INTERNATIONAL STANDARD

ISO/IEC 9593-4

First edition
1991-12-15

AMENDMENT 1
1994-05-01

Information technology — Computer graphics — Programmer's Hierarchical Interactive Graphics System (PHIGS) language bindings —

Part 4:
C

AMENDMENT 1

Technologies de l'information — Infographie — Interfaces langage entre un programme d'application et son support graphique —

Partie 4: C

AMENDEMENT 1

Reference number
ISO/IEC 9593-4:1991/Amd.1:1994(E)

Contents

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Amendment 1 to International Standard ISO/IEC 9593-4:1991 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*.

Introduction

Replace the current text with the following:

Part 1 of the Programmer's Hierarchical Interactive Graphics System (PHIGS) functional description is registered as ISO/IEC 9592-1:1989. Part 1 is extended by part 4, ISO/IEC 9592-4:1991 to incorporate the effects of lighting, shading and other properties important for the display of surfaces and multidimensional data. The purpose of this part of ISO/IEC 9593 is to define a standard binding for the C computer programming language to the functionality defined in ISO/IEC 9592-1:1989 and ISO/IEC 9592-4:1991.

This page intentionally left blank

Information technology — Computer graphics — Programmer's Hierarchical Interactive Graphics System (PHIGS) language bindings —

Part 4:
C

AMENDMENT 1

1 Scope

Replace the text with the following:

Part 1 of the Programmer's Hierarchical Interactive Graphics System (PHIGS), ISO/IEC 9592-1, and part 4 (PHIGS PLUS), ISO/IEC 9592-4, specify a language independent nucleus of a graphics system. For integration into a programming language, PHIGS and PHIGS PLUS are embedded in a language dependent layer obeying the particular conventions of that language. This part of ISO/IEC 9593 specifies such a language dependent layer for the C language.

2 Normative references

Add the following reference:

ISO/IEC 9592-4:1992, *Information processing systems — Computer graphics — Programmer's Hierarchical Interactive Graphics System (PHIGS) — Part 4: Plus Lumière und Surfaces, PHIGS PLUS.*

3 The C language binding of PHIGS

Change the clause heading to **The C language binding of PHIGS and PHIGS PLUS.**

3.1 Conformance

Replace the last two paragraphs with the following text:

In order to conform, an implementation of the C binding of PHIGS shall implement those functions specified in ISO/IEC 9592-1. The implementation shall make visible all of the declarations in the C binding specified in clauses 5 to 7 in this part of ISO/IEC 9593.

In order to conform, an implementation of the C binding of PHIGS PLUS shall implement those functions specified in ISO/IEC 9592-1 and also those functions specified in ISO/IEC 9592-4. The implementation shall make visible all of the declarations in the C binding specified in clauses 5 to 10 in this part of ISO/IEC 9593.

Thus, for example, the syntax of the function names shall be precisely as specified in this part of ISO/IEC 9593 and the parameters shall be of the data types stated in this part of ISO/IEC 9593.

3.2 Functions versus macros

No changes.

3.3 Character strings

No changes.

3.4 Function identifiers

No changes.

3.5 Registration

No changes.

3.6 Identifiers for graphical items

No changes.

3.7 Return values

No changes.

3.8 Header files

No changes.

3.9 Memory management

No changes.

3.9.1 Inquiry functions which return simple lists

No changes.

3.9.2 Inquiry functions which return complex data structures

No changes.

3.9.3 Meaning of the size of an element

No changes.

3.10 Inquiries returning structure elements

No changes.

3.11 Error handling

3.11.1 Application defined error handlers

No changes.

3.11.2 Error codes

No changes.

3.11.3 C specific PHIGS errors

Change the heading to **C binding specific errors** *and add the following binding specific errors:*

2207 `Ignoring function, two or more vertices do not contain the same type of data`
Is issued when an output primitive is created in which one or more of the vertices are specified with a particular type of data, such as normals, and other vertices are specified without the same type of data.

3.12 Storage of two-dimensional data

No changes.

3.12.1 Storage of matrices

No changes.

3.12.2 Storage of colour arrays

Change the heading to be **Storage of arrays** *and replace the text with the following:*

The entries of the `Ppat_rep` and `Px_array` data types (where x is one of `colrv`, `colrv_ctrl_point`, `ctrl_point3`, `data_ctrl_point`, `edge_flag`, `facet`, `float`, `vertex`, and `vertex3`) shall be stored such that the `(i,j)`th element in the array is given by the following expression:

```
i + j * DX
```

where:

`i = 0,..., DX-1,`
`j = 0,..., DY-1,`
`DX` = the `size_x` value of the array's `Pint_size` field, and
`DY` = the `size_y` value of the array's `Pint_size` field.

3.13 Data type descriptions

Add the clause with the following text:

The `Pedge_flag_array` data type defines an edge visibility flag for each edge of the quadrilateral mesh. The edge visibility flags are arranged in a *MXN* array of `Pedge_flag_pairs`. For each (*i,j*) `Pedge_flag_pair`, the first element speechifies the edge flag for the edge between the *i,j* and *i+1,j* vertices and the second element speechifies the edge flag for the edge between the *i,j* and *i,j+1* vertices. The first element of the (*M,j*) `Pedge_flag_pair`, where $1 \le j \le N$, and second element of the (*i, N*) `Pedge_flag_pair`, where $1 \le i \le M$, are not used.

The `Pedge_flag_triplet_list` data type consists of a list of `Pedge_flag_triplets`; one for each triangle in a triangle set. The first element of a `Pedge_flag_triplet` specifies the edge flag between the first and the second vertices of the triangle, the second element specifies the edge flag between the second and third vertices of the triangle. The third element specifies the edge flag between the third and first vertices of the triangle.

The `Px_set` data types (where x is one of `colrv`, `contour`, `edge_flag`, `float`, `vertex`, and `vertex3`) contain a list of `Px_lists`.

The `Px_set_list` data types (where x is one of `contour` and `edge_flag`) contains a list of `Px_sets`.

4 Tables

4.1 Abbreviation policy for construction of identifiers

No changes.

4.2 Table of abbreviations

Add alphabetically, to table 1, the following abbreviations:

Table 1 - Abbreviations ordered alphabetically

Word or Phrase	Abbreviation
activation	act
ambient	amb
approximation	approx
attenuation	atten
between	NULL
characteristics	chars
coefficient	coeff
concentration	conc
constant	const
compute	comp
criteria	crit
culling	cull
deactivate	deact
deactivation	deact
deviation	dev
diffuse	diff
directional	dir
distinguishing	disting
exponent	exp
geometric	geom
homogeneous	homo
isoparametric	isoparam
parametric	param
positional	pos
product	prod
properties	props
quadrilateral	quad

Table 1 - Abbreviations ordered alphabetically

Word or Phrase	Abbreviation
reflectance	refl
rendering	rend
shading	shad
specular	spec
subdivision	subd
triangle	tri
triangular	tri
trimming	trim
with	NULL

4.3 Function names

4.3.1 List ordered alphabetically by bound name

In table 2 change the column heading PHIGS Name *to* PHIGS and PHIGS PLUS Name *and merge alphabetically by bound name, the following function names:*

Table 2 - Function names ordered by bound name

C Name	PHIGS and PHIGS PLUS Name
pcell_array3_plus	CELL ARRAY 3 PLUS
pfill_area_set_data	FILL AREA SET WITH DATA
pfill_area_set3_data	FILL AREA SET 3 WITH DATA
pinq_b_spline_surf_facs	INQUIRE B-SPLINE SURFACE FACILITIES
pinq_colr_map_facs	INQUIRE COLOUR MAPPING FACILITIES
pinq_colr_map_method_facs	INQUIRE COLOUR MAPPING METHOD FACILITIES
pinq_colr_map_rep	INQUIRE COLOUR MAPPING REPRESENTATION
pinq_colr_map_st	INQUIRE COLOUR MAPPING STATE
pinq_curve_facs	INQUIRE CURVE FACILITIES
pinq_data_map_facs	INQUIRE DATA MAPPING FACILITIES
pinq_data_map_rep	INQUIRE DATA MAPPING REPRESENTATION
pinq_depth_cue_facs	INQUIRE DEPTH CUE FACILITIES
pinq_depth_cue_rep	INQUIRE DEPTH CUE REPRESENTATION
pinq_direct_colr_model_facs	INQUIRE DIRECT COLOUR MODEL FACILITIES
pinq_dyns_ws_plus	INQUIRE DYNAMICS OF WORKSTATION PLUS
pinq_edge_rep_plus	INQUIRE EDGE REPRESENTATION PLUS
pinq_int_facs_plus	INQUIRE INTERIOR FACILITIES PLUS
pinq_int_rep_plus	INQUIRE INTERIOR REPRESENTATION PLUS
pinq_light_source_facs	INQUIRE LIGHT SOURCE FACILITIES
pinq_light_source_rep	INQUIRE LIGHT SOURCE REPRESENTATION
pinq_line_facs_plus	INQUIRE POLYLINE FACILITIES PLUS
pinq_line_rep_plus	INQUIRE POLYLINE REPRESENTATION PLUS
pinq_list_colr_map_inds	INQUIRE LIST OF COLOUR MAPPING INDICES

Table 2 - Function names ordered by bound name

C Name	PHIGS and PHIGS PLUS Name
pinq_list_data_map_inds	INQUIRE LIST OF DATA MAPPING INDICES
pinq_list_depth_cue_inds	INQUIRE LIST OF DEPTH CUE INDICES
pinq_list_light_source_inds	INQUIRE LIST OF LIGHT SOURCE INDICES
pinq_list_param_surf_inds	INQUIRE LIST OF PARAMETRIC SURFACE INDICES
pinq_list_refl_inds	INQUIRE LIST OF REFLECTANCE INDICES
pinq_marker_rep_plus	INQUIRE POLYMARKER REPRESENTATION PLUS
pinq_param_surf_facs	INQUIRE PARAMETRIC SURFACE FACILITIES
pinq_param_surf_rep	INQUIRE PARAMETRIC SURFACE REPRESENTATION
pinq_pat_rep_plus	INQUIRE PATTERN REPRESENTATION PLUS
pinq_pred_colr_map_rep	INQUIRE PREDEFINED COLOUR MAPPING REPRESENTATION
pinq_pred_data_map_rep	INQUIRE PREDEFINED DATA MAPPING REPRESENTATION
pinq_pred_depth_cue_rep	INQUIRE PREDEFINED DEPTH CUE REPRESENTATION
pinq_pred_edge_rep_plus	INQUIRE PREDEFINED EDGE REPRESENTATION PLUS
pinq_pred_int_rep_plus	INQUIRE PREDEFINED INTERIOR REPRESENTATION PLUS
pinq_pred_light_source_rep	INQUIRE PREDEFINED LIGHT SOURCE REPRESENTATION
pinq_pred_line_rep_plus	INQUIRE PREDEFINED POLYLINE REPRESENTATION PLUS
pinq_pred_marker_rep_plus	INQUIRE PREDEFINED POLYMARKER REPRESENTATION PLUS
pinq_pred_param_surf_rep	INQUIRE PREDEFINED PARAMETRIC SURFACE REPRESENTATION
pinq_pred_pat_rep_plus	INQUIRE PREDEFINED PATTERN REPRESENTATION PLUS
pinq_pred_refl_rep	INQUIRE PREDEFINED REFLECTANCE REPRESENTATION
pinq_pred_text_rep_plus	INQUIRE PREDEFINED TEXT REPRESENTATION PLUS
pinq_refl_facs	INQUIRE REFLECTANCE FACILITIES
pinq_refl_rep	INQUIRE REFLECTANCE REPRESENTATION
pinq_rend_colr_model_facs	INQUIRE RENDERING COLOUR MODEL FACILITIES
pinq_text_rep_plus	INQUIRE TEXT REPRESENTATION PLUS
pinq_trim_curve_facs	INQUIRE TRIMMING CURVE FACILITIES
pinq_ws_st_table_length_plus	INQUIRE WORKSTATION STATE TABLE LENGTHS PLUS
pnon_uniform_b_spline_curve3	NON-UNIFORM B-SPLINE CURVE 3
pnon_uniform_b_spline_curve3_colr	NON-UNIFORM B-SPLINE CURVE 3 WITH COLOUR
pnon_uniform_b_spline_surf3	NON-UNIFORM B-SPLINE SURFACE 3
pnon_uniform_b_spline_surf3_data	NON-UNIFORM B-SPLINE SURFACE 3 WITH DATA
ppolyline_set3_colr	POLYLINE SET 3 WITH COLOUR
pquad_mesh_data	QUADRILATERAL MESH WITH DATA
pquad_mesh3_data	QUADRILATERAL MESH 3 WITH DATA
pset_back_data_map_ind	SET BACK DATA MAPPING INDEX
pset_back_data_map_method	SET BACK DATA MAPPING METHOD
pset_back_int_colr	SET BACK INTERIOR COLOUR
pset_back_int_ind	SET BACK INTERIOR INDEX
pset_back_int_shad_method	SET BACK INTERIOR SHADING METHOD
pset_back_int_style	SET BACK INTERIOR STYLE
pset_back_int_style_ind	SET BACK INTERIOR STYLE INDEX
pset_back_refl_ind	SET BACK REFLECTANCE INDEX
pset_back_refl_model	SET BACK REFLECTANCE MODEL
pset_back_refl_props	SET BACK REFLECTANCE PROPERTIES
pset_colr_map_ind	SET COLOUR MAPPING INDEX
pset_colr_map_rep	SET COLOUR MAPPING REPRESENTATION
pset_curve_approx_crit	SET CURVE APPROXIMATION CRITERIA

Table 2 - Function names ordered by bound name

C Name	PHIGS and PHIGS PLUS Name
pset_data_map_ind	SET DATA MAPPING INDEX
pset_data_map_method	SET DATA MAPPING METHOD
pset_data_map_rep	SET DATA MAPPING REPRESENTATION
pset_depth_cue_ind	SET DEPTH CUE INDEX
pset_depth_cue_rep	SET DEPTH CUE REPRESENTATION
pset_edge_colr	SET EDGE COLOUR
pset_edge_rep_plus	SET EDGE REPRESENTATION PLUS
pset_facet_cull_mode	SET FACET CULLING MODE
pset_facet_disting_mode	SET FACET DISTINGUISHING MODE
pset_int_colr	SET INTERIOR COLOUR
pset_int_rep_plus	SET INTERIOR REPRESENTATION PLUS
pset_int_shad_method	SET INTERIOR SHADING METHOD
pset_light_source_rep	SET LIGHT SOURCE REPRESENTATION
pset_light_source_st	SET LIGHT SOURCE STATE
pset_line_colr	SET POLYLINE COLOUR
pset_line_rep_plus	SET POLYLINE REPRESENTATION PLUS
pset_line_shad_method	SET POLYLINE SHADING METHOD
pset_marker_colr	SET POLYMARKER COLOUR
pset_marker_rep_plus	SET POLYMARKER REPRESENTATION PLUS
pset_of_fill_area_sets_data	SET OF FILL AREA SETS WITH DATA
pset_of_fill_area_sets3_data	SET OF FILL AREA SETS 3 WITH DATA
pset_param_surf_chars	SET PARAMETRIC SURFACE CHARACTERISTICS
pset_param_surf_ind	SET PARAMETRIC SURFACE INDEX
pset_param_surf_rep	SET PARAMETRIC SURFACE REPRESENTATION
pset_pat_rep_plus	SET PATTERN REPRESENTATION PLUS
pset_refl_ind	SET REFLECTANCE INDEX
pset_refl_model	SET REFLECTANCE MODEL
pset_refl_props	SET REFLECTANCE PROPERTIES
pset_refl_rep	SET REFLECTANCE REPRESENTATION
pset_rend_colr_model	SET RENDERING COLOUR MODEL
pset_surf_approx_crit	SET SURFACE APPROXIMATION CRITERIA
pset_text_colr	SET TEXT COLOUR
pset_text_rep_plus	SET TEXT REPRESENTATION PLUS
ptri_set_data	TRIANGLE SET WITH DATA
ptri_set3_data	TRIANGLE SET 3 WITH DATA
ptri_strip_data	TRIANGLE STRIP WITH DATA
ptri_strip3_data	TRIANGLE STRIP 3 WITH DATA

4.3.2 List ordered alphabetically by PHIGS function name

In table 3 change the clause heading to **List ordered alphabetically by PHIGS and PHIGS PLUS function name**, *change the table caption to* **Table 3 - Function names ordered by PHIGS and PHIGS PLUS names,**

change the column heading **PHIGS Name** *to* **PHIGS and PHIGS PLUS Name** *and merge alphabetically by function name the following function names:*

Table 3 - Function names ordered by PHIGS and PHIGS PLUS function name

PHIGS and PHIGS PLUS Name	C Name
CELL ARRAY 3 PLUS	pcell_array3_plus
FILL AREA SET WITH DATA	pfill_area_set_data
FILL AREA SET 3 WITH DATA	pfill_area_set3_data
INQUIRE B-SPLINE SURFACE FACILITIES	pinq_b_spline_surf_facs
INQUIRE COLOUR MAPPING FACILITIES	pinq_colr_map_facs
INQUIRE COLOUR MAPPING METHOD FACILITIES	pinq_colr_map_method_facs
INQUIRE COLOUR MAPPING REPRESENTATION	pinq_colr_map_rep
INQUIRE COLOUR MAPPING STATE	pinq_colr_map_st
INQUIRE CURVE FACILITIES	pinq_curve_facs
INQUIRE DATA MAPPING FACILITIES	pinq_data_map_facs
INQUIRE DATA MAPPING REPRESENTATION	pinq_data_map_rep
INQUIRE DEPTH CUE FACILITIES	pinq_depth_cue_facs
INQUIRE DEPTH CUE REPRESENTATION	pinq_depth_cue_rep
INQUIRE DIRECT COLOUR MODEL FACILITIES	pinq_direct_colr_model_facs
INQUIRE DYNAMICS OF WORKSTATION PLUS	pinq_dyns_ws_plus
INQUIRE EDGE REPRESENTATION PLUS	pinq_edge_rep_plus
INQUIRE INTERIOR FACILITIES PLUS	pinq_int_facs_plus
INQUIRE INTERIOR REPRESENTATION PLUS	pinq_int_rep_plus
INQUIRE LIGHT SOURCE FACILITIES	pinq_light_source_facs
INQUIRE LIGHT SOURCE REPRESENTATION	pinq_light_source_rep
INQUIRE LIST OF COLOUR MAPPING INDICES	pinq_list_colr_map_inds
INQUIRE LIST OF DATA MAPPING INDICES	pinq_list_data_map_inds
INQUIRE LIST OF DEPTH CUE INDICES	pinq_list_depth_cue_inds
INQUIRE LIST OF LIGHT SOURCE INDICES	pinq_list_light_source_inds
INQUIRE LIST OF PARAMETRIC SURFACE INDICES	pinq_list_param_surf_inds
INQUIRE LIST OF REFLECTANCE INDICES	pinq_list_refl_inds
INQUIRE PARAMETRIC SURFACE FACILITIES	pinq_param_surf_facs
INQUIRE PARAMETRIC SURFACE REPRESENTATION	pinq_param_surf_rep
INQUIRE PATTERN REPRESENTATION PLUS	pinq_pat_rep_plus
INQUIRE POLYLINE FACILITIES PLUS	pinq_line_facs_plus
INQUIRE POLYLINE REPRESENTATION PLUS	pinq_line_rep_plus
INQUIRE POLYMARKER REPRESENTATION PLUS	pinq_marker_rep_plus
INQUIRE PREDEFINED COLOUR MAPPING REPRESENTATION	pinq_pred_colr_map_rep
INQUIRE PREDEFINED DATA MAPPING REPRESENTATION	pinq_pred_data_map_rep
INQUIRE PREDEFINED DEPTH CUE REPRESENTATION	pinq_pred_depth_cue_rep
INQUIRE PREDEFINED EDGE REPRESENTATION PLUS	pinq_pred_edge_rep_plus
INQUIRE PREDEFINED INTERIOR REPRESENTATION PLUS	pinq_pred_int_rep_plus
INQUIRE PREDEFINED LIGHT SOURCE REPRESENTATION	pinq_pred_light_source_rep
INQUIRE PREDEFINED PARAMETRIC SURFACE REPRESENTATION	pinq_pred_param_surf_rep
INQUIRE PREDEFINED PATTERN REPRESENTATION PLUS	pinq_pred_pat_rep_plus
INQUIRE PREDEFINED POLYLINE REPRESENTATION PLUS	pinq_pred_line_rep_plus
INQUIRE PREDEFINED POLYMARKER REPRESENTATION PLUS	pinq_pred_marker_rep_plus
INQUIRE PREDEFINED REFLECTANCE REPRESENTATION	pinq_pred_refl_rep
INQUIRE PREDEFINED TEXT REPRESENTATION PLUS	pinq_pred_text_rep_plus

