there
             you want work.
             It must be greater than 0 otherwise the result is undefined.
     modo  = new view mode :
             0 -> wire frame               (use macro WIREF)
             1 -> flat shading             (use macro FLAT)
             2 -> solid shading            (use macro SOLID)
     b_col = color register n# to use for the border of polygon of
             objects.
             If it is minor than 0 than no border.

RESULT
     esi = result , if different than 0 all ok, if equal to 0 than error
           operation aborted.
BUGS
     anyone note, if you find any tell me.

NOTES

SEE ALSO
     GD_changeviewmodeobj
```

1.5 graphics3d.library/GD_changeviewmodeobj()

```
NAME
     GD_changeviewmodeobj  --  change the view mode of selected object

SYNOPSIS
     esi=GD_changeviewmodeobj(in, modo)
                              A0  D0
     LONG GD_changeviewmodeobj(struct ambient3d *,LONG);

FUNCTION
     change the view mode of the actually selected object in the 3d scene
     considered.

INPUTS
     in    = pointer to ambient3d structure of the 3d scene over there
```

```
            you want work.
            It must be greater than 0 otherwise the result is undefined.
     modo  = new view mode :
            0 -> wire frame               (use macro WIREF)
            1 -> flat shading             (use macro FLAT)
            2 -> solid shading            (use macro SOLID)

RESULT
     esi = result , if different than 0 all ok, if equal to 0 than error
           operation aborted.
BUGS
     really it wasn't tested now but I hope that it can work.

NOTES

SEE ALSO
     GD_changeviewmode
```

1.6 graphics3d.library/GD_touchpalette()

```
NAME
     GD_touchpalette  --  create a shaded color palette

SYNOPSIS
     GD_touchpalette(in, fr, lr, init_color, lastcolor)
                     A0  D0  D1  A1          A2
     void GD_touchpalette(struct ambient3d *,LONG,LONG,struct rgbtype *,struct  ↩
        rgbtype *);

FUNCTION
     create a shaded color palette from two register color to use
     corrected the flat shading view mode.

INPUTS
     in         = pointer to ambient3d structure of the 3d scene over
                  there you want work.
                  It must be greater than 0 otherwise the result is
                  undefined.
     fr         = first color register to set.
     lr         = last color register to set.
     init_color = pointer to rgbtype structure with initial RGB value
                  of color.
     lastcolor  = pointer to rgbtype structure with final RGB value
                  of color.

RESULT

BUGS
     anyone note, if you find any tell me.

NOTES
     the color assegned to the object will use us reference to 'fr' for
     the flat shading and it will be the darker shade , 'lr' the ligther
     shade.
     It must be always used because the solid shading use the central
```

```
     shade us visualization color for polygons, using the object's color
     us reference to 'fr'.

SEE ALSO
     GD_rgb4
```

1.7 graphics3d.library/GD_moveforward()

```
NAME
     GD_moveforward  --  move the observer

SYNOPSIS
     GD_moveforward(in, dist)
                    A0  D0
     void GD_moveforward(struct ambient3d *,LONG);

FUNCTION
     move the observer of dist units forward to the point of view.

INPUTS
     in   = pointer to ambient3d structure of the 3d scene over there
            you want work.
            It must be greater than 0 otherwise the result is undefined.
     dist = n# of units of observer displacement ,it can be negative
            (move backward) but it must be in FIX POINT that is:
             integer value * 256 (or FIXV) +
            fractional value that will considered in 256 units.

RESULT

BUGS
     anyone note, if you find any tell me.

NOTES
     The fix point values are makes so :
     integer value*multiplier + fractional value.
     Where multiplier is the point position ane is always equal, in
     this library is = 256 (use macro FIXV).
     ex.: the floating point number 10.2 in fix point it will be :
             integer_portion * multiplier +
             multiplier / inverse_fractional_portion
             that is with multiplier equal to 256 (macro FIXV):
             (10 * 256) + (256 / (1/0.2)) = 2611
     that is too :
             float number * multiplier -> 10.2 * 256 = 2611

SEE ALSO
     GD_viewangle, GD_positioncamera
```

1.8 graphics3d.library/GD_viewangle()

```
NAME
     GD_viewangle  --  gira l'osservatore

SYNOPSIS
     GD_viewangle(in ,ax ,ay ,az )
                  A0  D0  D1  D2
     void GD_viewangle(struct ambient3d *,LONG,LONG,LONG);

FUNCTION
     permit of change the observer's angle of viewing.