Table 3 - Function names ordered by PHIGS and PHIGS PLUS function name

PHIGS and PHIGS PLUS Name	C Name
INQUIRE REFLECTANCE FACILITIES	pinq_refl_facs
INQUIRE REFLECTANCE REPRESENTATION	pinq_refl_rep
INQUIRE RENDERING COLOUR MODEL FACILITIES	pinq_rend_colr_model_facs
INQUIRE TEXT REPRESENTATION PLUS	pinq_text_rep_plus
INQUIRE TRIMMING CURVE FACILITIES	pinq_trim_curve_facs
INQUIRE WORKSTATION STATE TABLE LENGTHS PLUS	pinq_ws_st_table_length_plus
NON-UNIFORM B-SPLINE CURVE 3	pnon_uniform_b_spline_curve3
NON-UNIFORM B-SPLINE SURFACE 3	pnon_uniform_b_spline_surf3
NON-UNIFORM B-SPLINE SURFACE 3 WITH DATA	pnon_uniform_b_spline_surf3_data
NON-UNIFORM B-SPLINE CURVE 3 WITH COLOUR	pnon_uniform_b_spline_curve3_colr
POLYLINE SET 3 WITH COLOUR	ppolyline_set3_colr
QUADRILATERAL MESH WITH DATA	pquad_mesh_data
QUADRILATERAL MESH 3 WITH DATA	pquad_mesh3_data
SET BACK DATA MAPPING INDEX	pset_back_data_map_ind
SET BACK DATA MAPPING METHOD	pset_back_data_map_method
SET BACK INTERIOR COLOUR	pset_back_int_colr
SET BACK INTERIOR INDEX	pset_back_int_ind
SET BACK INTERIOR SHADING METHOD	pset_back_int_shad_method
SET BACK INTERIOR STYLE INDEX	pset_back_int_style_ind
SET BACK INTERIOR STYLE	pset_back_int_style
SET BACK REFLECTANCE INDEX	pset_back_refl_ind
SET BACK REFLECTANCE MODEL	pset_back_refl_model
SET BACK REFLECTANCE PROPERTIES	pset_back_refl_props
SET COLOUR MAPPING INDEX	pset_colr_map_ind
SET COLOUR MAPPING REPRESENTATION	pset_colr_map_rep
SET CURVE APPROXIMATION CRITERIA	pset_curve_approx_crit
SET DATA MAPPING INDEX	pset_data_map_ind
SET DATA MAPPING METHOD	pset_data_map_method
SET DATA MAPPING REPRESENTATION	pset_data_map_rep
SET DEPTH CUE INDEX	pset_depth_cue_ind
SET DEPTH CUE REPRESENTATION	pset_depth_cue_rep
SET EDGE COLOUR	pset_edge_colr
SET EDGE REPRESENTATION PLUS	pset_edge_rep_plus
SET FACET CULLING MODE	pset_facet_cull_mode
SET FACET DISTINGUISHING MODE	pset_facet_disting_mode
SET INTERIOR COLOUR	pset_int_colr
SET INTERIOR REPRESENTATION PLUS	pset_int_rep_plus
SET INTERIOR SHADING METHOD	pset_int_shad_method
SET LIGHT SOURCE REPRESENTATION	pset_light_source_rep
SET LIGHT SOURCE STATE	pset_light_source_st
SET OF FILL AREA SETS WITH DATA	pset_of_fill_area_sets_data
SET OF FILL AREA SETS 3 WITH DATA	pset_of_fill_area_sets3_data
SET PARAMETRIC SURFACE CHARACTERISTICS	pset_param_surf_chars
SET PARAMETRIC SURFACE INDEX	pset_param_surf_ind
SET PARAMETRIC SURFACE REPRESENTATION	pset_param_surf_rep
SET PATTERN REPRESENTATION PLUS	pset_pat_rep_plus
SET POLYLINE COLOUR	pset_line_colr

Table 3 - Function names ordered by PHIGS and PHIGS PLUS function name

PHIGS and PHIGS PLUS Name	C Name
SET POLYLINE REPRESENTATION PLUS	pset_line_rep_plus
SET POLYLINE SHADING METHOD	pset_line_shad_method
SET POLYMARKER COLOUR	pset_marker_colr
SET POLYMARKER REPRESENTATION PLUS	pset_marker_rep_plus
SET REFLECTANCE INDEX	pset_refl_ind
SET REFLECTANCE MODEL	pset_refl_model
SET REFLECTANCE PROPERTIES	pset_refl_props
SET REFLECTANCE REPRESENTATION	pset_refl_rep
SET RENDERING COLOUR MODEL	pset_rend_colr_model
SET SURFACE APPROXIMATION CRITERIA	pset_surf_approx_crit
SET TEXT COLOUR	pset_text_colr
SET TEXT REPRESENTATION PLUS	pset_text_rep_plus
TRIANGLE SET WITH DATA	ptri_set_data
TRIANGLE SET 3 WITH DATA	ptri_set3_data
TRIANGLE STRIP WITH DATA	ptri_strip_data
TRIANGLE STRIP 3 WITH DATA	ptri_strip3_data

5 Type definitions

Change the clause heading to **C PHIGS type definitions.**

6 Macro definitions

Change the clause heading to **C PHIGS macro definitions.**

7 C PHIGS functions

Change the following function definitions:

INQUIRE GENERALIZED DRAWING PRIMITIVE 3

```
void pinq_gdp3 (

  Pint                  ws_type,        /* workstation type                */
  Pint                  gdp,            /* GDP function identifier         */
  Pint                  *err_ind,       /* OUT error indicator             */
  Pint                  *num_attr,      /* OUT number of attributes used   */
  Pattrs                attr[7]         /* OUT list of attributes used     */

);
```

INQUIRE GENERALIZED DRAWING PRIMITIVE

```
void pinq_gdp (

  Pint                  ws_type,        /* workstation type                */
  Pint                  gdp,            /* GDP function identifier         */
  Pint                  *err_ind,       /* OUT error indicator             */
  Pint                  *num_attr,      /* OUT number of attributes used   */
  Pattrs                attr[7]         /* OUT list of attributes used     */

);
```

8 C PHIGS PLUS type definitions

8.1 Mapping of PHIGS PLUS data types

ISO/IEC 9592-4 specifies a set of abstract data types beyond the types defined in ISO/IEC 9592-1. Table 4 - gives the mapping from those additional data types defined in ISO/IEC 9592-4 to the data types defined later in this binding.

Table 4 - Mapping of PHIGS PLUS data types to C

PHIGS PLUS data type		C binding data type
COLRV	colour value	Pcolrv
COLRVH	homogeneous colour value	Phomo_colr_rep_ptr
GCOLR	general colour	Pgcolr
NORM	normal vector	Pvec3
P3H	3D homogeneous point	Phomo_point3
P4H	4D homogeneous point	Phomo_point4

8.2 Modifications to PHIGS data types

Paspect **ASPECT**

```
typedef enum {
  /* start of PHIGS enumerations */
  ...,
  /* end of PHIGS enumerations */
  /* start of PHIGS PLUS enumerations */
  PASPECT_LINE_COLR,
  PASPECT_MARKER_COLR,
  PASPECT_TEXT_COLR,
  PASPECT_INT_COLR,
  PASPECT_EDGE_COLR,
  PASPECT_LINE_SHAD_METHOD,
  PASPECT_INT_SHAD_METHOD,
  PASPECT_DATA_MAP_METHOD,
  PASPECT_REFL_PROPS,
  PASPECT_REFL_MODEL,
  PASPECT_BACK_INT_STYLE,
  PASPECT_BACK_INT_STYLE_IND,
  PASPECT_BACK_INT_COLR,
  PASPECT_BACK_INT_SHAD_METHOD,
  PASPECT_BACK_DATA_MAP_METHOD,
  PASPECT_BACK_REFL_PROPS,
  PASPECT_BACK_REFL_MODEL,
  PASPECT_CURVE_APPROX_CRIT,
  PASPECT_SURF_APPROX_CRIT,
  PASPECT_PARAM_SURF_CHARS
  /* end of PHIGS PLUS enumerations */

} Paspect;
```

Pattrs **ATTRIBUTES USED**

```
typedef enum {
  /* start of PHIGS enumerations */
  ...,
  /* end of PHIGS enumerations */
  /* start of PHIGS PLUS enumerations */
  PATTR_REFL,
  PATTR_PARAM_SURF
  /* end of PHIGS PLUS enumerations */

} Pattrs;
```

Pelem_data **ELEMENT DATA**

```
typedef union {
  /* start of PHIGS element data */
  ...,
  /* end of PHIGS element data */
  /* start of PHIGS PLUS element data */
  struct Pelem_polyline_set3_colr {
    Pint             colr_type;          /* colour type                    */
    Pvertex3_set     vertex_data;        /* list of vertex lists           */
  } polyline_set3_colr;
  struct Pelem_fill_area_set3_data {
    Pint             colr_type;          /* colour type                    */
    Pfacet           *facet_data;        /* facet data                     */
    Pedge_flag_set   *edge_flags;        /* list of edge flag lists        */
    Pvertex3_set     vertex_data;        /* list of vertex lists           */
  } fill_area_set3_data;
  struct Pelem_fill_area_set_data {
    Pint             colr_type;          /* colour type                    */
    Pfacet           *facet_data;        /* facet data                     */
    Pedge_flag_set   *edge_flags;        /* list of edge flag lists        */
    Pvertex_set      vertex_data;        /* list of vertex lists           */
  } fill_area_set_data;
  struct Pelem_cell_array3_plus {
    Pparal           paral;              /* parallelogram P, Q, R          */
    Ppat_rep_plus    pattern;            /* pattern colour array           */
  } cell_array3_plus;
  struct Pelem_set_of_fill_area_sets3_data {
    Pint                  colr_type;     /* colour type                    */
    Pset_of_fill_area_sets_data *set_data;/* fill area sets data           */
    Pedge_flag_set_list   *edge_flags;   /* list of edge flag sets         */
    Pvertex3_list         vertex_data;   /* list of vertices               */
    Pcontour_set_list     contours;      /* list of contour sets           */
  } set_of_fill_area_sets3_data;
  struct Pelem_set_of_fill_area_sets_data {
    Pint                  colr_type;     /* colour type                    */
    Pset_of_fill_area_sets_data *set_data;/* fill area sets data           */
    Pedge_flag_set_list   *edge_flags;   /* list of edge flag sets         */
    Pvertex_list          vertex_data;   /* list of vertices               */
    Pcontour_set_list     contours;      /* list of contour sets           */
  } set_of_fill_area_sets_data;
  struct Pelem_tri_set3_data {
    Pint                      colr_type;    /* colour type                 */
    Ptri_set_data             *set_data;    /* triangle set data           */
    Pedge_flag_triplet_list *edge_flags;    /* list of triangle edge flag  */
                                            /* triplets                    */
    Pvertex3_list             vertex_data;  /* list of vertices            */
    Pvertex_ind_triplet_list triplets;      /* list of triangle vertex index */
                                            /* triplets                    */
```

```
} tri_set3_data;
struct Pelem_tri_set_data {
  Pint                      colr_type;    /* colour type                    */
  Ptri_set_data             *set_data;    /* triangle set data              */
  Pedge_flag_triplet_list *edge_flags;  /* list of triangle edge flag     */
                                        /* triplets                       */
  Pvertex_list              vertex_data;  /* list of vertices               */
  Pvertex_ind_triplet_list triplets;    /* list of triangle vertex index  */
                                        /* triplets                       */
} tri_set_data;
struct Pelem_tri_strip3_data {
  Pint                    colr_type;      /* colour type                    */
  Pfacet_list             *facet_data;    /* list of facet data             */
  Pedge_flag_list         *edge_flags;    /* list of edge flags             */
  Pvertex3_list           vertex_data;    /* list of vertex data            */
} tri_strip3_data;
struct Pelem_tri_strip_data {
  Pint                    colr_type;      /* colour type                    */
  Pfacet_list             *facet_data;    /* list of facet data             */
  Pedge_flag_list         *edge_flags;    /* list of edge flags             */
  Pvertex_list            vertex_data;    /* list of vertex data            */
} tri_strip_data;
struct Pelem_quad_mesh3_data {
  Pint                    colr_type;      /* colour type                    */
  Pfacet_array            *facet_data;    /* array of facet data            */
  Pedge_flag_array        *edge_flags;    /* array of edge flags            */
  Pvertex3_array          vertex_data;    /* array of vertex data           */
} quad_mesh3_data;
struct Pelem_quad_mesh_data {
  Pint                    colr_type;      /* colour type                    */
  Pfacet_array            *facet_data;    /* array of facet data            */
  Pedge_flag_array        *edge_flags;    /* array of edge flags            */
  Pvertex_array           vertex_data;    /* array of vertex data           */
} quad_mesh_data;
struct Pelem_non_uniform_b_spline_curve3 {
  Pcurve_geom_spline      geom_spline;    /* geometry spline                */
} non_uniform_b_spline_curve3;
struct Pelem_non_uniform_b_spline_curve3_colr {
  Pcurve_geom_spline      geom_spline;    /* geometry spline                */
  Pcurve_colr_spline      *colr_spline;   /* colour spline                  */
} non_uniform_b_spline_curve3_colr;
struct Pelem_non_uniform_b_spline_surf3 {
  Psurf_geom_spline       geom_spline;    /* geometry spline                */
  Ptrim_curve_loop_list   *trim_loops;    /* list of trimming curve loops   */
} non_uniform_b_spline_surf3;
struct Pelem_non_uniform_b_spline_surf3_data {
  Psurf_geom_spline       geom_spline;    /* geometry spline                */
  Ptrim_curve_loop_list   *trim_loops;    /* list of trimming curve loops   */
  Psurf_colr_spline       *colr_spline;   /* colour spline                  */
  Psurf_data_spline_list *data_spline;    /* data spline                    */
} non_uniform_b_spline_surf3_data;
```

```
  Pgcolr                colr;              /* general colour                */
  Pfacet_disting_mode   facet_disting_mode;/*facet distinguishing mode      */
  Pfacet_cull_mode      facet_cull_mode;   /* facet culling mode            */
  Pdata_map_rep         data_map_method;   /* data mapping method           */
  Prefl_prop            refl_prop;         /* reflectance property data     */
  struct Pelem_light_source_st {
    Pint_list              act_list;       /* activation list               */
    Pint_list              deact_list;     /* deactivation list             */
  } light_source_st;
  Pcurve_approx_crit  curve_approx_crit;   /* curve approximation criteria  */
                                           /* data record                   */
  Psurf_approx_crit   surf_approx_crit;    /* surface approximation criteria */
                                           /* data record                   */
  Pparam_surf_chars   param_surf_chars;    /* parametric surface            */
                                           /* characteristics data record   */
  /* end of PHIGS PLUS element data */

} Pelem_data;
```

NOTE — A NULL value for the `facet_data`, `edge_flags`, `set_data`, `colr_spline`, `trim_loops`, or `data_spline` fields indicates the corresponding optional data is not defined.

`Pelem_type` **ELEMENT TYPE**

```
typedef enum {
  /* start of PHIGS enumerations */
  ...,
  /* end of PHIGS enumerations */
  /* start of PHIGS PLUS enumerations */
  PELEM_POLYLINE_SET3_COLR,
  PELEM_FILL_AREA_SET3_DATA,
  PELEM_FILL_AREA_SET_DATA,
  PELEM_CELL_ARRAY3_PLUS,
  PELEM_SET_OF_FILL_AREA_SETS3_DATA,
  PELEM_SET_OF_FILL_AREA_SETS_DATA,
  PELEM_TRI_SET3_DATA,
  PELEM_TRI_SET_DATA,
  PELEM_TRI_STRIP3_DATA,
  PELEM_TRI_STRIP_DATA,
  PELEM_QUAD_MESH3_DATA,
  PELEM_QUAD_MESH_DATA,
  PELEM_NON_UNIFORM_B_SPLINE_CURVE3,
  PELEM_NON_UNIFORM_B_SPLINE_CURVE3_COLR,
  PELEM_NON_UNIFORM_B_SPLINE_SURF3,
  PELEM_NON_UNIFORM_B_SPLINE_SURF3_DATA,
  PELEM_DATA_MAP_IND,
  PELEM_REFL_IND,
  PELEM_BACK_INT_IND,
  PELEM_BACK_DATA_MAP_IND,
  PELEM_BACK_REFL_IND,
  PELEM_PARAM_SURF_IND,
  PELEM_LINE_COLR,
  PELEM_LINE_SHAD_METHOD,
  PELEM_MARKER_COLR,
  PELEM_TEXT_COLR,
  PELEM_FACET_DISTING_MODE,
  PELEM_FACET_CULL_MODE,
  PELEM_INT_COLR,
  PELEM_INT_SHAD_METHOD,
  PELEM_DATA_MAP_METHOD,
  PELEM_REFL_PROPS,
  PELEM_REFL_MODEL,
  PELEM_BACK_INT_STYLE,
  PELEM_BACK_INT_STYLE_IND,
  PELEM_BACK_INT_COLR,
  PELEM_BACK_INT_SHAD_METHOD,
  PELEM_BACK_DATA_MAP_METHOD,
  PELEM_BACK_REFL_PROPS,
  PELEM_BACK_REFL_MODEL,
  PELEM_LIGHT_SOURCE_ST,
  PELEM_EDGE_COLR,
```

```
    PELEM_CURVE_APPROX_CRIT,
    PELEM_SURF_APPROX_CRIT,
    PELEM_PARAM_SURF_CHARS,
    PELEM_REND_COLR_MODEL,
    PELEM_DEPTH_CUE_IND,
    PELEM_COLR_MAP_IND
    /* end of PHIGS PLUS enumerations */

} Pelem_type;
```

8.3 Implementation dependent PHIGS PLUS type definitions

The following type definitions may be modified by the implementation. If the implementation adds new fields to the unions, the convention specified in clause 5.3 shall be used for naming the new field with the following addition:

For the name of a new field in a PHIGS PLUS implementation dependent type, called "xxx" in clause 5.3, use "type."