INPUTS
     in = pointer to ambient3d structure of the 3d scene over there
          you want work.
          It must be greater than 0 otherwise the result is undefined.
     ax = rotation value on X axis of observer.
          it must be express in integer value (non fixpoint) and it
          is used in sexagesimal(??) degrees (0-90-180-360..).
     ay = rotation value on Y axis of observer.
          it must be express in integer value (non fixpoint) and it
          is used in sexagesimal(??) degrees (0-90-180-360..).
     az = rotation value on Z axis of observer.
          it must be express in integer value (non fixpoint) and it
          is used in sexagesimal(??) degrees (0-90-180-360..).

RESULT

BUGS
     anyone note, if you find any tell me.

NOTES

SEE ALSO
     GD_moveforward, GD_positioncamera
```

1.9 graphics3d.library/GD_frustum()

```
NAME
     GD_frustum  --  set the planes that delimit the visible space.

SYNOPSIS
     GD_frustum(in ,near ,far )
                A0  D0    D1
     void GD_frustum(struct ambient3d *,LONG,LONG);

FUNCTION
     set the distance of planes that delimt the field of view ,
     perpendicular at Z axis of observer.

INPUTS
     in   = pointer to ambient3d structure of the 3d scene over there
            you want work.
            It must be greater than 0 otherwise the result is undefined.
     near = integer value (not fixpoint) distance of plane that signal
```

```
            start of field of view of the defineted space.
     far  = integer value (not fixpoint) distance of plane that signal
            end of field of view of the defineted space.

RESULT

BUGS
     anyone note, if you find any tell me.

NOTES

SEE ALSO
     GD_clipmode
```

1.10 graphics3d.library/GD_createlightsource()

```
NAME
     GD_createlightsource  --  place the light source in the space

SYNOPSIS
     GD_createlightsource(in ,x ,y ,z )
                          A0  D0 D1 D2
     void GD_createlightsource(struct ambient3d *,LONG,LONG,LONG);

FUNCTION
     create and place a light source in the space.

INPUTS
     in = pointer to ambient3d structure of the 3d scene over there
          you want work.
          It must be greater than 0 otherwise the result is undefined.
     x  = X coordinate of the light as regards to space origin.
          Value in fix point (see notes of GD_moveforward).
     y  = Y coordinate of the light as regards to space origin.
          Value in fix point (see notes of GD_moveforward).
     z  = Z coordinate of the light as regards to space origin.
          Value in fix point (see notes of GD_moveforward).

RESULT

BUGS
     I suspect that not place corrected the light, I must verify it
     better.

NOTES
     Really this function do not create the light but only place it,
     because it can exist only one and is present from the use of
     GD_display3d otherwise the flatshading can't run from the begging.
     In the future pherhaps I can permit to use more than one and at
     this monent this function will can really create the light souces.

SEE ALSO
     GD_ambientlight
```

1.11 graphics3d.library/GD_ambientlight()

```
NAME
     GD_ambientlight  --  set the intensity of ambient light

SYNOPSIS
     GD_ambientlight(in ,inte )
                     A0  D0
     void GD_ambientlight(struct ambient3d *,LONG);

FUNCTION
     set the intensity of ambient light but NOT the color.

INPUTS
     in   = pointer to ambient3d structure of the 3d scene over there
            you want work.
            It must be greater than 0 otherwise the result is undefined.
     inte = intensity value in fixpoint (see notes of GD_moveforward).

RESULT

BUGS

NOTES
     this function has effect only if you using the flat shading for now.

SEE ALSO
     GD_createlightsource
```

1.12 graphics3d.library/GD_positioncamera()

```
NAME
     GD_positioncamera  --  observer  posizioning

SYNOPSIS
     GD_positioncamera(in ,x ,y ,z )
                       A0  D0 D1 D2
     void GD_positioncamera(struct ambient3d *,LONG,LONG,LONG);

FUNCTION
     place the observer as regards to space origin.

INPUTS
     in = pointer to ambient3d structure of the 3d scene over there
          you want work.
          It must be greater than 0 otherwise the result is undefined.
     x  = X coordinate of the observer as regards to space origin.
          Value in fix point (see notes of GD_moveforward).
     y  = Y coordinate of the observer as regards to space origin.
          Value in fix point (see notes of GD_moveforward).
     z  = Z coordinate of the observer as regards to space origin.
          Value in fix point (see notes of GD_moveforward).

RESULT
```

```
BUGS
     anyone note, if you find any tell me.

NOTES

SEE ALSO
     GD_moveforward, GD_viewangle
```

1.13 graphics3d.library/GD_aspectratio()

```
NAME
     GD_aspectratio  --  change the aspect ratio

SYNOPSIS
     GD_aspectratio(in ,ratio )
                    A0  D0
     void GD_aspectratio(struct ambient3d *,LONG);

FUNCTION
     change the aspect ratio of visualized scene , by default is equal
     to 1:1.