`Pcolr_map_method` **COLOUR MAPPING METHOD**

```
typedef struct {
  Pint        method;                   /* colour mapping method          */
  union Pcolr_map_method_data {
    Pint      num_true_colrs;           /* number of true colours         */
    Pint      max_num_pseudo_colrs;     /* maximum number of pseudo       */
                                        /* colours                        */
    ...                                 /* implementation defined         */
  } data;

} Pcolr_map_method;
```

`Pcolr_map_rep` **COLOUR MAPPING REPRESENTATION**

```
typedef struct {

  Pint              method;              /* colour mapping method          */
  union Pcolr_map_rep_data {
    struct Pcolr_map_rep_pseudo {
      Pint          model;               /* colour model                   */
      Pfloat_list   weight_vector;       /* list of weight vector          */
      Pcolrv_list   colrs;               /* list of colour values          */
    } pseudo;
    struct Pcolr_map_rep_pseudo_N {
      Pint          model;               /* colour model                   */
      Pfloat_set    colrs;               /* list of lists of colour        */
                                         /* coordinates                    */
    } pseudo_N;
  ...                                    /* implementation defined         */
  } data;

} Pcolr_map_rep;
```

`Pcolr_map_st` **COLOUR MAPPING STATE**

```
typedef struct {

  Pint        method;                    /* colour mapping method          */
  union Pcolr_map_st_data {
    Pint      num_true_colrs;            /* number of available true       */
                                         /* colours                        */
    Pint      max_num_pseudo_colrs;      /* maximum number of pseudo       */
                                         /* colour entries                 */
    ...                                  /* implementation defined         */
  } data;

} Pcolr_map_st;
```

`Pcolr_rep_ptr` **COLOUR REPRESENTATION POINTER**

```
typedef union {
  Prgb              *rgb;              /* pointer to RGB colour values    */
  Pcieluv           *cieluv;           /* pointer to CIELUV colour values */
  Phls              *hls;              /* pointer to HLS colour values    */
  Phsv              *hsv;              /* pointer to HSV colour values    */
  Pdata             *unsupp;           /* pointer to unsupported colour   */
                                       /* values                          */
  ...                                  /* implementation defined          */
} Pcolr_rep_ptr;
```

Pcurve_approx_crit **CURVE APPROXIMATION CRITERIA**

```
typedef struct {
  Pint              type;               /* curve approximation criteria */
                                        /* type                         */
  union Pcurve_approx_crit_data {
    struct Pcurve_approx_const_param_subd {
      Pint          count;              /* count                        */
    } const_param_subd;
    struct Pcurve_approx_chordal_size_wc {
      Pfloat        approx;             /* approximation value          */
    } chordal_size_wc;
    struct Pcurve_approx_chordal_size_npc {
      Pfloat        approx;             /* approximation value          */
    } chordal_size_npc;
    struct Pcurve_approx_chordal_size_dc {
      Pfloat        approx;             /* approximation value          */
    } chordal_size_dc;
    struct Pcurve_approx_chordal_dev_wc {
      Pfloat        approx;             /* approximation value          */
    } chordal_dev_wc;
    struct Pcurve_approx_chordal_dev_npc {
      Pfloat        approx;             /* approximation value          */
    } chordal_dev_npc;
    struct Pcurve_approx_chordal_dev_dc {
      Pfloat        approx;             /* approximation value          */
    } chordal_dev_dc;
    struct Pcurve_approx_rel_wc {
      Pfloat        approx;             /* approximation value          */
    } rel_wc;
    struct Pcurve_approx_rel_npc {
      Pfloat        approx;             /* approximation value          */
    } rel_npc;
    struct Pcurve_approx_rel_dc {
      Pfloat        approx;             /* approximation value          */
    } rel_dc;
    Pdata           unsupp;             /* unsupported curve approximation*/
                                        /* criteria data                */
    ...                                 /* implementation defined       */
  } data;
} Pcurve_approx_crit;
```

Pdata_map_rep **DATA MAPPING REPRESENTATION**

```
typedef struct {
  Pint                    method;          /* data mapping method              */
  Psource_select_list     selectors;       /* list of source selectors         */
  union Pdata_map_rep_data {
    struct Pdata_map_rep_single_uniform {
      Pint                ind;             /* data value index                 */
      Pfloat              lower_limit;     /* lower range limit                */
      Pfloat              upper_limit;     /* upper range limit                */
      Pint                colr_type;       /* colour type                      */
      Pcolrv_list         colr_values;     /* list of colour values            */
    } single_uniform;
    struct Pdata_map_rep_single_non_uniform {
      Pint                ind;             /* data value index                 */
      Pfloat_list         range;           /* range boundaries                 */
      Pint                colr_type;       /* colour type                      */
      Pcolrv_list         colr_values;     /* list of colour values            */
    } single_non_uniform;
    struct Pdata_map_rep_bi_uniform {
      Pint                inds[2];         /* data value indices               */
      Pfloat              lower_limit_a;   /* lower limit of Ra range          */
      Pfloat              upper_limit_a;   /* upper limit of Ra range          */
      Pfloat              lower_limit_b;   /* lower limit of Rb range          */
      Pfloat              upper_limit_b;   /* upper limit of Rb range          */
      Pint                colr_type;       /* colour type                      */
      Pcolrv_set          colr_values;     /* list of colour value lists       */
    } bi_uniform;
    struct Pdata_map_rep_bi_non_uniform {
      Pint                inds[2];         /* data value indices               */
      Pfloat_list         range_a;         /* Ra range boundaries              */
      Pfloat_set          range_b;         /* Rb range boundaries              */
      Pint                colr_type;       /* colour type                      */
      Pcolrv_set          colr_values;     /* array of colour values           */
    } bi_non_uniform;
    ...                                    /* implementation defined           */
  } data;

} Pdata_map_rep;
```

NOTE — The indexing of data mapping values is 1 based.

`Phomo_colr_rep_ptr` **HOMOGENEOUS COLOUR REPRESENTATION POINTER**

```
typedef union {
  Phomo_rgb          *rgb;              /* pointer to homogeneous rgb     */
                                        /* colour value                   */
  Phomo_cieluv       *cieluv;           /* pointer to homogeneous cieluv  */
                                        /* colour value                   */
  Phomo_hls          *hls;              /* pointer to homogeneous hls     */
                                        /* colour value                   */
  Phomo_hsv          *hsv;              /* pointer to homogeneous hsv     */
                                        /* colour value                   */
  ...                                   /* implementation defined         */
} Phomo_colr_rep_ptr;
```

`Plight_source_rep` **LIGHT SOURCE REPRESENTATION**

```
typedef struct {

  Pint              type;               /* light source type              */
  union Plight_source_rep_data {
    struct Plight_source_rep_amb {
      Pgcolr        colr;               /* light source colour            */
    } amb;
    struct Plight_source_rep_dir {
      Pgcolr        colr;               /* light source colour            */
      Pvec3         dir;                /* light source direction         */
    } dir;
    struct Plight_source_rep_pos {
      Pgcolr        colr;               /* light source colour            */
      Ppoint3       pos;                /* light source position          */
      Pfloat        c1;                 /* light source attenuation       */
                                        /* coefficient                    */
      Pfloat        c2;                 /* light source attenuation       */
                                        /* coefficient                    */
    } pos;
    struct Plight_source_rep_spot {
      Pgcolr        colr;               /* light source colour            */
      Ppoint3       pos;                /* light source position          */
      Pvec3         dir;                /* light source direction         */
      Pfloat        concent_exp;        /* concentration exponent         */
      Pfloat        c1;                 /* light source attenuation       */
                                        /* coefficient                    */
      Pfloat        c2;                 /* light source attenuation       */
                                        /* coefficient                    */
      Pfloat        spread_angle;       /* light source spread angle      */
    } spot;
    ...                                 /* implementation defined         */
  } data;

} Plight_source_rep;
```

`Pparam_surf_chars` **PARAMETRIC SURFACE CHARACTERISTICS**

```
typedef struct {

  Pint              type;                 /* parametric surface          */
                                          /* characteristics type        */

  union Pparam_surf_chars_data {
    struct Pparam_surf_isoparam {
      Pplacement    placement;            /* curve placement: UNIFORM OVER  */
                                          /* SURFACE, UNIFORM BETWEEN KNOTS */
      Pint          num_u_curves;         /* curve count in u direction     */
      Pint          num_v_curves;         /* curve count in v direction     */
    } isoparam;
    struct Pparam_surf_level_mc {
      Ppoint3       origin;               /* origin point                   */
      Pvec3         dir;                  /* direction vector               */
      Pfloat_list   params;               /* parameter list                 */
    } level_mc;
    struct Pparam_surf_level_wc {
      Ppoint3       origin;               /* origin point                   */
      Pvec3         dir;                  /* direction vector               */
      Pfloat_list   params;               /* parameter list                 */
    } level_wc;
    Pdata           unsupp;               /* unsupported parametric         */
                                          /* surface characteristics data   */
    ...                                   /* implementation defined         */
  } data;

} Pparam_surf_chars;
```

`Prefl_prop` **REFLECTANCE PROPERTY**

```
typedef struct {

  Pint              type;              /* reflectance property type      */
  union Prefl_prop_data {
    struct Prefl_prop_simple_refl {
      Pfloat        amb_coeff;         /* ambient reflection             */
                                       /* coefficient                    */
      Pfloat        diff_coeff;        /* diffuse reflection             */
                                       /* coefficient                    */
      Pfloat        spec_coeff;        /* specular reflection            */
                                       /* coefficient                    */
      Pgcolr        spec_colr;         /* specular colour                */
      Pfloat        spec_exp;          /* specular exponent              */
    } simple_refl;
    Pdata           unsupp;            /* unsupported reflectance        */
                                       /* property data                  */
    ...                                /* implementation defined         */
  } data;

} Prefl_prop;
```

`Psurf_approx_crit` **SURFACE APPROXIMATION CRITERIA**

```
typedef struct {
  Pint              type;                 /* surface approximation criteria */
                                          /* type                           */
  union Psurf_approx_crit_data {
    struct Psurf_approx_const_param_subd {
      Pint          u_count;              /* u count                        */
      Pint          v_count;              /* v count                        */
    } const_param_subd;
    struct Psurf_approx_chordal_size_wc {
      Pfloat        u_approx;             /* u approximation value          */
      Pfloat        v_approx;             /* v approximation value          */
    } chordal_size_wc;
    struct Psurf_approx_chordal_size_npc {
      Pfloat        u_approx;             /* u approximation value          */
      Pfloat        v_approx;             /* v approximation value          */
    } chordal_size_npc;
    struct Psurf_approx_chordal_aize_dc {
      Pfloat        u_approx;             /* u approximation value          */
      Pfloat        v_approx;             /* v approximation value          */
    } chordal_size_dc;
    struct Psurf_approx_planar_dev_wc {
      Pfloat        approx;               /* approximation value            */
    } planar_dev_wc;
    struct Psurf_approx_planar_dev_npc {
      Pfloat        approx;               /* approximation value            */
    } planar_dev_npc;
    struct Psurf_approx_planar_dev_dc {
      Pfloat        approx;               /* approximation value            */
    } planar_dev_dc;
    struct Psurf_approx_rel_wc {
      Pfloat        approx;               /* approximation value            */
    } rel_wc;
    struct Psurf_approx_rel_npc {
      Pfloat        approx;               /* approximation value            */
    } rel_npc;
    struct Psurf_approx_rel_dc {
      Pfloat        approx;               /* approximation value            */
    } rel_dc;
    Pdata           unsupp;               /* unsupported surface            */
                                          /* approximation criteria data    */
    ...                                   /* implementation defined         */
  } data;

} Psurf_approx_crit;
```

8.4 Implementation independent PHIGS PLUS type definitions

`Pcolr_ctrl_point_array` **COLOUR CONTROL POINT ARRAY**

```
typedef struct {

  Prationality          rationality;    /* control point rationality     */
  Pint_size             dims;           /* dimensions of control point   */
                                        /* array                         */
  Pcolr_ctrl_point_ptr  colr_points;    /* array of colour control points */

} Pcolr_ctrl_point_array;
```

`Pcolr_ctrl_point_list` **COLOUR CONTROL POINT LIST**

```
typedef struct {

  Prationality          rationality;    /* control point rationality      */
  Pint                  num_points;     /* number of control points       */
  Pcolr_ctrl_point_ptr  points;         /* list of colour control points  */

} Pcolr_ctrl_point_list;
```

`Pcolr_ctrl_point_ptr` **COLOUR CONTROL POINT POINTER**

```
typedef union {

  Pcolr_rep_ptr         points;         /* pointer to colour points       */
  Phomo_colr_rep_ptr    homo_points;    /* pointer to homogeneous colour  */
                                        /* points                         */

} Pcolr_ctrl_point_ptr;
```

`Pcolrv` **COLOUR VALUE**

```
typedef union {

  Pint                  colr_ind;       /* colour index                   */
  Pcolr_rep             colr_rep;       /* colour representation          */

} Pcolrv;
```

`Pcolrv_array` **COLOUR VALUE ARRAY**

```
typedef struct {
  Pint_size          dims;          /* colour value array dimensions */
  Pcolrv_ptr         colrs;         /* array of colour values        */
} Pcolrv_array;
```

`Pcolrv_list` **COLOUR VALUE LIST**

```
typedef struct {
  Pint               num_colrs;     /* number of colours             */
  Pcolrv_ptr         colrs;         /* list of colour values         */
} Pcolrv_list;
```

`Pcolrv_ptr` **COLOUR VALUE POINTER**

```
typedef union {
  Pint               *colr_inds;    /* pointer to colour indices     */
  Pcolr_rep_ptr      colr_reps;     /* pointer to colour             */
                                    /* representations               */
} Pcolrv_ptr;
```

`Pcolrv_set` **COLOUR VALUE SET**

```
typedef struct {
  Pint               num_lists;     /* number of colour value lists  */
  Pcolrv_list        *colrs;        /* list of colour value lists    */
} Pcolrv_set;
```

`Pcontour_set` **CONTOUR SET**

```
typedef struct {
  Pint              num_lists;          /* number of contour lists      */
  Pint_list         *contours;          /* list of contour lists        */
} Pcontour_set;
```

`Pcontour_set_list` **CONTOUR SET LIST**

```
typedef struct {
  Pint              num_contour_sets;   /* number of contour sets       */
  Pcontour_set      *contours;          /* list of contour sets         */
} Pcontour_set_list;
```

`Pctrl_point3_array` **CONTROL POINT 3 ARRAY**

```
typedef struct {
  Prationality      rationality;        /* control point rationality    */
  Pint_size         dims;               /* control point array u,v      */
                                        /* dimensions                   */
  Ppoint3_ptr       points;             /* array of control points      */
} Pctrl_point3_array;
```

`Pctrl_point_list` **CONTROL POINT LIST**

```
typedef struct {
  Prationality      rationality;        /* control point rationality    */
  Pint              num_points;         /* number of control points     */
  Ppoint_ptr        points;             /* list of control points       */
} Pctrl_point_list;
```

`Pctrl_point3_list` **CONTROL POINT 3 LIST**

```
typedef struct {

  Prationality      rationality;        /* control point rationality      */
  Pint              num_points;         /* number of control points       */
  Ppoint3_ptr       points;             /* list of control points         */

} Pctrl_point3_list;
```

`Pcurve_colr_spline` **CURVE COLOUR SPLINE**

```
typedef struct {

  Pint                      order;      /* spline order                   */
  Pfloat_list               knots;      /* list of knot values            */
  Pint                      colr_type;  /* colour type                    */
  Pcolr_ctrl_point_list     colrs;      /* list of colour control points  */

} Pcurve_colr_spline;
```

`Pcurve_geom_spline` **CURVE GEOMETRY SPLINE**

```
typedef struct {

  Pint                  order;          /* spline order                   */
  Pfloat_list           knots;          /* knots                          */
  Pfloat                low_limit;      /* lower parameter range limit    */
  Pfloat                high_limit;     /* high parameter range limit     */
  Pctrl_point3_list     ctrl_points;    /* list of control points         */

} Pcurve_geom_spline;
```

Pdata_ctrl_point_array **DATA CONTROL POINT ARRAY**

```
typedef struct {
  Prationality      rationality;        /* control point rationality      */
  Pint_size         dims;               /* control point array u,v        */
                                        /* dimensions                     */
  Pint              points_dims;        /* control point dimensionality   */
  Pfloat            *ctrl_points;       /* array of control points        */
} Pdata_ctrl_point_array;
```

Pdepth_cue_mode **DEPTH CUE MODE**

```
typedef enum {
  PDEPTH_CUE_SUPPR,
  PDEPTH_CUE_ALLOWED
} Pdepth_cue_mode;
```

Pdepth_cue_mode_list **DEPTH CUE MODE LIST**

```
typedef struct {
  Pint              num_modes;          /* number of depth cue modes      */
  Pdepth_cue_mode   *modes;             /* list of depth cue modes        */
} Pdepth_cue_mode_list;
```

Pdepth_cue_rep **DEPTH CUE REPRESENTATION**

```
typedef struct {
  Pdepth_cue_mode   mode;               /* depth cue mode                 */
  Pfloat            ref_planes[2];      /* depth cue reference planes     */
                                        /* (min, max)                     */
  Pfloat            scale[2];           /* depth cue scale factors        */
                                        /* (min, max)                     */
  Pgcolr            gcolr;              /* depth cue colour               */
} Pdepth_cue_rep;
```

`Pdyn_ws_attrs_plus` **DYNAMICS OF WORKSTATION ATTRIBUTES PLUS**

```
typedef struct {

  Pdyn_mod          data_map;          /* data mapping representation   */
  Pdyn_mod          refl;              /* reflectance representation    */
  Pdyn_mod          param_surf;        /* parametric surface            */
                                       /* representation                */
  Pdyn_mod          light_source;      /* light source representation   */
  Pdyn_mod          depth_cue;         /* depth cue representation      */
  Pdyn_mod          colr_map;          /* colour mapping representation */

} Pdyn_ws_attrs_plus;
```

`Pedge_bundle_plus` **EDGE BUNDLE PLUS**

```
typedef struct {

  Pedge_flag        flag;              /* edge flag                     */
  Pint              type;              /* edgetype                      */
  Pfloat            width;             /* edgewidth scale factor        */
  Pgcolr            colr;              /* edge colour                   */

} Pedge_bundle_plus;
```

`Pedge_flag_array` **EDGE FLAG ARRAY**

```
typedef struct {

  Pint_size         dims;              /* edge flag array dimensions    */
  Pedge_flag_pair   *edge_flags;       /* array of edge flag pairs      */

} Pedge_flag_array;
```

`Pedge_flag_list` **EDGE FLAG LIST**

```
typedef struct {

  Pint              num_edges;         /* number of edges in list       */
  Pedge_flag        *edge_flags;       /* list of edge flags            */

} Pedge_flag_list;
```

`Pedge_flag_pair` **EDGE FLAG PAIR**

```
typedef Pedge_flag Pedge_flag_pair[2];
```

`Pedge_flag_set` **EDGE FLAG SET**

```
typedef struct {
  Pint              num_lists;          /* number of edge flag lists      */
  Pedge_flag_list   *edge_flags;        /* list of edge flag lists        */
} Pedge_flag_set;
```

`Pedge_flag_set_list` **EDGE FLAG SET LIST**

```
typedef struct {
  Pint              num_edge_flag_sets; /* number of edge flag sets       */
  Pedge_flag_set    *edge_flag_sets;    /* list of edge flag sets         */
} Pedge_flag_set_list;
```

`Pedge_flag_triplet` **EDGE FLAG TRIPLET**

```
typedef Pedge_flag Pedge_flag_triplet[3];
```

`Pedge_flag_triplet_list` **EDGE FLAG TRIPLET LIST**

```
typedef struct {
  Pint              num_edge_flag_triplets;/*number of edge flag triplets  */
  Pedge_flag_triplet*edge_flag_triplets; /* list of edge flag triplets     */
} Pedge_flag_triplet_list;
```

`Pfacet` **FACET**

```
typedef struct {

  Pint              num_data_per_facet;  /* number of data values per-facet*/
  Pcolrv            *facet_colrv;        /* facet colour                   */
  Pvec3             *facet_norm;         /* facet normal                   */
  Pfloat            *facet_data;         /* list of facet data             */

} Pfacet;
```

NOTE — A NULL value for the `facet_colrv`, `facet_norm`, or `facet_data` fields indicates the corresponding optional data is not supplied.

`Pfacet_array` **FACET ARRAY**

```
typedef struct {

  Pint_size         dims;                /* facet array dimensions         */
  Pint              num_data_per_facet;  /* number of data values per-facet*/
  Pcolrv_ptr        facet_colrvs;        /* array of facet colour values   */
  Pvec3             *facet_norms;        /* array of facet normals         */
  Pfloat            *facet_data;         /* array of facet data lists      */

} Pfacet_array;
```

NOTE — A NULL value for the `facet_colrvs`, `facet_norms`, or `facet_data` fields indicates the corresponding optional data is not supplied.

`Pfacet_cull_mode` **FACET CULLING MODE**

```
typedef enum {

  PFACET_CULL_NONE,
  PFACET_CULL_BACKFACING,
  PFACET_CULL_FRONTFACING

} Pfacet_cull_mode;
```

`Pfacet_disting_mode` **FACET DISTINGUISHING MODE**

```
typedef enum {
  PFACET_DISTING_OFF,
  PFACET_DISTING_ON
} Pfacet_disting_mode;
```

`Pfacet_list` **FACET LIST**

```
typedef struct {
  Pint              num_facets;          /* number of facets                */
  Pint              num_data_per_facet;  /* number of data values per-facet*/
  Pcolrv_ptr        facet_colrvs;        /* list of facet colour values     */
  Pvec3             *facet_norms;        /* list of facet normals           */
  Pfloat            *facet_data;         /* list of facet data lists        */
} Pfacet_list;
```

NOTE — A `NULL` value for the `facet_colrvs`, `facet_norms`, or `facet_data` fields indicates the corresponding optional data is not supplied.