INPUTS
     in    = pointer to ambient3d structure of the 3d scene over there
             you want work.
             It must be greater than 0 otherwise the result is undefined.
     ratio = new value for aspect ratio so expressed, ex.: if 1:2 than
             is equal to 0.5.
             Value in fix point (see notes of GD_moveforward).

RESULT

BUGS
     anyone note, if you find any tell me.

NOTES

SEE ALSO
```

1.14 graphics3d.library/GD_clipmode()

```
NAME
     GD_clipmode  --  set a particular clip mode

SYNOPSIS
     GD_clipmode(in ,mode )
                 A0  D0
     void GD_clipmode(struct ambient3d *,LONG);

FUNCTION
     set the clipping node of the objects in the space ,between the two
```

```
     availables:
     ZETA PLANE : to clip the object it use the boundig box only on Z
                  axis as regards to planes near and far.
     FRUSTUM    : to clip the object it use the bounding box on all 3
                  axis , the Z as regards to planes near and far , the
                  X and the Y as regards to max limits of visualization
                  box.

INPUTS
     in   = pointer to ambient3d structure of the 3d scene over there
            you want work.
            It must be greater than 0 otherwise the result is undefined.
     mode = integer value new clip mode :
             0 - ZETA PLANE (use macro ZPLANE)
             1 - FRUSTUM    (use macro FRUSTUM)

RESULT

BUGS
     anyone note, if you find any tell me.

NOTES

SEE ALSO
     GD_frustum
```

1.15 graphics3d.library/GD_pickobj()

```
NAME
     GD_pickobj  --  given a point identify polygon and object.

SYNOPSIS
     idobj=GD_pickobj(in ,np ,x ,y )
                      A0  A1  D0 D1
     LONG GD_pickobj(struct ambient3d *,LONG *,LONG,LONG);

FUNCTION
     given a point on the visualization window (then 2D) identify inside
     which poligon and which object behind those visibles in that moment ,
     it are then if you not change the 2D point but change the viewing the
     result can change.

INPUTS
     in = pointer to ambient3d structure of the 3d scene over there
          you want work.
          It must be greater than 0 otherwise the result is undefined.
     np = pointer to an integer where place n# polygon find.
          It will are a integer value that start from 0 but it will
          valid only if the function result great than 0.
     x  = integer value (not fix point) co-ordinate X as regards to the
          visualization window of 3D scene.
     y  = integer value (not fix point) co-ordinate Y as regards to the
          visualization window of 3D scene.

RESULT
```

```
     integer value (not fix point) with univocal identifier of the object
     where is place the point given.
     If is equal to 0 the point is out of all object at that moment
     visualzed.

BUGS
     It can failed because not really found all point inside an object.

NOTES
     The used algoritm is totally empiric for speed reason and than not
     found all point inside a poligon, but certainly those find are
     INSIDE the found poligon.
     If anyone know algoritm more exact and can explain it to me will
     are welcome.
     For information this algoritm must can understand if a point is
     inside or not to a triangle or quadrilateral(this is not exential)
     but must do it faster as possible , because it will can tested
     hundred of polygon.

SEE ALSO
     GD_setobj, GD_getobj
```

1.16 graphics3d.library/GD_newobj()

```
NAME
     GD_newobj  --  create a new object

SYNOPSIS
     esi=GD_newobj(in ,name ,pol ,vert)
                   A0  A2    D0   D1
     LONG GD_newobj(struct ambient3d *,char *,LONG,LONG);

FUNCTION
     create and initialize the memories areas to generate a new object
     place it over the axis origin of the space and make it the actually
     selected.

INPUTS
     in   = pointer to ambient3d structure of the 3d scene over there
            you want work.
            It must be greater than 0 otherwise the result is undefined.
     name = pointer to a string (0x00 terminated) with name object.
     pol  = integer value with the total n# of poligons to assign at the
            object.
     vert = integer value with the total n# of vertices to assign at the
            object.

RESULT
     if esi equal to 0 oparation failed, otherwise all ok and the returned
     value is the identifier (univocal) of created object.

BUGS
     anyone note, if you find any tell me.

NOTES
```

```
     remember that the so created object have the vertices and polygons
     undefined than you must use the function GD_addobjvertex to
     define the vertices , GD_addobjpoly and GD_cattpoly to define the
     polygons and than GD_recalcobj to initialize correctly all internal
     value.

SEE ALSO
     GD_deleteobject ,GD_addobjvertex ,GD_addobjpoly ,GD_recalcobj
```

1.17 graphics3d.library/GD_deleteobject()

```
NAME
     GD_deleteobject  --  delete an object

SYNOPSIS
     GD_deleteobject(in)
                     A0
     void GD_deleteobject(struct ambient3d *);

FUNCTION
     erase an object and all memory areas that regards it than make
     the actually selected the previous, or if it is the first the next.

INPUTS
     in = pointer to ambient3d structure of the 3d scene over there
          you want work.
          It must be greater than 0 otherwise the result is undefined.

RESULT

BUGS
     anyone note, if you find any tell me.

NOTES
     if there is no defined object than it end to do nothing.

SEE ALSO
     GD_newobj
```

1.18 graphics3d.library/GD_addobjvertex()

```
NAME
     GD_addobjvertex  --  add a vertex to the current object

SYNOPSIS
     esi=GD_addobjvertex(in ,num ,x ,y ,z )
                         A0  D0   D1 D2 D3
     LONG GD_addobjvertex(struct ambient3d *,LONG,LONG,LONG,LONG);

FUNCTION
     insert a vertex in the object actually selected to the position
     pointing by num.
```

```
   INPUTS
        in  = pointer to ambient3d structure of the 3d scene over there
              you want work.
              It must be greater than 0 otherwise the result is undefined.
        num = integer number pointing what vertex you want insert.
              (#1->num=0 , #2->num=1 , ....).
        x   = integer value X co-ordinate of vertex to insert.
              Value in fix point (see notes of GD_moveforward).
        y   = integer value Y co-ordinate of vertex to insert.
              Value in fix point (see notes of GD_moveforward).
        z   = integer value Z co-ordinate of vertex to insert.
              Value in fix point (see notes of GD_moveforward).

   RESULT
        if esi greather than 0 than all ok otherwise inserting aborted.

   BUGS
        anyone note, if you find any tell me.

   NOTES
        for now this function really only modified the vertex, because
        GD_newobj create all vertex yet but with unsense value.
        In the future it is possible that it add really vertices.

   SEE ALSO
        GD_newobj, GD_deleteobject ,GD_addobjpoly
```

1.19 graphics3d.library/GD_addobjpoly()

```
NAME
        GD_addobjpoly -- a polygon adds to running object

SYNOPSIS
        esi=GD_addobjpoly(in, num,p1,p2,p3,p4)
                             A0 D0  D1 D2 D3 D4
        LONG GD_addobjpoly(struct ambient3d *, LONG, LONG, LONG, LONG, LONG);

FUNCTION
        inserts a polygon in the currently selected object to the
        position indicated from num.  For polygon the directory of the three
        or four apexes in hour sense agrees one or two that ne the chines
        compose or apexes that compose the point or the line.

INPUTS
        in  = pointer to ambient3d structure of the 3d scene over there
              you want work.
              It must be greater than 0 otherwise the result is undefined.
        num = entire number which polygon is becoming part.
              (# 1->num=0, #2->num=1,ecc.).
        p1  = number index #1 apex polygon on directory apexes object.
        p2  = number index #2 apex polygon on directory apexes object.
              In particular if this is equal to -1 then polygon with solo a
              point that is designs a single point.
        p3  = number index #3 apex polygon on directory apexes object.
```

```
              In particular if this is equal to -1 then polygon with solo
              two sides that is designs a segment.
        p4  = number index #4 apex polygon on directory apexes object.
              In particular if this is equal to -1 then polygon with solo
              three sides.

RESULT
        if 0 greater then all ok otherwise bankrupt insertion.

BUGS
        anyone note, if you find any tell me.

NOTES
        is worth the same note made for GD_addobjvertex, ciois in realta
        does not join to a polygon but modification one already present.
        At least for hour in future perhaps.

SEE ALSO
        GD_newobj, GD_deleteobject, GD_addobjvertex,
```

1.20 graphics3d.library/GD_cattpoly()

```
   NAME
        GD_cattpoly  --  change polygon attributes

   SYNOPSIS
        esi=GD_cattpoly(in ,num ,color ,twoside )
                        A0  D0   D1     D2
        LONG GD_cattpoly(struct ambient3d *,LONG,LONG,LONG);

   FUNCTION
        change the features of polygon pointing by num in the actually
        selected object.

   INPUTS
        in      = pointer to ambient3d structure of the 3d scene over there
                  you want work.
                  It must be greater than 0 otherwise the result is
                  undefined.
        num     = integer number pointing what polygon you want change.
                  (#1->num=0 , #2->num=1 , ....).
        color   = integer number for base color of polygon.
                  For FLAT visualization will use the next color to shade to
                  tones more light.
        twoside = integer number to show if poligon with two sides (1) or
                  with only one side (0).
                  If with two sides that is with back and front side than
                  it will are ever visible.
                  If with only one side than it will are visible only the
                  side that see to out of the object and to the observer.
                  This is a fast metod to reduce the n# of polygons in the
                  3d scene.

   RESULT
        if esi greather than 0 than all ok otherwise inserting aborted.
```

```
BUGS
     anyone note, if you find any tell me.