`Pfloat_list` **FLOAT LIST**

```
typedef struct {
  Pint              num_floats;          /* number of floats in list        */
  Pfloat            *floats;             /* list of floats                  */
} Pfloat_list;
```

`Pfloat_set` **FLOAT SET**

```
typedef struct {
  Pint              num_lists;           /* number of float lists           */
  Pfloat_list       *floats;             /* list of float lists             */
} Pfloat_set;
```

`Pgcolr` **GENERAL COLOUR**

```
typedef struct {
  Pint              colr_type;          /* colour type                */
  Pcolrv            colr_value;         /* colour value               */
} Pgcolr;
```

`Phomo_cieluv` **HOMOGENEOUS CIEL*U*V***

```
typedef struct {
  Pfloat            cieluv_x;           /* x coefficient              */
  Pfloat            cieluv_y;           /* y coefficient              */
  Pfloat            cieluv_y_lum;       /* y luminance                */
  Pfloat            w;                  /* homogeneous coordinate     */
} Phomo_cieluv;
```

`Phomo_hls` **HOMOGENEOUS HUE LIGHTNESS SATURATION**

```
typedef struct {
  Pfloat            hue;                /* hue                        */
  Pfloat            lightness;          /* lightness                  */
  Pfloat            satur;              /* saturation                 */
  Pfloat            w;                  /* homogeneous coordinate     */
} Phomo_hls;
```

`Phomo_hsv` **HOMOGENEOUS HUE SATURATION VALUE**

```
typedef struct {
  Pfloat            hue;                /* hue                        */
  Pfloat            satur;              /* saturation                 */
  Pfloat            value;              /* value                      */
  Pfloat            w;                  /* homogeneous coordinate     */
} Phomo_hsv;
```

`Phomo_point3` **HOMOGENEOUS POINT 3**

```
typedef struct {

  Pfloat            wx;             /* wx coordinate                */
  Pfloat            wy;             /* wy coordinate                */
  Pfloat            w;              /* homogeneous coordinate       */

} Phomo_point3;
```

`Phomo_point4` **HOMOGENEOUS POINT 4**

```
typedef struct {

  Pfloat            wx;             /* wx coordinate                */
  Pfloat            wy;             /* wy coordinate                */
  Pfloat            wz;             /* wz coordinate                */
  Pfloat            w;              /* homogeneous coordinate       */

} Phomo_point4;
```

`Phomo_rgb` **HOMOGENEOUS RED GREEN BLUE**

```
typedef struct {

  Pfloat            red;            /* red                          */
  Pfloat            green;          /* green                        */
  Pfloat            blue;           /* blue                         */
  Pfloat            w;              /* homogeneous coordinate       */

} Phomo_rgb;
```

`Pint_bundle_plus` **INTERIOR BUNDLE PLUS**

```
typedef struct {

  Pint_style        style;          /* interior style               */
  Pint              style_ind;      /* interior style index         */
  Pgcolr            colr;           /* interior colour              */
  Pint              shad_method;    /* shading method               */

} Pint_bundle_plus;
```

`Pint_facs_plus` **INTERIOR FACILITIES PLUS**

```
typedef struct {

  Pint              num_int_styles;      /* number of interior styles      */
  Pint_style        int_styles[5];       /* list of available interior     */
                                         /* styles                         */
  Pint              num_hatch_styles;    /* number of available hatch      */
                                         /* styles                         */
  Pint_list         hatch_styles;        /* list of available hatch styles */
  Pint_list         shad_methods;        /* list of available shading      */
                                         /* methods                        */
  Pint              num_pred_int_inds;   /* number of predefined interior  */
                                         /* indices                        */

} Pint_facs_plus;
```

`Pline_bundle_plus` **POLYLINE BUNDLE PLUS**

```
typedef struct {

  Pint                type;              /* line type                      */
  Pfloat              width;             /* linewidth scale factor         */
  Pgcolr              colr;              /* polyline colour                */
  Pint                shad_method;       /* polyline shading method        */
  Pcurve_approx_crit  approx_crit;       /* curve approximation criteria   */
                                         /* data record                    */

} Pline_bundle_plus;
```

`Pline_facs_plus` **POLYLINE FACILITIES PLUS**

```
typedef struct {

  Pint              num_types;          /* number of available linetypes  */
  Pint_list         types;              /* list of line types             */
  Pint              num_widths;         /* number of available line widths*/
  Pfloat            nom_width;          /* nominal linewidth              */
  Pfloat            min_width;          /* minimum linewidth              */
  Pfloat            max_width;          /* maximum linewidth              */
  Pint_list         shad_methods;       /* list of available line shading */
                                        /* methods                        */
  Pint              num_inds;           /* number of predefined bundle    */
                                        /* indices                        */

} Pline_facs_plus;
```

`Pmarker_bundle_plus` **POLYMARKER BUNDLE PLUS**

```
typedef struct {

  Pint              type;               /* marker type                    */
  Pfloat            size;               /* marker size scale factor       */
  Pgcolr            colr;               /* polymarker colour              */

} Pmarker_bundle_plus;
```

`Pparam_surf_rep` **PARAMETRIC SURFACE REPRESENTATION**

```
typedef struct {

  Psurf_approx_crit     approx_crit;    /* surface approximation criteria */
                                        /* data                           */
  Pparam_surf_chars     chars_data;     /* parametric surface             */
                                        /* characteristics data           */

} Pparam_surf_rep;
```

`Ppat_rep_plus` **PATTERN REPRESENTATION PLUS**

```
typedef struct {
  Pint                colr_type;      /* colour type                  */
  Pcolrv_array        colrs;          /* array of colour values       */
} Ppat_rep_plus;
```

`Pplacement` **PLACEMENT**

```
typedef enum {
  PPLACE_UNIFORM_OVER_SURF,
  PPLACE_UNIFORM_BETWEEN_KNOTS
} Pplacement;
```

`Ppoint_ptr` **POINT POINTER**

```
typedef union {
  Ppoint              *point;         /* pointer to 2D non-homogeneous */
                                      /* points                        */
  Phomo_point3        *homo_point3;   /* pointer to 3D homogeneous     */
                                      /* points                        */
} Ppoint_ptr;
```

`Ppoint3_ptr` **POINT 3 POINTER**

```
typedef union {
  Ppoint3             *point3;        /* pointer to 3D non-homogeneous */
                                      /* points                        */
  Phomo_point4        *homo_point4;   /* pointer to 4D homogeneous     */
                                      /* points                        */
} Ppoint3_ptr;
```

`Prationality` **RATIONALITY**

```
typedef enum {
  PB_SPLINE_RATIONAL,
  PB_SPLINE_NON_RATIONAL
} Prationality;
```

`Prefl_rep` **REFLECTANCE REPRESENTATION**

```
typedef struct {
  Pint              refl_model;       /* reflectance model              */
  Prefl_prop        refl_prop;        /* reflectance property data      */
} Prefl_rep;
```

`Pset_of_fill_area_sets_data` **SET OF FILL AREA SETS DATA**

```
typedef struct {
  Pint              num_sets;         /* number of fill area sets       */
  Pint              num_data_per_facet;  /* number of data values per facet*/
  Pcolrv_ptr        facet_colrvs;     /* list of facet colours          */
  Pvec3             *facet_norms;     /* list of facet normals          */
  Pfloat            *facet_data;      /* list of facet data lists       */
} Pset_of_fill_area_sets_data;
```

NOTE — A NULL value for the `facet_colrvs`, `facet_norms`, or `facet_data` fields indicates the corresponding optional data is not supplied.

`Psource_select` **SOURCE SELECT**

```
typedef enum {
  PSOURCE_SELECT_COLR_ASPECT,
  PSOURCE_SELECT_VERT_COLR,
  PSOURCE_SELECT_VERT_DATA,
  PSOURCE_SELECT_FACET_COLR,
  PSOURCE_SELECT_FACET_DATA
} Psource_select;
```

`Psource_select_list` **SOURCE SELECT LIST**

```
typedef struct {
  Pint              num_selectors;      /* number of selectors          */
  Psource_select    *selectors;         /* list of source selectors     */
} Psource_select_list;
```

`Psurf_colr_spline` **SURFACE COLOUR SPLINE**

```
typedef struct {
  Pint                      u_order;     /* u spline order                */
  Pint                      v_order;     /* v spline order                */
  Pfloat_list               u_knots;     /* list of u knot values         */
  Pfloat_list               v_knots;     /* list of v knot values         */
  Pint                      colr_type;   /* colour type                   */
  Pcolr_ctrl_point_array    colrs;       /* array of colour control points */
} Psurf_colr_spline;
```

`Psurf_data_spline` **SURFACE DATA SPLINE**

```
typedef struct {

  Pint                        u_order;      /* u spline order                */
  Pint                        v_order;      /* v spline order                */
  Pfloat_list                 u_knots;      /* list of u knot values         */
  Pfloat_list                 v_knots;      /* list of v knot values         */
  Pdata_ctrl_point_array      data_points;  /* array of data control         */
                                            /* points                        */

} Psurf_data_spline;
```

`Psurf_data_spline_list` **SURFACE DATA SPLINE LIST**

```
typedef struct {

  Pint                num_data_spline;    /* number of data splines        */
  Psurf_data_spline   *data_splines;      /* list of data splines          */

} Psurf_data_spline_list;
```

`Psurf_geom_spline` **SURFACE GEOMETRY SPLINE**

```
typedef struct {

  Pint                  u_order;          /* u spline order                */
  Pint                  v_order;          /* v spline order                */
  Pfloat_list           u_knots;          /* u knots                       */
  Pfloat_list           v_knots;          /* v knots                       */
  Pctrl_point3_array    ctrl_points;      /* array of control points       */

} Psurf_geom_spline;
```

`Ptext_bundle_plus` **TEXT BUNDLE PLUS**

```
typedef struct {
  Pint              font;              /* text font                    */
  Ptext_prec        precision;         /* text precision               */
  Pfloat            char_expan;        /* character expansion factor   */
  Pfloat            char_space;        /* character spacing            */
  Pgcolr            colr;              /* text colour                  */
} Ptext_bundle_plus;
```

`Ptri_set_data` **TRIANGLE SET DATA**

```
typedef struct {
  Pint              num_sets;          /* number of triangle sets      */
  Pint              num_data_per_facet;/*number of data values per     */
                                       /* facet                        */
  Pcolrv_ptr        facet_colrvs;      /* list of facet colours        */
  Pvec3             *facet_norms;      /* list of facet normals        */
  Pfloat            *facet_data;       /* list of facet data lists     */
} Ptri_set_data;
```

NOTE — A NULL value for the `facet_colrvs`, `facet_norms`, `facet_data`, or `edge_flags` fields indicates the corresponding optional data is not supplied.

`Ptrim_curve` **TRIMMING CURVE**

```
typedef struct {
  Pcurve_approx_crit  approx_crit;     /* trimming curve approx.       */
                                       /* criteria data record         */
  Pvis_flag           visibility_flag; /* curve visibility flag        */
  Pint                order;           /* curve order                  */
  Pfloat_list         knots;           /* curve knot vector            */
  Pfloat              low_limit;       /* curve parameter range lower  */
                                       /* limit                        */
  Pfloat              high_limit;      /* curve parameter range upper  */
                                       /* limit                        */
  Pctrl_point_list    ctrl_points;     /* curve control points         */
} Ptrim_curve;
```

`Ptrim_curve_loop` **TRIMMING CURVE LOOP**

```
typedef struct {
  Pint              num_trim_curves;     /* number of trimming curves       */
  Ptrim_curve       *trim_curves;        /* list of trimming curves forming */
                                         /* a closed loop                   */
} Ptrim_curve_loop;
```

`Ptrim_curve_loop_list` **TRIMMING CURVE LOOP LIST**

```
typedef struct {
  Pint                 num_trim_loops;   /* number of trimming curve loops */
  Ptrim_curve_loop     *trim_loops;      /* list of trimming curve loops   */
} Ptrim_curve_loop_list;
```

`Pvertex_array` **VERTEX ARRAY**

```
typedef struct {
  Pint_size         dims;                /* dimensions of vertex array     */
  Pint              num_data_per_vertex; /* number of data values          */
                                         /* per-vertex                     */
  Ppoint            *vertex_points;      /* array of vertex points         */
  Pcolrv_ptr        vertex_colrvs;       /* array of vertex colours        */
  Pvec3             *vertex_norms;       /* array of vertex normals        */
  Pfloat            *vertex_data;        /* array of vertex data lists     */
} Pvertex_array;
```

NOTE — A `NULL` value for the `vertex_colrvs`, `vertex_norms`, or `vertex_data` fields indicates the corresponding optional data is not supplied.

`Pvertex_ind_triplet` **VERTEX INDEX TRIPLET**

```
typedef Pint Pvertex_ind_triplet[3];
```

`Pvertex_ind_triplet_list` **VERTEX INDEX TRIPLET LIST**

```
typedef struct {
  Pint            num_vertex_ind_triplets;/* number of vertex index      */
                                          /* triplets                    */
  Pvertex_ind_triplets*vertex_ind_triplets;/*list of vertex index triplets */
} Pvertex_ind_triplet_list;
```

`Pvertex_list` **VERTEX LIST**

```
typedef struct {
  Pint              num_vertices;         /* number of vertices          */
  Pint              num_data_per_vertex;  /* number of data values       */
                                          /* per-vertex                  */
  Ppoint            *vertex_points;       /* list of vertex points       */
  Pcolrv_ptr        vertex_colrvs;        /* list of vertex colours      */
  Pvec3             *vertex_norms;        /* list of vertex normals      */
  Pfloat            *vertex_data;         /* list of vertex data lists   */
} Pvertex_list;
```

NOTE — A NULL value for the `vertex_colrvs`, `vertex_norms`, or `vertex_data` fields indicates the corresponding optional data is not supplied.

`Pvertex_set` **VERTEX SET**

```
typedef struct {
  Pint              num_lists;            /* number of vertex lists      */
  Pvertex_list      *vertices;            /* list of vertex lists        */
} Pvertex_set;
```

`Pvertex3_array` **VERTEX 3 ARRAY**

```
typedef struct {
  Pint_size          dims;                /* dimensions of vertex array     */
  Pint               num_data_per_vertex; /* number of data values          */
                                          /* per-vertex                     */
  Ppoint3            *vertex_points;      /* array of vertex points         */
  Pcolrv_ptr         vertex_colrvs;       /* array of vertex colours        */
  Pvec3              *vertex_norms;       /* array of vertex normals        */
  Pfloat             *vertex_data;        /* array of vertex data lists     */
} Pvertex3_array;
```

NOTE — A `NULL` value for the `vertex_colrvs`, `vertex_norms`, or `vertex_data` fields indicates the corresponding optional data is not supplied.

`Pvertex3_list` **VERTEX 3 LIST**

```
typedef struct {
  Pint               num_vertices;        /* number of vertices             */
  Pint               num_data_per_vertex; /* number of data values          */
                                          /* per-vertex                     */
  Ppoint3            *vertex_points;      /* list of vertex points          */
  Pcolrv_ptr         vertex_colrvs;       /* list of vertex colours         */
  Pvec3              *vertex_norms;       /* list of vertex normals         */
  Pfloat             *vertex_data;        /* list of vertex data lists      */
} Pvertex3_list;
```

NOTE — A `NULL` value for the `vertex_colrvs`, `vertex_norms`, or `vertex_data` fields indicates the corresponding optional data is not supplied.

`Pvertex3_set` **VERTEX 3 SET**

```
typedef struct {
  Pint               num_lists;           /* number of vertex lists         */
  Pvertex3_list      *vertices;           /* list of vertex lists           */
} Pvertex3_set;
```

`Pvis_flag` **VISIBILITY FLAG**

```
typedef enum {
  PVIS_OFF,
  PVIS_ON
} Pvis_flag;
```

`Pws_st_tables_plus` **LENGTHS OF WORKSTATION STATE TABLES**

```
typedef struct {
  Pint              data_map_rep;       /* max. # of data mapping table   */
                                        /* entries                        */
  Pint              refl_rep;           /* max. # of reflectance table    */
                                        /* entries                        */
  Pint              param_surf_rep;     /* max. # of param. surface table */
                                        /* entries                        */
  Pint              light_source_rep;   /* max. # of light source table   */
                                        /* entries                        */
  Pint              depth_cue_rep;      /* max. # of depth cue table      */
                                        /* entries                        */
  Pint              colr_map_rep;       /* max. # of colour mapping table */
                                        /* entries                        */
} Pws_st_tables_plus;
```

9 C PHIGS PLUS macro definitions

9.1 Function identifiers

The error functions require a unique mapping of the PHIGS PLUS functions to a set of numbers. The names for these function identifiers are the same as the bound PHIGS PLUS function names except that the sentinel character has been replaced by `Pfn_`.

Function Name Macro	Function Number
`#define Pfn_polyline_set3_colr`	(183)
`#define Pfn_fill_area_set3_data`	(184)
`#define Pfn_fill_area_set_data`	(185)
`#define Pfn_cell_array3_plus`	(186)
`#define Pfn_set_of_fill_area_sets3_data`	(187)
`#define Pfn_set_of_fill_area_sets_data`	(188)
`#define Pfn_tri_set3_data`	(189)
`#define Pfn_tri_set_data`	(190)
`#define Pfn_tri_strip3_data`	(191)
`#define Pfn_tri_strip_data`	(192)
`#define Pfn_quad_mesh3_data`	(193)
`#define Pfn_quad_mesh_data`	(194)
`#define Pfn_non_uniform_b_spline_curve3`	(195)
`#define Pfn_non_uniform_b_spline_curve3_colr`	(196)
`#define Pfn_non_uniform_b_spline_surf3`	(197)
`#define Pfn_non_uniform_b_spline_surf3_data`	(198)
`#define Pfn_set_data_map_ind`	(199)
`#define Pfn_set_refl_ind`	(200)
`#define Pfn_set_back_int_ind`	(201)
`#define Pfn_set_back_data_map_ind`	(202)
`#define Pfn_set_back_refl_ind`	(203)
`#define Pfn_set_param_surf_ind`	(204)
`#define Pfn_set_line_colr`	(205)
`#define Pfn_set_line_shad_method`	(206)
`#define Pfn_set_marker_colr`	(207)
`#define Pfn_set_text_colr`	(208)
`#define Pfn_set_facet_disting_mode`	(209)
`#define Pfn_set_facet_cull_mode`	(210)
`#define Pfn_set_int_colr`	(211)
`#define Pfn_set_int_shad_method`	(212)
`#define Pfn_set_data_map_method`	(213)
`#define Pfn_set_refl_props`	(214)
`#define Pfn_set_refl_model`	(215)
`#define Pfn_set_back_int_style`	(216)
`#define Pfn_set_back_int_style_ind`	(217)
`#define Pfn_set_back_int_colr`	(218)
`#define Pfn_set_back_int_shad_method`	(219)

```
#define Pfn_set_back_data_map_method        (220)
#define Pfn_set_back_refl_props             (221)
#define Pfn_set_back_refl_model             (222)
#define Pfn_set_light_source_st             (223)
#define Pfn_set_edge_colr                   (224)
#define Pfn_set_curve_approx_crit           (225)
#define Pfn_set_surf_approx_crit            (226)
#define Pfn_set_param_surf_chars            (227)
#define Pfn_set_rend_colr_model             (228)
#define Pfn_set_depth_cue_ind               (229)
#define Pfn_set_colr_map_ind                (230)
#define Pfn_set_line_rep_plus               (231)
#define Pfn_set_marker_rep_plus             (232)
#define Pfn_set_text_rep_plus               (233)
#define Pfn_set_int_rep_plus                (234)
#define Pfn_set_edge_rep_plus               (235)
#define Pfn_set_data_map_rep                (236)
#define Pfn_set_refl_rep                    (237)
#define Pfn_set_param_surf_rep              (238)
#define Pfn_set_pat_rep_plus                (239)
#define Pfn_set_light_source_rep            (240)
#define Pfn_set_depth_cue_rep               (241)
#define Pfn_set_colr_map_rep                (242)
```