NOTES

SEE ALSO
     GD_addobjpoly
```

1.21 graphics3d.library/GD_recalcobj()

```
NAME
     GD_recalcobj  --  recalc the fixed parameter of the object

SYNOPSIS
     GD_recalcobj(in )
                  A0
     void GD_recalcobj(struct ambient3d *);

FUNCTION
     recalc any parameter usually not variable of actually selected object
     (as the bounding box) it must be run only if it is change the
     co-ordinates of one or more vetices.

INPUTS
     in   = pointer to ambient3d structure of the 3d scene over there
            you want work.
            It must be greater than 0 otherwise the result is undefined.

RESULT

BUGS
     anyone note, if you find any tell me.

NOTES
     This functions it must be used ever after the and of definition of
     a new object.
     That is after use of GD_newobj,GD_addobjvertex,GD_addobjpoly.

SEE ALSO
     GD_newobj ,GD_addobjvertex ,GD_addobjpoly
```

1.22 graphics3d.library/GD_setobj()

```
NAME
     GD_setobj  --  set as actually selacted an object

SYNOPSIS
     esi=GD_setobj(in ,num )
                   A0  D0
     LONG GD_setobj(struct ambient3d *,LONG);
```

```
FUNCTION
     It set as actually selected object that pointing by identifier
     in num.

INPUTS
     in  = pointer to ambient3d structure of the 3d scene over there
           you want work.
           It must be greater than 0 otherwise the result is undefined.
     num = integer number with identifier if object that will be set.

RESULT
     if esi greather than 0 than all ok otherwise inserting aborted.

BUGS
     anyone note, if you find any tell me.

NOTES

SEE ALSO
     GD_getobj
```

1.23 graphics3d.library/GD_getobj()

```
NAME
     GD_getobj  --  return identifier of an object

SYNOPSIS
     id=GD_getobj(in)
                  A0
     LONG GD_getobj(struct ambient3d *);

FUNCTION
     return identifier of actually selected object.

INPUTS
     in = pointer to ambient3d structure of the 3d scene over there
          you want work.
          It must be greater than 0 otherwise the result is undefined.

RESULT
     if id greather than 0 than object's identifier otherwise no one
     actually selected.

BUGS
     anyone note, if you find any tell me.

NOTES

SEE ALSO
     GD_setobj
```

1.24 graphics3d.library/GD_close_display3d()

```
NAME
     GD_rgb4  --  change a register colour in the palette

SYNOPSIS
     GD_rgb4(in ,nr ,red ,green ,blue )
             A0  D0  D1   D2     D3
     void GD_rgb4(struct ambient3d *,LONG,LONG,LONG,LONG);

FUNCTION
     change the register colour point in the screen palette where is
     visualized the window with the 3d scene.

INPUTS
     in    = pointer to ambient3d structure of the 3d scene over there
             you want work.
             It must be greater than 0 otherwise the result is undefined.
     nr    = number of register colour to change.
     red   = new value for red (0-15).
     green = new value for green (0-15).
     blue  = new value for blue (0-15).

RESULT

BUGS
     anyone note, if you find any tell me.

NOTES

SEE ALSO
     GD_touchpalette
```

1.25 graphics3d.library/GD_paintframe()

```
NAME
     GD_paintframe  --  really paint all poligons

SYNOPSIS
     rast=GD_paintframe(in)
                        A0
     struct RastPort *GD_paintframe(struct ambient3d *);

FUNCTION
     really paint all poligons really visible in the current view but not
     visualized them.

INPUTS
     in = pointer to ambient3d structure of the 3d scene over there
          you want work.
          It must be greater than 0 otherwise the result is undefined.

RESULT
     pointer to RastPort used to paint the poligons (is not visible), it
     can be used as pointer for other graphics function if this used the
     layers (used for clipping) otherwise aspected a big crash.
```

```
     Moreover this rasport have as origin, width and height the orginal
     value setting with GD_display3d and not those eventually change
     with GD_clipbox.

BUGS
     anyone note, if you find any tell me.

NOTES
     To erase the hidden faces, before to paint the polygons it
     reorganize them on base of their average point Z distance from
     the observer .
     Unfortunately this algoritm can wrong on case of intersection.

SEE ALSO
     GD_newview ,GD_switch_rp
```

1.26 graphics3d.library/GD_newview()

```
NAME
     GD_newview  --  recalc the actual view of the 3d scene

SYNOPSIS
     GD_newview(in)
                A0
     void GD_newview(struct ambient3d *);

FUNCTION
     recalc the list of polygons really visible in the actual view than
     projet them on the plane projection.

INPUTS
     in = pointer to ambient3d structure of the 3d scene over there
          you want work.
          It must be greater than 0 otherwise the result is undefined.

RESULT

BUGS
     anyone note, if you find any tell me.