9.2 Error codes

```
/* Output Attribute Errors */
#define PE_BAD_DEPTH_CUE_MODE          (119) /* Ignoring function, the spec-
                                                ified depth cue mode is not
                                                available on the workstation
                                                */
#define PE_DEPTH_CUE_INDX_LT_1         (120) /* Ignoring function, the
                                                depth cue index is less
                                                than one */
#define PE_COLR_MAP_INDX_LT_1          (121) /* Ignoring function, the
                                                colour mapping index is
                                                less than one */
#define PE_BAD_LINE_SHAD_METHOD        (122) /* Ignoring function, the spec-
                                                ified polyline shading
                                                method is not available on
                                                the workstation */
#define PE_BAD_INT_SHAD_METHOD         (123) /* Ignoring function, the spec-
                                                ified interior shading
                                                method is not available on
                                                the workstation */
#define PE_BAD_REFL_MODEL              (124) /* Ignoring function, the spec-
                                                ified reflectance model is
```

```
                                                    not available on the work-
                                                    station */
#define PE_TOTAL_COLR_RANGE_FIELDS_TOO_LARGE   (125)
                                                  /* Ignoring function, the
                                                    total of the colour range
                                                    fields in all the table
                                                    entries is too large */
#define PE_BAD_COLR_MAP_METHOD          (126) /* Ignoring function, the spec-
                                                    ified colour mapping method
                                                    is not available on the
                                                    specified workstation */
#define PE_BAD_APPROX_CRIT_TYPE         (127) /* Ignoring function, the spec-
                                                    ified approximation crite-
                                                    ria type is not available
                                                    on the specified workstation
                                                    */
#define PE_BAD_PARAM_SURF_CHAR_TYPE     (128) /* Ignoring function, the spec-
                                                    ified parametric surface
                                                    characteristics type is not
                                                    available on the specified
                                                    workstation */
#define PE_LIGHT_SOURCE_INDX_LT_1       (129) /* Ignoring function, the
                                                    light source index is less
                                                    than one */
#define PE_INVALID_REF_PLANES           (130) /* Ignoring function, invalid
                                                    reference planes; DQMIN >
                                                    DQMAX */
#define PE_BAD_LIGHT_SOURCE_TYPE        (131) /* Ignoring function, the spec-
                                                    ified light source type is
                                                    not available on the work-
                                                    station */
#define PE_SPOTLIGHT_SPREAD_ANGLE_RANGE (132) /* Ignoring function, the spec-
                                                    ified spot light spread
                                                    angle is out of range */
#define PE_BAD_ACTIVATION_LIST          (133) /* Ignoring function, one of
                                                    the entries in the activa-
                                                    tion list or deactivation
                                                    list is less than one */
#define PE_GEN_COLR_NOT_INDIRECT        (134) /* Ignoring function, the
                                                    requested entry contains a
                                                    general colour specifica-
                                                    tion with colour type other
                                                    than INDIRECT */
#define PE_SAME_ACTIVATION_DEACTIVATION_ENTRY  (135)
                                                  /* Ignoring function, the same
                                                    entry exists in both the
                                                    activation and the deactiva-
                                                    tion list */
#define PE_COLR_COMPONENT_RANGE         (136) /* Ignoring function, one of
                                                    the components of the
```

```
                                                colour specification is out
                                                of range */
#define PE_BAD_DATA_MAP_METHOD          (137) /* Ignoring function, the spec-
                                                ified data mapping method
                                                is not available on the
                                                specified workstation */
#define PE_BAD_DATA_RECORD_FIELD        (138) /* Ignoring function, one or
                                                more of the fields within
                                                the specified data record
                                                is in error */
#define PE_BAD_REFL_PROP_TYPE           (139) /* Ignoring function, the spec-
                                                ified reflectance property
                                                type is not available on
                                                the specified workstation */
#define PE_INVALID_DEPTH_CUE_SCALE      (140) /* Ignoring function, one of
                                                the depth cue scale factors
                                                is not in the required range
                                                */

/* Output Primitive Errors */
#define PE_ORDER_LT_1                   (500) /* Ignoring function, the
                                                order of a spline is less
                                                than one */
#define PE_INSUFF_CTRL_POINTS_FOR_ORDER (501) /* Ignoring function, a spline
                                                does not contain enough con-
                                                trol points for its speci-
                                                fied order */
#define PE_ORDER_INCONSISTENT_KNOTS_CTRL_POINTS   (502)
                                              /* Ignoring function, the
                                                order of a spline is incon-
                                                sistent with number of
                                                knots and control points */
#define PE_KNOT_SEQUENCE_NOT_NONDECREASING     (503)
                                              /* Ignoring function, the knot
                                                sequence for a spline is
                                                not non-decreasing */
#define PE_VERTEX_INDX_OUT_OF_RANGE     (504) /* Ignoring function, one or
                                                more of the vertex indices
                                                is out of range */
#define PE_PARAM_RANGE_INCONSISTENT_KNOTS      (506)
                                              /* Ignoring function, the
                                                parameter range for a
                                                spline is inconsistent with
                                                its knots */
#define PE_W_VALUE_LE_0                 (507) /* Ignoring function, the
                                                fourth coordinate of a
                                                rational control point is
                                                less than or equal to zero */
#define PE_TRIM_CURVE_ORDER_LT_2        (508) /* Ignoring function, a trim-
                                                ming curve's order is less
                                                than two */
```

```
#define PE_INCONSISTENT_EDGE_FLAG_SPEC   (513) /* Ignoring function, the num-
                                                  ber of edge visibility
                                                  flags is inconsistent with
                                                  the number of edges in the
                                                  primitive */
#define PE_DATA_LIST_LENGTH_NOT_SAME     (514) /* Ignoring function, the data
                                                  lists do not all contain
                                                  the same number of entries */
#define PE_INCONSISTENT_FACET_SPEC       (516) /* Ignoring function, the
                                                  facet data is inconsistent
                                                  with the number of facets
                                                  in the primitive */
#define PE_PARAM_RANGE_COLR_SPLINE       (517) /* Ignoring function, the
                                                  parameter range of the
                                                  colour spline does not
                                                  include the effective param-
                                                  eter range of the geometry
                                                  spline */
#define PE_PARAM_RANGE_DATA_SPLINE       (518) /* Ignoring function, the
                                                  parameter range of the data
                                                  spline does not include the
                                                  effective parameter range
                                                  of the geometry spline */
#define PE_BAD_VERTEX_DATA_TYPE          (2207)/* Ignoring function, two or
                                                  more vertices do not con-
                                                  tain the same type of data*/
```

9.3 Miscellaneous

9.3.1 Colour mapping methods

```
#define PCOLR_MAP_METHOD_TRUE            (1)
#define PCOLR_MAP_METHOD_PSEUDO          (2)
#define PCOLR_MAP_METHOD_PSEUDO_N        (3)
```

9.3.2 Colour types

```
#define PCOLR_INDIRECT                   (0)
#define PCOLR_RGB                        (1)
#define PCOLR_CIELUV                     (2)
#define PCOLR_HSV                        (3)
#define PCOLR_HLS                        (4)
```

9.3.3 Curve approximation criteria types

```
#define PCURVE_APPROX_WS_DEP                (1)
#define PCURVE_APPROX_CONST_PARAM_SUBD      (2)
#define PCURVE_APPROX_CHORDAL_SIZE_WC       (3)
#define PCURVE_APPROX_CHORDAL_SIZE_NPC      (4)
#define PCURVE_APPROX_CHORDAL_SIZE_DC       (5)
#define PCURVE_APPROX_CHORDAL_DEV_WC        (6)
#define PCURVE_APPROX_CHORDAL_DEV_NPC       (7)
#define PCURVE_APPROX_CHORDAL_DEV_DC        (8)
#define PCURVE_APPROX_REL_WC                (9)
#define PCURVE_APPROX_REL_NPC               (10)
#define PCURVE_APPROX_REL_DC                (11)
```

9.3.4 Data mapping methods

```
#define PDATA_MAP_METHOD_COLR                   (1)
#define PDATA_MAP_METHOD_SINGLE_UNIFORM         (2)
#define PDATA_MAP_METHOD_SINGLE_NON_UNIFORM     (3)
#define PDATA_MAP_METHOD_BI_UNIFORM             (4)
#define PDATA_MAP_METHOD_BI_NON_UNIFORM         (5)
```

9.3.5 Interior shading method

```
#define PINT_SHAD_METHOD_NONE       (1)
#define PINT_SHAD_METHOD_COLR       (2)
#define PINT_SHAD_METHOD_DATA       (3)
#define PINT_SHAD_METHOD_DATA_DOT   (4)
#define PINT_SHAD_METHOD_DATA_NORM  (5)
```

9.3.6 Light source types

```
#define PLIGHT_SOURCE_AMB       (1)
#define PLIGHT_SOURCE_DIR       (2)
#define PLIGHT_SOURCE_POS       (3)
#define PLIGHT_SOURCE_SPOT      (4)
```

9.3.7 Parametric surface characteristic types

```
#define PPARAM_SURF_CHARS_NONE          (1)
#define PPARAM_SURF_CHARS_WS_DEP        (2)
#define PPARAM_SURF_CHARS_ISOPARAM      (3)
#define PPARAM_SURF_CHARS_LEVEL_MC      (4)
#define PPARAM_SURF_CHARS_LEVEL_WC      (5)
```

9.3.8 Polyline shading method

```
#define PLINE_SHAD_METHOD_NONE                (1)
#define PLINE_SHAD_METHOD_COLR                (2)
```

9.3.9 Reflectance models

```
#define PREFL_MODEL_NO_REFL                   (1)
#define PREFL_MODEL_AMB_REFL                  (2)
#define PREFL_MODEL_AMB_DIFF_REFL             (3)
#define PREFL_MODEL_AMB_DIFF_SPEC_REFL        (4)
```

9.3.10 Reflectance properties

```
#define PREFL_PROPS_SIMPLE_REFL               (1)
```

9.3.11 Rendering Colour Model

```
#define PREND_COLR_MODEL_WS_DEP               (0)
#define PREND_COLR_MODEL_RGB                  (1)
#define PREND_COLR_MODEL_CIELUV               (2)
#define PREND_COLR_MODEL_HSV                  (3)
#define PREND_COLR_MODEL_HLS                  (4)
```

9.3.12 Surface approximation criteria types

```
#define PSURF_APPROX_WS_DEP                   (1)
#define PSURF_APPROX_CONST_PARAM_SUBD         (2)
#define PSURF_APPROX_CHORDAL_SIZE_WC          (3)
#define PSURF_APPROX_CHORDAL_SIZE_NPC         (4)
#define PSURF_APPROX_CHORDAL_SIZE_DC          (5)
#define PSURF_APPROX_PLANAR_DEV_WC            (6)
#define PSURF_APPROX_PLANAR_DEV_NPC           (7)
#define PSURF_APPROX_PLANAR_DEV_DC            (8)
#define PSURF_APPROX_REL_WC                   (9)
#define PSURF_APPROX_REL_NPC                  (10)
#define PSURF_APPROX_REL_DC                   (11)
```

9.3.13 Element enumeration

```
#define PFIRST_PHIGS_PLUS_ELEM                (PELEM_POLYLINE_SET3_COLR)
#define PLAST_PHIGS_PLUS_ELEM                 (PELEM_COLR_MAP_IND)
```

10 C PHIGS PLUS functions

10.1 Output primitive functions

POLYLINE SET 3 WITH COLOUR

```
void ppolyline_set3_colr (

  Pint                    colr_type,      /* colour type                  */
  const Pvertex3_set      *vertex_data    /* list of vertex lists         */

);
```

NOTE — Only the vertex points and colour portions of the data pointed to by the `vertex_data` parameter are used. All other data is ignored.

FILL AREA SET 3 WITH DATA

```
void pfill_area_set3_data (

  Pint                    colr_type,      /* colour type                  */
  const Pfacet            *facet_data,    /* facet data                   */
  const Pedge_flag_set    *edge_flags,    /* list of edge flag lists      */
  const Pvertex3_set      *vertex_data    /* list of vertex lists         */

);
```

NOTE — A `NULL` value for the `facet_data` or `edge_flags` parameters indicates the corresponding optional data is not supplied.

FILL AREA SET WITH DATA

```
void pfill_area_set_data (

  Pint                    colr_type,      /* colour type                  */
  const Pfacet            *facet_data,    /* facet data                   */
  const Pedge_flag_set    *edge_flags,    /* list of edge flag lists      */
  const Pvertex_set       *vertex_data    /* list of vertex lists         */

);
```

NOTE — A NULL value for the facet_data or edge_flags parameters indicates the corresponding optional data is not supplied.

CELL ARRAY 3 PLUS

```
void pcell_array3_plus (

  const Pparal            *paral,         /* parallelogram P, Q, R        */
  const Ppat_rep_plus     *pat            /* pattern colour array         */

);
```

SET OF FILL AREA SETS 3 WITH DATA

```
void pset_of_fill_area_sets3_data (

  Pint                        colr_type,    /* colour type                */
  const Pset_of_fill_area_sets_data  *set_data,/*fill area set data        */
  const Pedge_flag_set_list   *edge_flags,  /* list of edge flag sets       */
  const Pvertex3_list         *vertex_data, /* list of vertices             */
  const Pcontour_set_list     *contours     /* list of contour sets         */

);
```

NOTE — A NULL value for the set_data or edge_flags parameters indicates the corresponding optional data is not supplied.

SET OF FILL AREA SETS WITH DATA

```
void pset_of_fill_area_sets_data (

  Pint                          colr_type,    /* colour type                */
  const Pset_of_fill_area_sets_data  *set_data,/*fill area set data          */
  const Pedge_flag_set_list     *edge_flags,  /* list of edge flag sets     */
  const Pvertex_list            *vertex_data, /* list of vertices           */
  const Pcontour_set_list       *contours     /* list of contour sets       */

);
```

NOTE — A NULL value for the set_data or edge_flags parameters indicates the corresponding optional data is not supplied.

TRIANGLE SET 3 WITH DATA

```
void ptri_set3_data (

  Pint                        colr_type,     /* colour type                 */
  const Ptri_set_data         *set_data,     /* triangle set data           */
  const Pedge_flag_triplet_list *edge_flags,/*list of triangle edge flag     */
                                             /* triplets                    */
  const Pvertex3_list         *vertex_data,  /* list of vertices            */
  const Pvertex_ind_triplet_list *triplets /* list of triangle vertex index */
                                             /* triplets                    */

);
```

NOTE — A NULL value for the set_data or edge_flags parameters indicates the corresponding optional data is not supplied.

TRIANGLE SET WITH DATA

```
void ptri_set_data (

  Pint                        colr_type,      /* colour type                    */
  const Ptri_set_data         *set_data,      /* triangle set data              */
  const Pedge_flag_triplet_list *edge_flags,/*list of triangle edge flag        */
                                              /* triplets                       */
  const Pvertex_list          *vertex_data,   /* list of vertices               */
  const Pvertex_ind_triplet_list *vertex_inds/*list of triangle vertex index */
                                              /* triplets                       */

);
```

NOTE — A NULL value for the `set_data` or `edge_flags` parameters indicates the corresponding optional data is not supplied.

TRIANGLE STRIP 3 WITH DATA

```
void ptri_strip3_data (

  Pint                      colr_type,      /* colour type                    */
  const Pfacet_list         *facet_data,    /* list of facet data             */
  const Pedge_flag_list     *edge_flags,    /* list of edge flags             */
  const Pvertex3_list       *vertex_data    /* list of vertex data            */

);
```

NOTE — A NULL value for the `facet_data` or `edge_flags` parameters indicates the corresponding optional data is not supplied.

TRIANGLE STRIP WITH DATA

```
void ptri_strip_data (

  Pint                      colr_type,      /* colour type                    */
  const Pfacet_list         *facet_data,    /* list of facet data             */
  const Pedge_flag_list     *edge_flags,    /* list of edge flags             */
  const Pvertex_list        *vertex_data    /* list of vertex data            */

);
```

NOTE — A NULL value for the `facet_data` or `edge_flags` parameters indicates the corresponding optional data is not supplied.

QUADRILATERAL MESH 3 WITH DATA

```
void pquad_mesh3_data (

  Pint                    colr_type,      /* colour type                  */
  const Pfacet_array      *facet_data,    /* array of facet data          */
  const Pedge_flag_array  *edge_flags,    /* array of edge flags          */
  const Pvertex3_array    *vertex_data    /* array of vertex data         */

);
```

NOTE — A NULL value for the facet_data or edge_flags parameters indicates the corresponding optional data is not supplied.

QUADRILATERAL MESH WITH DATA

```
void pquad_mesh_data (

  Pint                    colr_type,      /* colour type                  */
  const Pfacet_array      *facet_data,    /* array of facet data          */
  const Pedge_flag_array  *edge_flags,    /* array of edge flags          */
  const Pvertex_array     *vertex_data    /* array of vertex data         */

);
```

NOTE — A NULL value for the facet_data or edge_flags parameters indicates the corresponding optional data is not supplied.

NON-UNIFORM B-SPLINE CURVE 3

```
void pnon_uniform_b_spline_curve3 (

  const Pcurve_geom_spline  *geom_spline    /* curve geometry spline        */

);
```

NON-UNIFORM B-SPLINE CURVE 3 WITH COLOUR

```
void pnon_uniform_b_spline_curve3_colr (

  const Pcurve_geom_spline  *geom_spline    /* curve geometry spline          */
  const Pcurve_colr_spline  *colr_spline    /* curve colour spline            */

);
```

NOTE — A NULL value for the `colr_spline` parameter indicates the corresponding optional data is not supplied.

NON-UNIFORM B-SPLINE SURFACE 3

```
void pnon_uniform_b_spline_surf3 (

  const Psurf_geom_spline      *geom_spline,/*surface geometry spline       */
  const Ptrim_curve_loop_list  *trim_loops /* list of trimming curve loops  */

);
```

NOTE — A NULL value for the `trim_loops` parameter indicates the corresponding optional data is not supplied.

NON-UNIFORM B-SPLINE SURFACE 3 WITH DATA

```
void pnon_uniform_b_spline_surf3_data (

  const Psurf_geom_spline       *geom_spline,/*surface geometry spline      */
  const Ptrim_curve_loop_list   *trim_loops,/*list of trimming curve loops  */
  const Psurf_colr_spline       *colr_spline,/*colour spline               */
  const Psurf_data_spline_list  *data_splines/*list of data splines         */

);
```

NOTE — A NULL value for the `trim_loops`, `colr_spline`, and `data_splines` parameters indicates the corresponding optional data is not supplied.

10.2 Attribute specification functions

10.2.1 Bundled attribute selection

SET DATA MAPPING INDEX

```
void pset_data_map_ind (
  Pint                  index                 /* data mapping index            */
);
```

SET REFLECTANCE INDEX

```
void pset_refl_ind (
  Pint                  index                 /* reflectance index             */
);
```

SET BACK INTERIOR INDEX

```
void pset_back_int_ind (
  Pint                  index                 /* back interior index           */
);
```

SET BACK DATA MAPPING INDEX

```
void pset_back_data_map_ind (
  Pint                  index                 /* back data mapping index       */
);
```

SET BACK REFLECTANCE INDEX

```
void pset_back_refl_ind (
  Pint                  index                 /* back reflectance index          */
);
```

SET PARAMETRIC SURFACE INDEX

```
void pset_param_surf_ind (
  Pint                  index                 /* parametric surface index        */
);
```

10.2.2 Individual attribute selection

SET POLYLINE COLOUR

```
void pset_line_colr (
  const Pgcolr          *colr                 /* polyline colour                 */
);
```

SET POLYLINE SHADING METHOD

```
void pset_line_shad_method (
  Pint                  method                /* polyline shading method         */
);
```

SET POLYMARKER COLOUR

```
void pset_marker_colr (
  const Pgcolr        *colr                 /* polymarker colour          */
);
```

SET TEXT COLOUR

```
void pset_text_colr (
  const Pgcolr        *colr                 /* text colour                */
);
```

SET FACET DISTINGUISHING MODE

```
void pset_facet_disting_mode (
  Pfacet_disting_mode     mode              /* distinguishing mode        */
);
```

SET FACET CULLING MODE

```
void pset_facet_cull_mode (
  Pfacet_cull_mode    mode                  /* facet culling mode         */
);
```

SET INTERIOR COLOUR

```
void pset_int_colr (
  const Pgcolr        *colr                 /* interior colour            */
);
```

SET INTERIOR SHADING METHOD

```
void pset_int_shad_method (
  Pint                  method                  /* interior shading method          */
);
```

SET DATA MAPPING METHOD

```
void pset_data_map_method (
  const Pdata_map_rep      *method              /* data mapping method              */
);
```

SET REFLECTANCE PROPERTIES

```
void pset_refl_props (
  const Prefl_prop     *refl_prop              /* reflectance property data        */
);
```

SET REFLECTANCE MODEL

```
void pset_refl_model (
  Pint                  model                   /* reflectance model                */
);
```

SET BACK INTERIOR STYLE

```
void pset_back_int_style (
  Pint_style            style                   /* back interior style              */
);
```

SET BACK INTERIOR STYLE INDEX

```
void pset_back_int_style_ind (
  Pint                  index                   /* back interior style index     */
);
```

SET BACK INTERIOR COLOUR

```
void pset_back_int_colr (
  const Pgcolr          *colr                   /* back interior colour          */
);
```

SET BACK INTERIOR SHADING METHOD

```
void pset_back_int_shad_method (
  Pint                  method                  /* interior shading method       */
);
```

SET BACK DATA MAPPING METHOD

```
void pset_back_data_map_method (
  const Pdata_map_rep     *method               /* data mapping method           */
);
```

SET BACK REFLECTANCE PROPERTIES

```
void pset_back_refl_props (
  const Prefl_prop        *refl_data            /* reflectance property data     */
);
```

SET BACK REFLECTANCE MODEL

```
void pset_back_refl_model (

  Pint                  model                   /* reflectance model              */

);
```

SET LIGHT SOURCE STATE

```
void pset_light_source_st (

  const Pint_list     *act_list,                /* activation list                */
  const Pint_list     *deact_list               /* deactivation list              */

);
```

SET EDGE COLOUR

```
void pset_edge_colr (

  const Pgcolr        *colr                     /* edge colour                    */

);
```

SET CURVE APPROXIMATION CRITERIA

```
void pset_curve_approx_crit (

  const Pcurve_approx_crit   *data              /* curve approximation criteria   */
                                                /* data record                    */

);
```

SET SURFACE APPROXIMATION CRITERIA

```
void pset_surf_approx_crit (
  const Psurf_approx_crit    *data          /* surface approximation criteria */
                                            /* data record                    */
);
```

SET PARAMETRIC SURFACE CHARACTERISTICS

```
void pset_param_surf_chars (
  const Pparam_surf_chars    *data          /* parametric surface             */
                                            /* characteristics data record    */
);
```

SET RENDERING COLOUR MODEL

```
void pset_rend_colr_model (
  Pint                  model               /* rendering colour model         */
);
```

SET DEPTH CUE INDEX

```
void pset_depth_cue_ind (
  Pint                  index               /* depth cue index                */
);
```

SET COLOUR MAPPING INDEX

```
void pset_colr_map_ind (

  Pint                  index                 /* colour mapping index          */

);
```