NOTES
     this function must be used if you want see the effect of
     trasformation on the object.
     After that you have run this you must run GD_paintframe() to paint
     the polygons and than GD_switch_rp() to visualized them.

SEE ALSO
     GD_paintframe ,GD_switch_rp
```

1.27 graphics3d.library/GD_switch_rp()

```
NAME
     GD_switch_rp  --  visualize the view painting with GD_paintframe()

SYNOPSIS
     GD_switch_rp(in)
                  A0
     void GD_switch_rp(struct ambient3d *);

FUNCTION
     visualize the view make with GD_paintframe and the addition make  ↩
                    after.

INPUTS
     in = pointer to ambient3d structure of the 3d scene over there
          you want work.
          It must be greater than 0 otherwise the result is undefined.

RESULT

BUGS
     anyone note, if you find any tell me.

NOTES
     To do the visualization I use the ClipBlit function to copy the
     rastport used by GD_paintframe to the rastport of visualization
     window, with this sistem I can eliminate all(almost) the flikering
     that there is if I paint the polygons directly on the rastport of
     window.
     I have try to use BltBitMap instead of ClipBlit because it appear
     to be faster , but on my machines sometimes make a beautiful crash.
     Than for now I use ClipBlit , but I accept suggest to resolve the
     problem.

SEE ALSO
     GD_paintframe ,GD_newview
```

1.28 graphics3d.library/GD_translateobject()

```
NAME
     GD_translateobject  --  relative move of an object's origin

SYNOPSIS
     GD_translateobject(in ,dx ,dy ,dz)
                        A0  D0  D1  D2
     void GD_translateobject(struct ambient3d *,LONG,LONG,LONG);

FUNCTION
     move the origin of actually selected object in relative mode as
     regards to the origin of 3d scene.

INPUTS
     in = pointer to ambient3d structure of the 3d scene over there
          you want work.
          It must be greater than 0 otherwise the result is undefined.
```

```
     dx = value of displacement object's origin on axis X.
          Value in fix point (see notes of GD_moveforward).
     dy = value of displacement object's origin on axis Y.
          Value in fix point (see notes of GD_moveforward).
     dz = value of displacement object's origin on axis Z.
          Value in fix point (see notes of GD_moveforward).

RESULT

BUGS
     anyone note, if you find any tell me.

NOTES
     for any run of this function make a permanent move , ex:
     two displacement of two units on axis X are equal to one of 4.

SEE ALSO
     GD_positionobject
```

1.29 graphics3d.library/GD_positionobject()

```
NAME
     GD_positionobject  --  absolute move of an object's origin

SYNOPSIS
     GD_positionobject(in ,x ,y ,z )
                       A0  D0 D1 D2
     void GD_positionobject(struct ambient3d *,LONG,LONG,LONG);

FUNCTION
     move the origin of actually selected object in absolute mode as
     regards to the origin of 3d scene.

INPUTS
     in = pointer to ambient3d structure of the 3d scene over there
          you want work.
          It must be greater than 0 otherwise the result is undefined.
     x  = new value of object's origin on axis X.
          Value in fix point (see notes of GD_moveforward).
     y  = new value of object's origin on axis Y.
          Value in fix point (see notes of GD_moveforward).
     z  = new value of object's origin on axis Z.
          Value in fix point (see notes of GD_moveforward).

RESULT

BUGS
     anyone note, if you find any tell me.

NOTES

SEE ALSO
     GD_translateobject
```

1.30 graphics3d.library/GD_scaleobject()

```
NAME
     GD_scaleobject  --  rescale an object

SYNOPSIS
     GD_scaleobject(in ,xscale_fact,yscale_fact,zscalefact)
                    A0  D0          D1          D2
     void GD_scaleobject(struct ambient3d *,LONG,LONG,LONG);

FUNCTION
     rescale the actually selected object as regards the axes of your
     origin but not permanently (have effect only on actual frame).

INPUTS
     in           = pointer to ambient3d structure of the 3d scene over
             there you want work.
             It must be greater than 0 otherwise the result is undefined.
     xscale_fact  = value of scale factor of object's axis X.
             Value in fix point (see notes of GD_moveforward).
     yscale_fact  = value of scale factor of object's axis Y.
             Value in fix point (see notes of GD_moveforward).
     zscale_fact  = value of scale factor of object's axis Z.
             Value in fix point (see notes of GD_moveforward).

RESULT

BUGS
     anyone note, if you find any tell me.

NOTES
     any time you use this function the scaling will are applied on the
     origin dimension of object, then if you want rescale more time the
     object the scale_factor must be change consequently.
     Ex: two rescale on axis X of two time is equal to only one of two
         to time (not 4 as it can appear).
     Note that it is really for combination of scale and rotation ,that is
     for gradual varation it necessary reapply both trasformation with
     their value to the object before run the GD_newview.