10.2.3 Workstation attribute table definition

SET POLYLINE REPRESENTATION PLUS

```
void pset_line_rep_plus (

  Pint                          ws_id,        /* workstation identifier        */
  Pint                          index,        /* polyline bundle index         */
  const Pline_bundle_plus       *rep          /* polyline representation       */

);
```

SET POLYMARKER REPRESENTATION PLUS

```
void pset_marker_rep_plus (

  Pint                          ws_id,        /* workstation identifier        */
  Pint                          index,        /* polymarker bundle index       */
  const Pmarker_bundle_plus     *rep          /* polymarker representation     */

);
```

SET TEXT REPRESENTATION PLUS

```
void pset_text_rep_plus (

  Pint                          ws_id,        /* workstation identifier        */
  Pint                          index,        /* text bundle index             */
  const Ptext_bundle_plus       *rep          /* text representation           */

);
```

SET INTERIOR REPRESENTATION PLUS

```
void pset_int_rep_plus (

  Pint                          ws_id,        /* workstation identifier          */
  Pint                          index,        /* interior bundle index           */
  const Pint_bundle_plus        *rep          /* interior representation         */

);
```

SET EDGE REPRESENTATION PLUS

```
void pset_edge_rep_plus (

  Pint                          ws_id,        /* workstation identifier          */
  Pint                          index,        /* edge bundle index               */
  const Pedge_bundle_plus       *rep          /* edge representation             */

);
```

SET DATA MAPPING REPRESENTATION

```
void pset_data_map_rep (

  Pint                          ws_id,        /* workstation identifier          */
  Pint                          index,        /* data mapping index              */
  const Pdata_map_rep           *rep          /* data mapping representation     */

);
```

SET REFLECTANCE REPRESENTATION

```
void pset_refl_rep (

  Pint                          ws_id,        /* workstation identifier          */
  Pint                          index,        /* reflectance index               */
  const Prefl_rep               *rep          /* reflectance representation      */

);
```

SET PARAMETRIC SURFACE REPRESENTATION

```
void pset_param_surf_rep (

  Pint                        ws_id,        /* workstation identifier          */
  Pint                        index,        /* parametric surface rep. index   */
  const Pparam_surf_rep       *rep          /* parametric surface              */
                                            /* representation                  */

);
```

SET PATTERN REPRESENTATION PLUS

```
void pset_pat_rep_plus (

  Pint                        ws_id,        /* workstation identifier          */
  Pint                        index,        /* pattern index                   */
  const Ppat_rep_plus         *rep          /* pattern representation plus     */

);
```

SET LIGHT SOURCE REPRESENTATION

```
void pset_light_source_rep (

  Pint                        ws_id,        /* workstation identifier          */
  Pint                        index,        /* light source index              */
  const Plight_source_rep     *rep          /* light source representation     */

);
```

SET DEPTH CUE REPRESENTATION

```
void pset_depth_cue_rep (

  Pint                        ws_id,        /* workstation identifier          */
  Pint                        index,        /* depth cue index                 */
  const Pdepth_cue_rep        *rep          /* depth cue representation        */

);
```

SET COLOUR MAPPING REPRESENTATION

```
void pset_colr_map_rep (

  Pint                        ws_id,        /* workstation identifier          */
  Pint                        index,        /* colour mapping index            */
  const Pcolr_map_rep         *rep          /* colour mapping representation   */

);
```

10.3 Inquiry functions

10.3.1 Inquiry functions for the workstation state list

INQUIRE POLYLINE REPRESENTATION PLUS

```
void pinq_line_rep_plus (

  Pint                ws_id,              /* workstation identifier          */
  Pint                index,              /* polyline index                  */
  Pinq_type           type,               /* type of returned values         */
  Pint                *err_ind,           /* OUT error indicator             */
  Pline_bundle_plus   *rep                /* OUT polyline representation     */

);
```

INQUIRE POLYMARKER REPRESENTATION PLUS

```
void pinq_marker_rep_plus (

  Pint                      ws_id,        /* workstation identifier          */
  Pint                      index,        /* polymarker index                */
  Pinq_type                 type,         /* type of returned values         */
  Pint                      *err_ind,     /* OUT error indicator             */
  Pmarker_bundle_plus       *rep          /* OUT polymarker representation   */

);
```

INQUIRE TEXT REPRESENTATION PLUS

```
void pinq_text_rep_plus (

  Pint                ws_id,              /* workstation identifier          */
  Pint                index,              /* text index                      */
  Pinq_type           type,               /* type of returned values         */
  Pint                *err_ind,           /* OUT error indicator             */
  Ptext_bundle_plus   *rep                /* OUT text representation         */

);
```

INQUIRE INTERIOR REPRESENTATION PLUS

```
void pinq_int_rep_plus (

  Pint                ws_id,              /* workstation identifier          */
  Pint                index,              /* interior index                  */
  Pinq_type           type,               /* type of returned values         */
  Pint                *err_ind,           /* OUT error indicator             */
  Pint_bundle_plus    *rep                /* OUT interior representation     */

);
```

INQUIRE EDGE REPRESENTATION PLUS

```
void pinq_edge_rep_plus (

  Pint                ws_id,              /* workstation identifier          */
  Pint                index,              /* edge index                      */
  Pinq_type           type,               /* type of returned values         */
  Pint                *err_ind,           /* OUT error indicator             */
  Pedge_bundle_plus   *rep                /* OUT edge representation         */

);
```

INQUIRE LIST OF DATA MAPPING INDICES

```
void pinq_list_data_map_inds (

  Pint                ws_id,                /* workstation identifier          */
  Pint                num_elems_appl_list,  /* # elems in appl list            */
  Pint                start_ind,            /* starting index                  */
  Pint                *err_ind,             /* OUT error indicator             */
  Pint_list           *indices,             /* OUT list of defined data        */
                                            /* mapping indices                 */
  Pint                *num_elems_impl_list  /* OUT # elems in impl list        */

);
```

INQUIRE DATA MAPPING REPRESENTATION

```
void pinq_data_map_rep (

  Pint                ws_id,                /* workstation identifier          */
  Pint                index,                /* data mapping index              */
  Pinq_type           type,                 /* type of returned values         */
  Pstore              store,                /* handle to Store object          */
  Pint                *err_ind,             /* OUT error indicator             */
  Pdata_map_rep       **rep                 /* OUT data mapping representation*/

);
```

NOTE — The memory referenced by `*rep` is managed by `store`.

INQUIRE LIST OF REFLECTANCE INDICES

```
void pinq_list_refl_inds (

  Pint                ws_id,                /* workstation identifier          */
  Pint                num_elems_appl_list,  /* # elems in appl list            */
  Pint                start_ind,            /* starting index                  */
  Pint                *err_ind,             /* OUT error indicator             */
  Pint_list           *indices,             /* OUT list of defined reflectance*/
                                            /* indices                         */
  Pint                *num_elems_impl_list  /* OUT # elems in impl list        */

);
```

INQUIRE REFLECTANCE REPRESENTATION

```
void pinq_refl_rep (

  Pint              ws_id,              /* workstation identifier          */
  Pint              index,              /* reflectance index               */
  Pinq_type         type,               /* type of returned values         */
  Pstore            store,              /* handle to Store object          */
  Pint              *err_ind,           /* OUT error indicator             */
  Prefl_rep         **rep               /* OUT reflectance representation */

);
```

NOTE — The memory referenced by `*rep` is managed by `store`.

INQUIRE LIST OF PARAMETRIC SURFACE INDICES

```
void pinq_list_param_surf_inds (

  Pint              ws_id,              /* workstation identifier          */
  Pint              num_elems_appl_list,/* # elems in appl list            */
  Pint              start_ind,          /* starting index                  */
  Pint              *err_ind,           /* OUT error indicator             */
  Pint_list         *indices,           /* OUT list of defined parametric */
                                        /* surface indices                 */
  Pint              *num_elems_impl_list /* OUT # elems in impl list       */

);
```

INQUIRE PARAMETRIC SURFACE REPRESENTATION

```
void pinq_param_surf_rep (

  Pint              ws_id,              /* workstation identifier          */
  Pint              index,              /* parametric surface bundle index*/
  Pinq_type         type,               /* type of returned values         */
  Pstore            store,              /* handle to Store object          */
  Pint              *err_ind,           /* OUT error indicator             */
  Pparam_surf_rep   **rep               /* OUT parametric surface          */
                                        /* representation                  */

);
```

NOTE — The memory referenced by `*rep` is managed by `store`.

INQUIRE PATTERN REPRESENTATION PLUS

```
void pinq_pat_rep_plus (

  Pint                ws_id,              /* workstation identifier          */
  Pint                index,              /* pattern index                   */
  Pinq_type           type,               /* type of returned values         */
  Pstore              store,              /* handle to Store object          */
  Pint                *err_ind,           /* OUT error indicator             */
  Ppat_rep_plus       **rep               /* OUT pattern representation plus*/

);
```

NOTE — The memory referenced by `*rep` is managed by `store`.

INQUIRE LIST OF LIGHT SOURCE INDICES

```
void pinq_list_light_source_inds (

  Pint                ws_id,              /* workstation identifier          */
  Pint                num_elems_appl_list, /* # elems in appl list           */
  Pint                start_ind,          /* starting index                  */
  Pint                *err_ind,           /* OUT error indicator             */
  Pint_list           *indices,           /* OUT list of defined light       */
                                          /* source indices                  */
  Pint                *num_elems_impl_list /* OUT # elems in impl list       */

);
```

INQUIRE LIGHT SOURCE REPRESENTATION

```
void pinq_light_source_rep (

  Pint                ws_id,              /* workstation identifier          */
  Pint                index,              /* light source index              */
  Pinq_type           type,               /* type of returned values         */
  Pint                *err_ind,           /* OUT error indicator             */
  Plight_source_rep   *rep                /* OUT light source representation*/

);
```

INQUIRE LIST OF DEPTH CUE INDICES

```
void pinq_list_depth_cue_inds (

  Pint                ws_id,                /* workstation identifier          */
  Pint                num_elems_appl_list,  /* # elems in appl list            */
  Pint                start_ind,            /* starting index                  */
  Pint                *err_ind,             /* OUT error indicator             */
  Pint_list           *indices,             /* OUT list of defined depth cue   */
                                            /* indices                         */
  Pint                *num_elems_impl_list  /* OUT # elems in impl list        */

);
```

INQUIRE DEPTH CUE REPRESENTATION

```
void pinq_depth_cue_rep (

  Pint                ws_id,                /* workstation identifier          */
  Pint                index,                /* depth cue index                 */
  Pinq_type           type,                 /* type of returned values         */
  Pint                *err_ind,             /* OUT error indicator             */
  Pdepth_cue_rep      *rep                  /* OUT depth cue representation    */

);
```

INQUIRE COLOUR MAPPING STATE

```
void pinq_colr_map_st (

  Pint                ws_id,                /* workstation identifier          */
  Pint                method,               /* colour mapping method           */
  Pstore              store,                /* handle to Store object          */
  Pint                *err_ind,             /* OUT error indicator             */
  Pcolr_map_st        **data                /* OUT data record                 */

);
```

NOTE — The memory referenced by `*data` is managed by `store`.

INQUIRE LIST OF COLOUR MAPPING INDICES

```
void pinq_list_colr_map_inds (

  Pint                ws_id,                /* workstation identifier          */
  Pint                num_elems_appl_list,  /* # elems in appl list            */
  Pint                start_ind,            /* starting index                  */
  Pint                *err_ind,             /* OUT error indicator             */
  Pint_list           *indices,             /* OUT list of defined colour      */
                                            /* mapping indices                 */
  Pint                *num_elems_impl_list  /* OUT # elems in impl list        */

);
```

INQUIRE COLOUR MAPPING REPRESENTATION

```
void pinq_colr_map_rep (

  Pint                ws_id,                /* workstation identifier          */
  Pint                index,                /* colour mapping index            */
  Pinq_type           type,                 /* type of returned values         */
  Pstore              store,                /* handle to Store object          */
  Pint                *err_ind,             /* OUT error indicator             */
  Pcolr_map_rep       **rep                 /* OUT colour mapping              */
                                            /* representation                  */

);
```

NOTE — The memory referenced by `*rep` is managed by `store`.

10.3.2 Inquiry functions for the workstation description table

---

INQUIRE DIRECT COLOUR MODEL FACILITIES

```
void pinq_direct_colr_model_facs (

  Pint           ws_type,                 /* workstation type                */
  Pint           num_elems_appl_list,     /* # elems in appl list            */
  Pint           start_ind,               /* starting index                  */
  Pint           *err_ind,                /* OUT error indicator             */
  Pint_list      *models,                 /* OUT list of directly            */
                                          /* specifiable colour models       */
  Pint           *num_elems_impl_list     /* OUT # elems in impl list        */

);
```

INQUIRE RENDERING COLOUR MODEL FACILITIES

```
void pinq_rend_colr_model_facs (

  Pint           ws_type,                 /* workstation type                */
  Pint           num_elems_appl_list,     /* # elems in appl list            */
  Pint           start_ind,               /* starting index                  */
  Pint           *err_ind,                /* OUT error indicator             */
  Pint_list      *models,                 /* OUT list of available rendering*/
                                          /* colour models                   */
  Pint           *num_elems_impl_list     /* OUT # elems in impl list        */

);
```

INQUIRE DYNAMICS OF WORKSTATION ATTRIBUTES PLUS

```
void pinq_dyns_ws_attrs_plus (

  Pint                  ws_type,         /* workstation type                */
  Pint                  *err_ind,        /* OUT error indicator             */
  Pdyn_ws_attrs_plus    *attr            /* OUT dynamics of workstation     */
                                         /* attributes plus                 */

);
```

INQUIRE POLYLINE FACILITIES PLUS

```
void pinq_line_facs_plus (

  Pint                ws_type,          /* workstation type                */
  Pstore              store,            /* handle to Store object          */
  Pint                *err_ind,         /* OUT error indicator             */
  Pline_facs_plus     **facs            /* OUT list of polyline            */
                                        /* facilities plus                 */

);
```

NOTE — The memory referenced by `*facs` is managed by `store`.

INQUIRE PREDEFINED POLYLINE REPRESENTATION PLUS

```
void pinq_pred_line_rep_plus (

  Pint                    ws_type,      /* workstation type                */
  Pint                    index,        /* predefined polyline index       */
  Pint                    *err_ind,     /* OUT error indicator             */
  Pline_bundle_plus       *rep          /* OUT predefined polyline         */
                                        /* representation                  */

);
```

INQUIRE PREDEFINED POLYMARKER REPRESENTATION PLUS

```
void pinq_pred_marker_rep_plus (

  Pint                    ws_type,      /* workstation type                */
  Pint                    index,        /* predefined polymarker index     */
  Pint                    *err_ind,     /* OUT error indicator             */
  Pmarker_bundle_plus     *rep          /* OUT predefined polymarker       */
                                        /* representation                  */

);
```

INQUIRE PREDEFINED TEXT REPRESENTATION PLUS

```
void pinq_pred_text_rep_plus (

  Pint                ws_type,          /* workstation type                */
  Pint                index,            /* predefined text index           */
  Pint                *err_ind,         /* OUT error indicator             */
  Ptext_bundle_plus   *rep              /* OUT predefined text             */
                                        /* representation                  */

);
```

INQUIRE INTERIOR FACILITIES PLUS

```
void pinq_int_facs_plus (

  Pint                ws_type,          /* workstation type                */
  Pstore              store,            /* handle to Store object          */
  Pint                *err_ind,         /* OUT error indicator             */
  Pint_facs_plus      **facs            /* OUT list of interior            */
                                        /* facilities plus                 */

);
```

NOTE — The memory referenced by `*facs` is managed by `store`.

INQUIRE PREDEFINED INTERIOR REPRESENTATION PLUS

```
void pinq_pred_int_rep_plus (

  Pint                ws_type,          /* workstation type                */
  Pint                index,            /* predefined interior index       */
  Pint                *err_ind,         /* OUT error indicator             */
  Pint_bundle_plus    *rep              /* OUT predefined interior         */
                                        /* representation                  */

);
```

INQUIRE PREDEFINED EDGE REPRESENTATION PLUS

```
void pinq_pred_edge_rep_plus (

  Pint                ws_type,            /* workstation type                */
  Pint                index,              /* predefined edge index           */
  Pint                *err_ind,           /* OUT error indicator             */
  Pedge_bundle_plus   *rep                /* OUT predefined edge             */
                                          /* representation                  */

);
```

INQUIRE DATA MAPPING FACILITIES

```
void pinq_data_map_facs (

  Pint            ws_type,                /* workstation type                */
  Pint            num_elems_appl_list,    /* # elems in appl list            */
  Pint            start_ind,              /* starting index                  */
  Pint            *err_ind,               /* OUT error indicator             */
  Pint_list       *methods,               /* OUT list of available data      */
                                          /* mapping methods                 */
  Pint            *num_inds,              /* OUT number of predefined data   */
                                          /* mapping indices                 */
  Pint            *num_elems_impl_list    /* OUT # elems in impl list        */

);
```

INQUIRE PREDEFINED DATA MAPPING REPRESENTATION

```
void pinq_pred_data_map_rep (

  Pint                ws_type,            /* workstation type                */
  Pint                index,              /* predefined data mapping index   */
  Pstore              store,              /* handle to Store object          */
  Pint                *err_ind,           /* OUT error indicator             */
  Pdata_map_rep       **rep               /* OUT data mapping representation*/

);
```

NOTE — The memory referenced by `*rep` is managed by `store`.

INQUIRE REFLECTANCE FACILITIES

```
void pinq_refl_facs (

  Pint            ws_type,                /* workstation type                */
  Pstore          store,                  /* handle to Store object          */
  Pint            *err_ind,               /* OUT error indicator             */
  Pint_list       **models,               /* OUT list of available           */
                                          /* reflectance models              */
  Pint_list       **refl_props,           /* OUT list of available           */
                                          /* reflectance properties          */
  Pint            *refl_indices           /* OUT number of predefined        */
                                          /* reflectance indices             */

);
```

NOTE — The memory referenced by `*models` and `*refl_props` is managed by `store`.

INQUIRE PREDEFINED REFLECTANCE REPRESENTATION

```
void pinq_pred_refl_rep (

  Pint            ws_type,                /* workstation type                */
  Pint            index,                  /* predefined reflectance index    */
  Pstore          store,                  /* handle to Store object          */
  Pint            *err_ind,               /* OUT error indicator             */
  Prefl_rep       **rep                   /* OUT predefined reflectance      */
                                          /* representation                  */

);
```

NOTE — The memory referenced by `*rep` is managed by `store`.

INQUIRE CURVE AND SURFACE FACILITIES

The **INQUIRE CURVE AND SURFACE FACILITIES** is broken into four related functions: **INQUIRE CURVE FACILITIES**, **INQUIRE PARAMETRIC SURFACE FACILITIES**, **INQUIRE B-SPLINE SURFACE FACILITIES**, and **INQUIRE TRIMMING CURVE FACILITIES**.

INQUIRE CURVE FACILITIES

```
void pinq_curve_facs (

  Pint            ws_type,                  /* workstation type                */
  Pint            num_elems_appl_list,      /* # elems in appl list            */
  Pint            start_ind,                /* starting index                  */
  Pint            *err_ind,                 /* OUT error indicator             */
  Pint            *max_order,               /* OUT maximum non-uniform         */
                                            /* b-spline                        */
                                            /* curve order supported           */
  Pint_list       *types,                   /* OUT list of available curve     */
                                            /* approximation criteria types    */
  Pint            *num_elems_impl_list      /* OUT # elems in impl list        */

);
```

INQUIRE PARAMETRIC SURFACE FACILITIES

```
void pinq_param_surf_facs (

  Pint            ws_type,                  /* workstation type                */
  Pint            num_elems_appl_list,      /* # elems in appl list            */
  Pint            start_ind,                /* starting index                  */
  Pint            *err_ind,                 /* OUT error indicator             */
  Pint_list       *types,                   /* OUT list of available           */
                                            /* parametric surface              */
                                            /* characteristic types            */
  Pint            *num_elems_impl_list,     /* OUT # elems in impl list        */
  Pint            *num_indices              /* OUT number of predefined        */
                                            /* parametric surface indices      */

);
```

INQUIRE B-SPLINE SURFACE FACILITIES

```
void pinq_b_spline_surf_facs (

  Pint            ws_type,                  /* workstation type                */
  Pint            num_elems_appl_list,      /* # elems in appl list            */
  Pint            start_ind,                /* starting index                  */
  Pint            *err_ind,                 /* OUT error indicator             */
  Pint            *max_order,               /* OUT maximum non-uniform         */
                                            /* b-spline surface order          */
                                            /* supported                       */
  Pint_list       *types,                   /* OUT list of available surface   */
                                            /* approximation criteria types    */
  Pint            *num_elems_impl_list      /* OUT # elems in impl list        */

);
```

INQUIRE TRIMMING CURVE FACILITIES

```
void pinq_trim_curve_facs (

  Pint          ws_type,                /* workstation type                */
  Pint          num_elems_appl_list,    /* # elems in appl list            */
  Pint          start_ind,              /* starting index                  */
  Pint          *err_ind,               /* OUT error indicator             */
  Pint          *max_order,             /* OUT maximum trimming curve      */
                                        /* order supported                 */
  Pint_list     *types,                 /* OUT list of available trimming  */
                                        /* curve approximation criteria    */
                                        /* types                           */
  Pint          *num_elems_impl_list    /* OUT # elems in impl list        */

);
```

INQUIRE PREDEFINED PARAMETRIC SURFACE REPRESENTATION

```
void pinq_pred_param_surf_rep (

  Pint              ws_type,            /* workstation type                */
  Pint              index,              /* predefined parametric surface   */
                                        /* index                           */
  Pstore            store,              /* handle to Store object          */
  Pint              *err_ind,           /* OUT error indicator             */
  Pparam_surf_rep   **rep               /* OUT parametric surface          */
                                        /* representation                  */

);
```

NOTE — The memory referenced by `*rep` is managed by `store`.

INQUIRE PREDEFINED PATTERN REPRESENTATION PLUS

```
void pinq_pred_pat_rep_plus (

  Pint              ws_type,            /* workstation type                */
  Pint              index,              /* predefined pattern index        */
  Pstore            store,              /* handle to Store object          */
  Pint              *err_ind,           /* OUT error indicator             */
  Ppat_rep_plus     **rep               /* OUT pattern representation plus*/

);
```

NOTE — The memory referenced by `*rep` is managed by `store`.

INQUIRE LIGHT SOURCE FACILITIES

```
void pinq_light_source_facs (

  Pint            ws_type,                  /* workstation type              */
  Pint            num_elems_appl_list,      /* # elems in appl list          */
  Pint            start_ind,                /* starting index                */
  Pint            *err_ind,                 /* OUT error indicator           */
  Pint_list       *types,                   /* OUT list of available light   */
                                            /* source types                  */
  Pint            *num_elems_impl_list,     /* OUT # elems in impl list      */
  Pint            *max_lights,              /* OUT maximum number of         */
                                            /* simultaneously active         */
                                            /* non-ambient lights            */
  Pint            *num_indices              /* OUT number of predefined light */
                                            /* source indices                */

);
```

INQUIRE PREDEFINED LIGHT SOURCE REPRESENTATION

```
void pinq_pred_light_source_rep (

  Pint                    ws_type,          /* workstation type              */
  Pint                    index,            /* predefined light source index */
  Pint                    *err_ind,         /* OUT error indicator           */
  Plight_source_rep       *rep              /* OUT light source representation*/

);
```

INQUIRE DEPTH CUE FACILITIES

```
void pinq_depth_cue_facs (

  Pint                  ws_type,            /* workstation type              */
  Pint                  num_elems_appl_list,/* # elems in appl list          */
  Pint                  start_ind,          /* starting index                */
  Pint                  *err_ind,           /* OUT error indicator           */
  Pint                  *num_indices        /* OUT number of predefined depth */
                                            /* cue indices                   */
  Pdepth_cue_mode_list  *modes,             /* OUT list of available depth   */
                                            /* cue modes                     */
  Pint                  *num_elems_impl_list/* OUT # elems impl list        */

);
```

INQUIRE PREDEFINED DEPTH CUE REPRESENTATION

```
void pinq_pred_depth_cue_rep (

  Pint              ws_type,          /* workstation type                */
  Pint              index,            /* predefined depth cue index      */
  Pint              *err_ind,         /* OUT error indicator             */
  Pdepth_cue_rep    *rep              /* OUT depth cue representation    */

);
```

INQUIRE COLOUR MAPPING FACILITIES

```
void pinq_colr_map_facs (

  Pint          ws_type,              /* workstation type                */
  Pint          num_elems_appl_list,  /* # elems in appl list            */
  Pint          start_ind,            /* starting index                  */
  Pint          *err_ind,             /* OUT error indicator             */
  Pint_list     *methods,             /* OUT list of available colour    */
                                      /* mapping methods                 */
  Pint          *num_elems_impl_list  /* OUT # elems in impl list        */
  Pint          *num_indices          /* OUT number of predefined colour*/
                                      /* mapping indices                 */

);
```

INQUIRE COLOUR MAPPING METHOD FACILITIES

```
void pinq_colr_map_method_facs (

  Pint                  ws_type,      /* workstation type                */
  Pint                  method,       /* colour mapping methods          */
  Pstore                store,        /* handle to Store object          */
  Pint                  *err_ind,     /* OUT error indicator             */
  Pcolr_map_method      **data        /* OUT data record                 */

);
```

NOTE — The memory referenced by `*data` is managed by `store`.

INQUIRE PREDEFINED COLOUR MAPPING REPRESENTATION

```
void pinq_pred_colr_map_rep (

  Pint                ws_type,            /* workstation type                */
  Pint                index,              /* predefined colour mapping index*/
  Pstore              store,              /* handle to Store object          */
  Pint                *err_ind,           /* OUT error indicator             */
  Pcolr_map_rep       **rep               /* OUT colour mapping              */
                                          /* representation                  */

);
```

NOTE — The memory referenced by `*rep` is managed by `store`.

INQUIRE WORKSTATION STATE TABLE LENGTHS PLUS

```
void pinq_ws_st_table_length_plus (

  Pint                      ws_type,      /* workstation type                */
  Pint                      *err_ind,     /* OUT error indicator             */
  Pws_st_tables_plus        *lengths      /* OUT lengths of workstation      */
                                          /* tables                          */

);
```

Annex A

(infomative)

Data types in compilation order and external functions

A.1 Macro definitions

Clause A.1, add the following macro definitions after the macro definition for `PE_BAD_COLR`*:*

```
/* Output Attribute Errors */
#define PE_BAD_DEPTH_CUE_MODE              (119) /* Ignoring function, the spec-
                                                    ified depth cue mode is not
                                                    available on the workstation
                                                    */
#define PE_DEPTH_CUE_INDX_LT_1             (120) /* Ignoring function, the
                                                    depth cue index is less
                                                    than one */
#define PE_COLR_MAP_INDX_LT_1              (121) /* Ignoring function, the
                                                    colour mapping index is
                                                    less than one */
#define PE_BAD_LINE_SHAD_METHOD            (122) /* Ignoring function, the spec-
                                                    ified polyline shading
                                                    method is not available on
                                                    the workstation */
#define PE_BAD_INT_SHAD_METHOD             (123) /* Ignoring function, the spec-
                                                    ified interior shading
                                                    method is not available on
                                                    the workstation */
#define PE_BAD_REFL_MODEL                  (124) /* Ignoring function, the spec-
                                                    ified reflectance model is
                                                    not available on the work-
                                                    station */
#define PE_TOTAL_COLR_RANGE_FIELDS_TOO_LARGE   (125)
                                                  /* Ignoring function, the
                                                    total of the colour range
                                                    fields in all the table
                                                    entries is too large */
#define PE_BAD_COLR_MAP_METHOD             (126) /* Ignoring function, the spec-
                                                    ified colour mapping method
                                                    is not available on the
                                                    specified workstation */
#define PE_BAD_APPROX_CRIT_TYPE            (127) /* Ignoring function, the spec-
                                                    ified approximation crite-
                                                    ria type is not available
                                                    on the specified workstation
                                                    */
```

```
#define PE_BAD_PARAM_SURF_CHAR_TYPE      (128) /* Ignoring function, the spec-
                                                  ified parametric surface
                                                  characteristics type is not
                                                  available on the specified
                                                  workstation */
#define PE_LIGHT_SOURCE_INDX_LT_1        (129) /* Ignoring function, the
                                                  light source index is less
                                                  than one */
#define PE_INVALID_REF_PLANES            (130) /* Ignoring function, invalid
                                                  reference planes; DQMIN >
                                                  DQMAX */
#define PE_BAD_LIGHT_SOURCE_TYPE         (131) /* Ignoring function, the spec-
                                                  ified light source type is
                                                  not available on the work-
                                                  station */
#define PE_SPOTLIGHT_SPREAD_ANGLE_RANGE  (132) /* Ignoring function, the spec-
                                                  ified spot light spread
                                                  angle is out of range */
#define PE_BAD_ACTIVATION_LIST           (133) /* Ignoring function, one of
                                                  the entries in the activa-
                                                  tion list or deactivation
                                                  list is less than one */
#define PE_GEN_COLR_NOT_INDIRECT         (134) /* Ignoring function, the
                                                  requested entry contains a
                                                  general colour specifica-
                                                  tion with colour type other
                                                  than INDIRECT */
#define PE_SAME_ACTIVATION_DEACTIVATION_ENTRY (135)
                                                  /* Ignoring function, the same
                                                  entry exists in both the
                                                  activation and the deactiva-
                                                  tion list */
#define PE_COLR_COMPONENT_RANGE          (136) /* Ignoring function, one of
                                                  the components of the
                                                  colour specification is out
                                                  of range */
#define PE_BAD_DATA_MAP_METHOD           (137) /* Ignoring function, the spec-
                                                  ified data mapping method
                                                  is not available on the
                                                  specified workstation */
#define PE_BAD_DATA_RECORD_FIELD         (138) /* Ignoring function, one or
                                                  more of the fields within
                                                  the specified data record
                                                  is in error */
#define PE_BAD_REFL_PROP_TYPE            (139) /* Ignoring function, the spec-
                                                  ified reflectance property
                                                  type is not available on
                                                  the specified workstation */
#define PE_INVALID_DEPTH_CUE_SCALE       (140) /* Ignoring function, one of
                                                  the depth cue scale factors
```

```
                                                    is not in the required range
                                                    */
```

Clause A.1, add the following macro definitions after the macro definition for PE_BAD_ERROR_FILE*:*

```
/* Output Primitive Errors */
#define PE_ORDER_LT_1                       (500) /* Ignoring function, the
                                                     order of a spline is less
                                                     than one */
#define PE_INSUFF_CTRL_POINTS_FOR_ORDER     (501) /* Ignoring function, a spline
                                                     does not contain enough con-
                                                     trol points for its speci-
                                                     fied order */
#define PE_ORDER_INCONSISTENT_KNOTS_CTRL_POINTS   (502)
                                                  /* Ignoring function, the
                                                     order of a spline is incon-
                                                     sistent with number of
                                                     knots and control points */
#define PE_KNOT_SEQUENCE_NOT_NONDECREASING  (503)
                                                  /* Ignoring function, the knot
                                                     sequence for a spline is
                                                     not non-decreasing */
#define PE_VERTEX_INDX_OUT_OF_RANGE         (504) /* Ignoring function, one or
                                                     more of the vertex indices
                                                     is out of range */
#define PE_PARAM_RANGE_INCONSISTENT_KNOTS   (506)
                                                  /* Ignoring function, the
                                                     parameter range for a
                                                     spline is inconsistent with
                                                     its knots */
#define PE_W_VALUE_LE_0                     (507) /* Ignoring function, the
                                                     fourth coordinate of a
                                                     rational control point is
                                                     less than or equal to zero */
#define PE_TRIM_CURVE_ORDER_LT_2            (508) /* Ignoring function, a trim-
                                                     ming curve's order is less
                                                     than two */
#define PE_INCONSISTENT_EDGE_FLAG_SPEC      (513) /* Ignoring function, the num-
                                                     ber of edge visibility
                                                     flags is inconsistent with
                                                     the number of edges in the
                                                     primitive */
#define PE_DATA_LIST_LENGTH_NOT_SAME        (514) /* Ignoring function, the data
                                                     lists do not all contain
                                                     the same number of entries */
#define PE_INCONSISTENT_FACET_SPEC          (516) /* Ignoring function, the
                                                     facet data is inconsistent
                                                     with the number of facets
                                                     in the primitive */
```

```
#define PE_PARAM_RANGE_COLR_SPLINE      (517) /* Ignoring function, the
                                                 parameter range of the
                                                 colour spline does not
                                                 include the effective param-
                                                 eter range of the geometry
                                                 spline */
#define PE_PARAM_RANGE_DATA_SPLINE      (518) /* Ignoring function, the
                                                 parameter range of the data
                                                 spline does not include the
                                                 effective parameter range
                                                 of the geometry spline */
#define PE_BAD_VERTEX_DATA_TYPE         (2207)/* Ignoring function, two or
                                                 more vertices do not con-
                                                 tain the same type of data*/
```

Clause A.1, add the following macro definitions after the macro definition for `Pfn_set_err_hand`:

```
#define Pfn_polyline_set3_colr                (183)
#define Pfn_fill_area_set3_data               (184)
#define Pfn_fill_area_set_data                (185)
#define Pfn_cell_array3_plus                  (186)
#define Pfn_set_of_fill_area_sets3_data       (187)
#define Pfn_set_of_fill_area_sets_data        (188)
#define Pfn_tri_set3_data                     (189)
#define Pfn_tri_set_data                      (190)
#define Pfn_tri_strip3_data                   (191)
#define Pfn_tri_strip_data                    (192)
#define Pfn_quad_mesh3_data                   (193)
#define Pfn_quad_mesh_data                    (194)
#define Pfn_non_uniform_b_spline_curve3       (195)
#define Pfn_non_uniform_b_spline_curve3_colr  (196)
#define Pfn_non_uniform_b_spline_surf3        (197)
#define Pfn_non_uniform_b_spline_surf3_data   (198)
#define Pfn_set_data_map_ind                  (199)
#define Pfn_set_refl_ind                      (200)
#define Pfn_set_back_int_ind                  (201)
#define Pfn_set_back_data_map_ind             (202)
#define Pfn_set_back_refl_ind                 (203)
#define Pfn_set_param_surf_ind                (204)
#define Pfn_set_line_colr                     (205)
#define Pfn_set_line_shad_method              (206)
#define Pfn_set_marker_colr                   (207)
#define Pfn_set_text_colr                     (208)
#define Pfn_set_facet_disting_mode            (209)
#define Pfn_set_facet_cull_mode               (210)
#define Pfn_set_int_colr                      (211)
#define Pfn_set_int_shad_method               (212)
#define Pfn_set_data_map_method               (213)
#define Pfn_set_refl_props                    (214)
#define Pfn_set_refl_model                    (215)
```

```
#define Pfn_set_back_int_style                (216)
#define Pfn_set_back_int_style_ind            (217)
#define Pfn_set_back_int_colr                 (218)
#define Pfn_set_back_int_shad_method          (219)
#define Pfn_set_back_data_map_method          (220)
#define Pfn_set_back_refl_props               (221)
#define Pfn_set_back_refl_model               (222)
#define Pfn_set_light_source_st               (223)
#define Pfn_set_edge_colr                     (224)
#define Pfn_set_curve_approx_crit             (225)
#define Pfn_set_surf_approx_crit              (226)
#define Pfn_set_param_surf_chars              (227)
#define Pfn_set_rend_colr_model               (228)
#define Pfn_set_depth_cue_ind                 (229)
#define Pfn_set_colr_map_ind                  (230)
#define Pfn_set_line_rep_plus                 (231)
#define Pfn_set_marker_rep_plus               (232)
#define Pfn_set_text_rep_plus                 (233)
#define Pfn_set_int_rep_plus                  (234)
#define Pfn_set_edge_rep_plus                 (235)
#define Pfn_set_data_map_rep                  (236)
#define Pfn_set_refl_rep                      (237)
#define Pfn_set_param_surf_rep                (238)
#define Pfn_set_pat_rep_plus                  (239)
#define Pfn_set_light_source_rep              (240)
#define Pfn_set_depth_cue_rep                 (241)
#define Pfn_set_colr_map_rep                  (242)
```

Clause A.1, add the following macro definitions after the macro definition for `PMODEL_HLS`*:*

```
#define PCOLR_MAP_METHOD_TRUE                 (1)
#define PCOLR_MAP_METHOD_PSEUDO               (2)
#define PCOLR_MAP_METHOD_PSEUDO_N             (3)

#define PCOLR_INDIRECT                        (0)
#define PCOLR_RGB                             (1)
#define PCOLR_CIELUV                          (2)
#define PCOLR_HSV                             (3)
#define PCOLR_HLS                             (4)

#define PCURVE_APPROX_WS_DEP                  (1)
#define PCURVE_APPROX_CONST_PARAM_SUBD        (2)
#define PCURVE_APPROX_CHORDAL_SIZE_WC         (3)
#define PCURVE_APPROX_CHORDAL_SIZE_NPC        (4)
#define PCURVE_APPROX_CHORDAL_SIZE_DC         (5)
#define PCURVE_APPROX_CHORDAL_DEV_WC          (6)
#define PCURVE_APPROX_CHORDAL_DEV_NPC         (7)
#define PCURVE_APPROX_CHORDAL_DEV_DC          (8)
#define PCURVE_APPROX_REL_WC                  (9)
#define PCURVE_APPROX_REL_NPC                 (10)
```

```
#define PCURVE_APPROX_REL_DC                    (11)

#define PDATA_MAP_METHOD_COLR                   (1)
#define PDATA_MAP_METHOD_SINGLE_UNIFORM         (2)
#define PDATA_MAP_METHOD_SINGLE_NON_UNIFORM     (3)
#define PDATA_MAP_METHOD_BI_UNIFORM             (4)
#define PDATA_MAP_METHOD_BI_NON_UNIFORM         (5)

#define PINT_SHAD_METHOD_NONE                   (1)
#define PINT_SHAD_METHOD_COLR                   (2)
#define PINT_SHAD_METHOD_DATA                   (3)
#define PINT_SHAD_METHOD_DATA_DOT               (4)
#define PINT_SHAD_METHOD_DATA_NORM              (5)

#define PLIGHT_SOURCE_AMB                       (1)
#define PLIGHT_SOURCE_DIR                       (2)
#define PLIGHT_SOURCE_POS                       (3)
#define PLIGHT_SOURCE_SPOT                      (4)

#define PPARAM_SURF_CHARS_NONE                  (1)
#define PPARAM_SURF_CHARS_WS_DEP                (2)
#define PPARAM_SURF_CHARS_ISOPARAM              (3)
#define PPARAM_SURF_CHARS_LEVEL_MC              (4)
#define PPARAM_SURF_CHARS_LEVEL_WC              (5)

#define PLINE_SHAD_METHOD_NONE                  (1)
#define PLINE_SHAD_METHOD_COLR                  (2)

#define PREFL_MODEL_NO_REFL                     (1)
#define PREFL_MODEL_AMB_REFL                    (2)
#define PREFL_MODEL_AMB_DIFF_REFL               (3)
#define PREFL_MODEL_AMB_DIFF_SPEC_REFL          (4)

#define PREFL_PROPS_SIMPLE_REFL                 (1)

#define PREND_COLR_MODEL_WS_DEP                 (0)
#define PREND_COLR_MODEL_RGB                    (1)
#define PREND_COLR_MODEL_CIELUV                 (2)
#define PREND_COLR_MODEL_HSV                    (3)
#define PREND_COLR_MODEL_HLS                    (4)

#define PSURF_APPROX_WS_DEP                     (1)
#define PSURF_APPROX_CONST_PARAM_SUBD           (2)
#define PSURF_APPROX_CHORDAL_SIZE_WC            (3)
#define PSURF_APPROX_CHORDAL_SIZE_NPC           (4)
#define PSURF_APPROX_CHORDAL_SIZE_DC            (5)
#define PSURF_APPROX_PLANAR_DEV_WC              (6)
#define PSURF_APPROX_PLANAR_DEV_NPC             (7)
#define PSURF_APPROX_PLANAR_DEV_DC              (8)
#define PSURF_APPROX_REL_WC                     (9)
#define PSURF_APPROX_REL_NPC                    (10)
```

```
#define PSURF_APPROX_REL_DC                    (11)
```

Clause A.1, add the following macro definitions after the macro definition for PLAST_PHIGS_ELEM*:*

```
#define PFIRST_PHIGS_PLUS_ELEM                 (PELEM_POLYLINE_SET3_COLR)
#define PLAST_PHIGS_PLUS_ELEM                  (PELEM_COLR_MAP_IND)
```