SEE ALSO
     GD_rotateobject
```

1.31 graphics3d.library/GD_rotateobject()

```
NAME
     GD_rotateobject  --  rotate an object

SYNOPSIS
     GD_rotateobject(in ,angle_x ,angle_y ,angle_z)
                     A0  D0       D1       D2
     void GD_rotateobject(struct ambient3d *,LONG,LONG,LONG);

FUNCTION
```

```
     rotate the actually selected object as regards the axes of your
     origin but not permanently (have effect only on actual frame).

INPUTS
     in      = pointer to ambient3d structure of the 3d scene over there
               you want work.
               It must be greater than 0 otherwise the result is
               undefined.
     angle_x = integer (not fix point) value sexagesimal degrees that
               tell the rotation angle on object's X axis.
     angle_y = integer (not fix point) value sexagesimal degrees that
               tell the rotation angle on object's Y axis.
     angle_z = integer (not fix point) value sexagesimal degrees that
               tell the rotation angle on object's Z axis.

RESULT

BUGS
     anyone note, if you find any tell me.

NOTES
     this trasformation is always refered to the original position of
     object as the GD_scaleobject.
     For major explanation see NETES of GD_scaleobject.

SEE ALSO
     GD_scaleobject
```

1.32 graphics3d.library/GD_clipbox()

```
NAME
     GD_clipbox  --  change the size of visualzition box.

SYNOPSIS
     esi=GD_clipbox(in ,minx ,miny ,dx ,dy )
                    A0  D0    D1    D2  D3
     LONG GD_clipbox(struct ambient3d *,LONG,LONG,LONG,LONG)

FUNCTION
     change the dimensions of the box that delimit the visualization
     area of 3d scene.

INPUTS
     in   = pointer to ambient3d structure of the 3d scene over there
            you want work.
            It must be greater than 0 otherwise the result is undefined.
     minx = X co-ordinate of box left upper edge, as regards to the
            visualization window.
     miny = Y co-ordinate of box left upper edge, as regards to the
            visualization window.
     dx   = width value of box.
     dy   = height value of box.

RESULT
     if esi greather than 0 than all ok otherwise change aborted.
```

```
BUGS
     anyone note, if you find any tell me.

NOTES
     warning, you can't exceed the orginal dimension of visualization
     box defined by GD_display3d otherwise it can crash the system.
     This function don't make any verify for this.

SEE ALSO
     GD_display3d
```

1.33 graphics3d.library/GD_over()

```
NAME
     GD_over  --  change the draw mode in the hidden rastport

SYNOPSIS
     GD_over(in, mod )
             A0  D0
     void GD_over(struct ambient3d *,LONG);

FUNCTION
     change the draw mode in the rastport used by GD_paintframe but not
     influence it.

INPUTS
     in  = pointer to ambient3d structure of the 3d scene over there
           you want work.
           It must be greater than 0 otherwise the result is undefined.
     mod = new draw mode between this :
             0 = JAM1 (use macro JAM1)
             1 = JAM2 (use macro JAM2)
             2 = COMPLEMENT (use macro COMPLEMENT)
             4 = INVERSVID (use macro INVERSVID)

RESULT

BUGS
     anyone note, if you find any tell me.

NOTES

SEE ALSO
```

1.34 graphics3d.library/GD_cascene()

```
NAME
        GD_cascene -- it varies some parameters of visualization of scene 3d

SYNOPSIS
```

```
        GD_cascene(in, new)
                   A0  A1
        LONG GD_cascene(struct ambient3d *, struct tag3d *);

FUNCTION
        Permette to vary some parameters of visualization of the defined
        scene 3d with dislplay3d.

INPUTS
        in  = pointer to ambient3d structure of the 3d scene over there
              you want work.
              It must be greater than 0 otherwise the result is undefined.
        new = pointer to Array of structures tag3d with new parameters,
              works in the same way of the TagList implemented in the system
              bookcases.
              For the moment the possible values are:
              CS_PROJECT - type of projection to use, usable values:
                           Perspective PROSP_P=proiezione (the current one).
                           PARAL_P=proiezione parallel (experimental).
              CS_SBUFF   - still not implemented.
              CS_GCOLOR  - n # registry color background scene 3d (default
                           n#0).
              CS_VDIST   - new value (entire not fix) for distance between
                           observer and plan of projection.
              CS_NPX0    - val. entire (not fix) with new origin X box in
                           the window.
              CS_NPY0    - val. entire (not fix) with new origin Y box in
                           the window.

RESULT
        0 equal if to no carried out variation, if > 0 then indicate number
        of carried out variations.

BUGS
        anyone note, if you find any tell me.

NOTES
        Trattare the Array of structures tag3d exactly as the Array of said
        TagItem structures also tag list implemented in the operating system
        Amiga from the 2.0 in then.  The last structure of the Array must be
        empty and must have as first element constant END_T.
        NOT TO DIRECTLY USE NEVER IN THE FIRST VALUE THE LABEL THE ALWAYS
        USED NUMBERS BUT THAT INDICATE THEM.

SEE ALSO
```

1.35 graphics3d.library/GD_fix2int()

```
NAME
        GD_fix2int -- a number fix point in an entire one converts.

SYNOPSIS
        GD_fix2int(in, out)
                   A0  A1
```

```
        LONG GD_fix2int(LONG *, LONG *)

FUNCTION
        converts a number fix point in the format of the bookcase in an
        entire one to 32bit approximating to the entire one piu' close.

INPUTS
        in  = pointer to a 32 bit entire with value fix point converting
        out = pointer to a 32 bit entire where to put turned out.

RESULT
        0 equal if to all ok if various from 0 not modified error and out.

BUGS
        anyone note, if you find any tell me.

NOTES

SEE ALSO
        GD_fix2sfl, GD_fix2dfl
```

1.36 graphics3d.library/GD_fix2sfl()

```
NAME
        GD_fix2sfl -- a number fix point in a single float converts.

SYNOPSIS
        GD_fix2sfl(in, out)
                   A0  A1
        LONG GD_fix2sfl(LONG *, float *);

FUNCTION
        converts a number fix point in the format of the bookcase in
        a float in single precision.

INPUTS
        in  = pointer to a 32 bit entire with value fix point converting
        out = pointer to number single float where to put turned out.

RESULT
        0 equal if to all ok if various from 0 not modified error and out.

BUGS
        anyone note, if you find any tell me.

NOTES

SEE ALSO
        GD_fix2int, GD_fix2dfl
```

1.37 graphics3d.library/GD_fix2dfl()

```
NAME
        GD_fix2dfl -- a number fix point in a double float converts.

SYNOPSIS
        GD_fix2dfl(in, out)
                   A0  A1
        LONG GD_fix2dfl(LONG *, double *);

FUNCTION
        converts a number fix point in the format of the bookcase in a
        float in double precision.

INPUTS
        in  = pointer to a 32 bit entire with value fix point converting
        out = pointer to number double float where to put turned out.

RESULT
        0 equal if to all ok if various from 0 not modified error and out.

BUGS
        anyone note, if you find any tell me.

NOTES

SEE ALSO
        GD_fix2sfl, GD_fix2int
```

1.38 graphics3d.library/GD_int2fix()

```
NAME
        GD_int2fix -- an entire one in a number in fix point converts.

SYNOPSIS
        GD_int2fix(in, out)
                   A0  A1
        LONG GD_int2fix(LONG *, LONG *);

FUNCTION
        converts an entire one to 32bit in a number fix point in the format
        demanded from the bookcase.

INPUTS
        in  = pointer to a 32 bit entire converting
        out = pointer to a 32 bit entire where to put number in fix point
              deliberate.

RESULT
        0 equal if to all ok if various from 0 not modified error and out.

BUGS
        anyone note, if you find any tell me.
```

```
NOTES

SEE ALSO
        GD_sfl2fix, GD_dfl2fix
```

1.39 graphics3d.library/GD_sfl2fix()

```
NAME
        GD_sfl2fix -- a single float converts in a number in fix point.

SYNOPSIS
        GD_sfl2fix(in, out)
                   A0  A1
        LONG GD_sfl2fix(float *, LONG *);

FUNCTION
        converts a number float in single precision in a number fix
        point in the format demanded from the bookcase.

INPUTS
        in  = pointer to number float to convert
        out = pointer to entire to 32 bit where to put number in fix point
              deliberate.

RESULT
        0 equal if to all ok if various from 0 not modified error and out.

BUGS
        anyone note, if you find any tell me.

NOTES

SEE ALSO
        GD_int2fix, GD_dfl2fix
```

1.40 graphics3d.library/GD_dfl2fix()

```
NAME
        GD_dfl2fix -- a double float converts in a number in fix point.

SYNOPSIS
        GD_dfl2fix(in, out)
                   A0  A1
        LONG GD_dfl2fix(double *, LONG *);

FUNCTION
        converts a number float in double precision in a number fix
        point in the format demanded from the bookcase.

INPUTS
```

```
        in  = pointer to number double float to convert
        out = pointer to 32 bit inumber where to put number in fix point
              deliberate.

RESULT
        0 equal if to all ok if various from 0 not modified error and out.

BUGS
        anyone note, if you find any tell me.

NOTES

SEE ALSO
        GD_sfl2fix, GD_int2fix
```