A.2 Types in compilation order

Clause A.2, modify the following PHIGS type definitions by adding the specified PHIGS PLUS enumerations:

```
typedef enum {
  /* start of PHIGS enumerations */
  ...,
  /* end of PHIGS enumerations */
  /* start of PHIGS PLUS enumerations */
  PASPECT_LINE_COLR,
  PASPECT_MARKER_COLR,
  PASPECT_TEXT_COLR,
  PASPECT_INT_COLR,
  PASPECT_EDGE_COLR,
  PASPECT_LINE_SHAD_METHOD,
  PASPECT_INT_SHAD_METHOD,
  PASPECT_DATA_MAP_METHOD,
  PASPECT_REFL_PROPS,
  PASPECT_REFL_MODEL,
  PASPECT_BACK_INT_STYLE,
  PASPECT_BACK_INT_STYLE_IND,
  PASPECT_BACK_INT_COLR,
  PASPECT_BACK_INT_SHAD_METHOD,
  PASPECT_BACK_DATA_MAP_METHOD,
  PASPECT_BACK_REFL_PROPS,
  PASPECT_BACK_REFL_MODEL,
  PASPECT_CURVE_APPROX_CRIT,
  PASPECT_SURF_APPROX_CRIT,
  PASPECT_PARAM_SURF_CHARS
  /* end of PHIGS PLUS enumerations */
} Paspect;
```

```
typedef enum {
  /* start of PHIGS enumerations */
  ...,
  /* end of PHIGS enumerations */
  /* start of PHIGS PLUS enumerations */
  PATTR_REFL,
  PATTR_PARAM_SURF
  /* end of PHIGS PLUS enumerations */
} Pattrs;
```

```
typedef enum {
  /* start of PHIGS enumerations */
  ...,
  /* end of PHIGS enumerations */
  /* start of PHIGS PLUS enumerations */
  PELEM_POLYLINE_SET3_COLR,
  PELEM_FILL_AREA_SET3_DATA,
  PELEM_FILL_AREA_SET_DATA,
  PELEM_CELL_ARRAY3_PLUS,
  PELEM_SET_OF_FILL_AREA_SETS3_DATA,
  PELEM_SET_OF_FILL_AREA_SETS_DATA,
  PELEM_TRI_SET3_DATA,
  PELEM_TRI_SET_DATA,
  PELEM_TRI_STRIP3_DATA,
  PELEM_TRI_STRIP_DATA,
  PELEM_QUAD_MESH3_DATA,
  PELEM_QUAD_MESH_DATA,
  PELEM_NON_UNIFORM_B_SPLINE_CURVE3,
  PELEM_NON_UNIFORM_B_SPLINE_CURVE3_COLR,
  PELEM_NON_UNIFORM_B_SPLINE_SURF3,
  PELEM_NON_UNIFORM_B_SPLINE_SURF3_DATA,
  PELEM_DATA_MAP_IND,
  PELEM_REFL_IND,
  PELEM_BACK_INT_IND,
  PELEM_BACK_DATA_MAP_IND,
  PELEM_BACK_REFL_IND,
  PELEM_PARAM_SURF_IND,
  PELEM_LINE_COLR,
  PELEM_LINE_SHAD_METHOD,
  PELEM_MARKER_COLR,
  PELEM_TEXT_COLR,
  PELEM_FACET_DISTING_MODE,
  PELEM_FACET_CULL_MODE,
  PELEM_INT_COLR,
  PELEM_INT_SHAD_METHOD,
  PELEM_DATA_MAP_METHOD,
  PELEM_REFL_PROPS,
  PELEM_REFL_MODEL,
  PELEM_BACK_INT_STYLE,
  PELEM_BACK_INT_STYLE_IND,
  PELEM_BACK_INT_COLR,
  PELEM_BACK_INT_SHAD_METHOD,
  PELEM_BACK_DATA_MAP_METHOD,
  PELEM_BACK_REFL_PROPS,
  PELEM_BACK_REFL_MODEL,
  PELEM_LIGHT_SOURCE_ST,
  PELEM_EDGE_COLR,
  PELEM_CURVE_APPROX_CRIT,
  PELEM_SURF_APPROX_CRIT,
  PELEM_PARAM_SURF_CHARS,
  PELEM_REND_COLR_MODEL,
```

```
  PELEM_DEPTH_CUE_IND,
  PELEM_COLR_MAP_IND
  /* end of PHIGS PLUS enumerations */
} Pelem_type;
```

Clause A.2, add the following type definitions after the type definition for `Pattrs`*:*

```
typedef enum {
  PDEPTH_CUE_SUPPR,
  PDEPTH_CUE_ALLOWED
} Pdepth_cue_mode;


typedef enum {
  PFACET_CULL_NONE,
  PFACET_CULL_BACKFACING,
  PFACET_CULL_FRONTFACING
} Pfacet_cull_mode;


typedef enum {
  PFACET_DISTING_OFF,
  PFACET_DISTING_ON
} Pfacet_disting_mode;


typedef enum {
  PPLACE_UNIFORM_OVER_SURF,
  PPLACE_UNIFORM_BETWEEN_KNOTS
} Pplacement;


typedef enum {
  PB_SPLINE_RATIONAL,
  PB_SPLINE_NON_RATIONAL
} Prationality;


typedef enum {
  PSOURCE_SELECT_COLR_ASPECT,
  PSOURCE_SELECT_VERT_COLR,
  PSOURCE_SELECT_VERT_DATA,
  PSOURCE_SELECT_FACET_COLR,
  PSOURCE_SELECT_FACET_DATA
} Psource_select;
```

```
typedef enum {
  PVIS_OFF,
  PVIS_ON
} Pvis_flag;
```

Clause A.2, add the following PHIGS PLUS types after the PHIGS type `Pescape_in_data`*:*

```
typedef struct {
  Pint              num_floats;          /* number of floats in list      */
  Pfloat            *floats;             /* list of floats                */
} Pfloat_list;


typedef struct {
  Pint              num_lists;           /* number of float lists         */
  Pfloat_list       *floats;             /* list of float lists           */
} Pfloat_set;


typedef union {
  Prgb              *rgb;                /* pointer to RGB colour values  */
  Pcieluv           *cieluv;             /* pointer to CIELUV colour values*/
  Phls              *hls;                /* pointer to HLS colour values  */
  Phsv              *hsv;                /* pointer to HSV colour values  */
  Pdata             *unsupp;             /* pointer to unsupported colour */
                                         /* values                        */
  int               impl_dep;            /* implementation defined        */
} Pcolr_rep_ptr;


typedef union {
  Pint              *colr_inds;          /* pointer to colour indices     */
  Pcolr_rep_ptr     colr_reps;           /* pointer to colour             */
                                         /* representations               */
} Pcolrv_ptr;


typedef struct {
  Pint              num_colrs;           /* number of colours             */
  Pcolrv_ptr        colrs;               /* list of colour values         */
} Pcolrv_list;


typedef struct {
  Pint              num_lists;           /* number of colour value lists  */
  Pcolrv_list       *colrs;              /* list of colour value lists    */
} Pcolrv_set;
```

```
typedef struct {
  Pint          method;                   /* colour mapping method          */
  union Pcolr_map_method_data {
    Pint        num_true_colrs;           /* number of true colours         */
    Pint        max_num_pseudo_colrs;     /* maximum number of pseudo       */
                                          /* colours                        */
    int         impl_dep;                 /* implementation defined         */
  } data;
} Pcolr_map_method;


typedef struct {
  Pint              method;               /* colour mapping method          */
  union Pcolr_map_rep_data {
    struct Pcolr_map_rep_pseudo {
      Pint          model;                /* colour model                   */
      Pfloat_list   weight_vector;        /* list of weight vector          */
      Pcolrv_list   colrs;                /* list of colour values          */
    } pseudo;
    struct Pcolr_map_rep_pseudo_N {
      Pint          model;                /* colour model                   */
      Pfloat_set    colrs;                /* list of lists of colour        */
                                          /* coordinates                    */
    } pseudo_N;
    int         impl_dep;                 /* implementation defined         */
  } data;
} Pcolr_map_rep;


typedef struct {
  Pint          method;                   /* colour mapping method          */
  union Pcolr_map_st_data {
    Pint        num_true_colrs;           /* number of available true       */
                                          /* colours                        */
    Pint        max_num_pseudo_colrs;     /* maximum number of pseudo       */
                                          /* colour entries                 */
    int         impl_dep;                 /* implementation defined         */
  } data;
} Pcolr_map_st;
```

```
typedef struct {
  Pint              type;                 /* curve approximation criteria   */
                                          /* type                           */
  union Pcurve_approx_crit_data {
    struct Pcurve_approx_const_param_subd {
      Pint          count;                /* count                          */
    } const_param_subd;
    struct Pcurve_approx_chordal_size_wc {
      Pfloat        approx;               /* approximation value            */
    } chordal_size_wc;
    struct Pcurve_approx_chordal_size_npc {
      Pfloat        approx;               /* approximation value            */
    } chordal_size_npc;
    struct Pcurve_approx_chordal_size_dc {
      Pfloat        approx;               /* approximation value            */
    } chordal_size_dc;
    struct Pcurve_approx_chordal_dev_wc {
      Pfloat        approx;               /* approximation value            */
    } chordal_dev_wc;
    struct Pcurve_approx_chordal_dev_npc {
      Pfloat        approx;               /* approximation value            */
    } chordal_dev_npc;
    struct Pcurve_approx_chordal_dev_dc {
      Pfloat        approx;               /* approximation value            */
    } chordal_dev_dc;
    struct Pcurve_approx_rel_wc {
      Pfloat        approx;               /* approximation value            */
    } rel_wc;
    struct Pcurve_approx_rel_npc {
      Pfloat        approx;               /* approximation value            */
    } rel_npc;
    struct Pcurve_approx_rel_dc {
      Pfloat        approx;               /* approximation value            */
    } rel_dc;
    Pdata           unsupp;               /* unsupported curve approximation*/
                                          /* criteria data                  */
    int             impl_dep;             /* implementation defined         */
  } data;
} Pcurve_approx_crit;


typedef struct {
  Pint              num_selectors;        /* number of selectors            */
  Psource_select    *selectors;           /* list of source selectors       */
} Psource_select_list;
```

```
typedef struct {
  Pint                      method;          /* data mapping method              */
  Psource_select_list       selectors;       /* list of source selectors         */
  union Pdata_map_rep_data {
    struct Pdata_map_rep_single_uniform {
      Pint                  ind;             /* data value index                 */
      Pfloat                lower_limit;     /* lower range limit                */
      Pfloat                upper_limit;     /* upper range limit                */
      Pint                  colr_type;       /* colour type                      */
      Pcolrv_list           colr_values;     /* list of colour values            */
    } single_uniform;
    struct Pdata_map_rep_single_non_uniform {
      Pint                  ind;             /* data value index                 */
      Pfloat_list           range;           /* range boundaries                 */
      Pint                  colr_type;       /* colour type                      */
      Pcolrv_list           colr_values;     /* list of colour values            */
    } single_non_uniform;
    struct Pdata_map_rep_bi_uniform {
      Pint                  inds[2];         /* data value indices               */
      Pfloat                lower_limit_a;   /* lower limit of Ra range          */
      Pfloat                upper_limit_a;   /* upper limit of Ra range          */
      Pfloat                lower_limit_b;   /* lower limit of Rb range          */
      Pfloat                upper_limit_b;   /* upper limit of Rb range          */
      Pint                  colr_type;       /* colour type                      */
      Pcolrv_set            colr_values;     /* list of colour value lists       */
    } bi_uniform;
    struct Pdata_map_rep_bi_non_uniform {
      Pint                  inds[2];         /* data value indices               */
      Pfloat_list           range_a;         /* Ra range boundaries              */
      Pfloat_set            range_b;         /* Rb range boundaries              */
      Pint                  colr_type;       /* colour type                      */
      Pcolrv_set            colr_values;     /* array of colour values           */
    } bi_non_uniform;
    int                     impl_dep;        /* implementation defined           */
  } data;
} Pdata_map_rep;


typedef struct {
  Pfloat            cieluv_x;                /* x coefficient                    */
  Pfloat            cieluv_y;                /* y coefficient                    */
  Pfloat            cieluv_y_lum;            /* y luminance                      */
  Pfloat            w;                       /* homogeneous coordinate           */
} Phomo_cieluv;
```

```
typedef struct {
  Pfloat            hue;              /* hue                          */
  Pfloat            lightness;        /* lightness                    */
  Pfloat            satur;            /* saturation                   */
  Pfloat            w;                /* homogeneous coordinate       */
} Phomo_hls;


typedef struct {
  Pfloat            hue;              /* hue                          */
  Pfloat            satur;            /* saturation                   */
  Pfloat            value;            /* value                        */
  Pfloat            w;                /* homogeneous coordinate       */
} Phomo_hsv;


typedef struct {
  Pfloat            red;              /* red                          */
  Pfloat            green;            /* green                        */
  Pfloat            blue;             /* blue                         */
  Pfloat            w;                /* homogeneous coordinate       */
} Phomo_rgb;


typedef union {
  Phomo_rgb         *rgb;             /* pointer to homogeneous rgb   */
                                      /* colour value                 */
  Phomo_cieluv      *cieluv;          /* pointer to homogeneous cieluv*/
                                      /* colour value                 */
  Phomo_hls         *hls;             /* pointer to homogeneous hls   */
                                      /* colour value                 */
  Phomo_hsv         *hsv;             /* pointer to homogeneous hsv   */
                                      /* colour value                 */
  int               impl_dep;         /* implementation defined       */
} Phomo_colr_rep_ptr;


typedef union {
  Pint                  colr_ind;     /* colour index                 */
  Pcolr_rep             colr_rep;     /* colour representation        */
} Pcolrv;


typedef struct {
  Pint              colr_type;        /* colour type                  */
  Pcolrv            colr_value;       /* colour value                 */
} Pgcolr;
```

```
typedef struct {
  Pint              type;                 /* light source type              */
  union Plight_source_rep_data {
    struct Plight_source_rep_amb {
      Pgcolr        colr;                 /* light source colour            */
    } amb;
    struct Plight_source_rep_dir {
      Pgcolr        colr;                 /* light source colour            */
      Pvec3         dir;                  /* light source direction         */
    } dir;
    struct Plight_source_rep_pos {
      Pgcolr        colr;                 /* light source colour            */
      Ppoint3       pos;                  /* light source position          */
      Pfloat        c1;                   /* light source attenuation       */
                                          /* coefficient                    */
      Pfloat        c2;                   /* light source attenuation       */
                                          /* coefficient                    */
    } pos;
    struct Plight_source_rep_spot {
      Pgcolr        colr;                 /* light source colour            */
      Ppoint3       pos;                  /* light source position          */
      Pvec3         dir;                  /* light source direction         */
      Pfloat        concent_exp;          /* concentration exponent         */
      Pfloat        c1;                   /* light source attenuation       */
                                          /* coefficient                    */
      Pfloat        c2;                   /* light source attenuation       */
                                          /* coefficient                    */
      Pfloat        spread_angle;         /* light source spread angle      */
    } spot;
    int             impl_dep;             /* implementation defined         */
  } data;
} Plight_source_rep;
```

```
typedef struct {
  Pint                type;                 /* parametric surface             */
                                            /* characteristics type           */

  union Pparam_surf_chars_data {
    struct Pparam_surf_isoparam {
      Pplacement      placement;            /* curve placement: UNIFORM OVER  */
                                            /* SURFACE, UNIFORM BETWEEN KNOTS */
      Pint            num_u_curves;         /* curve count in u direction     */
      Pint            num_v_curves;         /* curve count in v direction     */
    } isoparam;
    struct Pparam_surf_level_mc {
      Ppoint3         origin;               /* origin point                   */
      Pvec3           dir;                  /* direction vector               */
      Pfloat_list     params;               /* parameter list                 */
    } level_mc;
    struct Pparam_surf_level_wc {
      Ppoint3         origin;               /* origin point                   */
      Pvec3           dir;                  /* direction vector               */
      Pfloat_list     params;               /* parameter list                 */
    } level_wc;
    Pdata             unsupp;               /* unsupported parametric         */
                                            /* surface characteristics data   */
    int               impl_dep;             /* implementation defined         */
  } data;
} Pparam_surf_chars;


typedef struct {
  Pint                type;                 /* reflectance property type      */
  union Prefl_prop_data {
    struct Prefl_prop_simple_refl {
      Pfloat          amb_coeff;            /* ambient reflection             */
                                            /* coefficient                    */
      Pfloat          diff_coeff;           /* diffuse reflection             */
                                            /* coefficient                    */
      Pfloat          spec_coeff;           /* specular reflection            */
                                            /* coefficient                    */
      Pgcolr          spec_colr;            /* specular colour                */
      Pfloat          spec_exp;             /* specular exponent              */
    } simple_refl;
    Pdata             unsupp;               /* unsupported reflectance        */
                                            /* property data                  */
    int               impl_dep;             /* implementation defined         */
  } data;
} Prefl_prop;
```

```
typedef struct {
  Pint              type;                  /* surface approximation criteria */
                                           /* type                           */
  union Psurf_approx_crit_data {
    struct Psurf_approx_const_param_subd {
      Pint          u_count;               /* u count                        */
      Pint          v_count;               /* v count                        */
    } const_param_subd;
    struct Psurf_approx_chordal_size_wc {
      Pfloat        u_approx;              /* u approximation value          */
      Pfloat        v_approx;              /* v approximation value          */
    } chordal_size_wc;
    struct Psurf_approx_chordal_size_npc {
      Pfloat        u_approx;              /* u approximation value          */
      Pfloat        v_approx;              /* v approximation value          */
    } chordal_size_npc;
    struct Psurf_approx_chordal_aize_dc {
      Pfloat        u_approx;              /* u approximation value          */
      Pfloat        v_approx;              /* v approximation value          */
    } chordal_size_dc;
    struct Psurf_approx_planar_dev_wc {
      Pfloat        approx;                /* approximation value            */
    } planar_dev_wc;
    struct Psurf_approx_planar_dev_npc {
      Pfloat        approx;                /* approximation value            */
    } planar_dev_npc;
    struct Psurf_approx_planar_dev_dc {
      Pfloat        approx;                /* approximation value            */
    } planar_dev_dc;
    struct Psurf_approx_rel_wc {
      Pfloat        approx;                /* approximation value            */
    } rel_wc;
    struct Psurf_approx_rel_npc {
      Pfloat        approx;                /* approximation value            */
    } rel_npc;
    struct Psurf_approx_rel_dc {
      Pfloat        approx;                /* approximation value            */
    } rel_dc;
    Pdata           unsupp;                /* unsupported surface            */
                                           /* approximation criteria data    */
    int             impl_dep;              /* implementation defined         */
  } data;
} Psurf_approx_crit;


typedef union {
  Pcolr_rep_ptr         points;            /* pointer to colour points       */
  Phomo_colr_rep_ptr    homo_points;       /* pointer to homogeneous colour  */
                                           /* points                         */
} Pcolr_ctrl_point_ptr;
```

```
typedef struct {
  Prationality          rationality;      /* control point rationality      */
  Pint_size             dims;             /* dimensions of control point    */
                                          /* array                          */
  Pcolr_ctrl_point_ptr  colr_points;      /* array of colour control points */
} Pcolr_ctrl_point_array;


typedef struct {
  Prationality          rationality;      /* control point rationality      */
  Pint                  num_points;       /* number of control points       */
  Pcolr_ctrl_point_ptr  points;           /* list of colour control points  */
} Pcolr_ctrl_point_list;


typedef struct {
  Pint_size         dims;                 /* colour value array dimensions  */
  Pcolrv_ptr        colrs;                /* array of colour values         */
} Pcolrv_array;


typedef struct {
  Pint              num_lists;            /* number of contour lists        */
  Pint_list         *contours;            /* list of contour lists          */
} Pcontour_set;


typedef struct {
  Pint              num_contour_sets;     /* number of contour sets         */
  Pcontour_set      *contours;            /* list of contour sets           */
} Pcontour_set_list;


typedef struct {
  Pfloat            wx;                   /* wx coordinate                  */
  Pfloat            wy;                   /* wy coordinate                  */
  Pfloat            wz;                   /* wz coordinate                  */
  Pfloat            w;                    /* homogeneous coordinate         */
} Phomo_point4;


typedef union {
  Ppoint3           *point3;              /* pointer to 3D non-homogeneous  */
                                          /* points                         */
  Phomo_point4      *homo_point4;         /* pointer to 4D homogeneous      */
                                          /* points                         */
} Ppoint3_ptr;
```

```
typedef struct {
  Prationality      rationality;          /* control point rationality      */
  Pint_size         dims;                 /* control point array u,v        */
                                          /* dimensions                     */
  Ppoint3_ptr       points;               /* array of control points        */
} Pctrl_point3_array;


typedef struct {
  Pfloat            wx;                   /* wx coordinate                  */
  Pfloat            wy;                   /* wy coordinate                  */
  Pfloat            w;                    /* homogeneous coordinate         */
} Phomo_point3;


typedef union {
  Ppoint            *point;               /* pointer to 2D non-homogeneous  */
                                          /* points                         */
  Phomo_point3      *homo_point3;         /* pointer to 3D homogeneous      */
                                          /* points                         */
} Ppoint_ptr;


typedef struct {
  Prationality      rationality;          /* control point rationality      */
  Pint              num_points;           /* number of control points       */
  Ppoint_ptr        points;               /* list of control points         */
} Pctrl_point_list;


typedef struct {
  Prationality      rationality;          /* control point rationality      */
  Pint              num_points;           /* number of control points       */
  Ppoint3_ptr       points;               /* list of control points         */
} Pctrl_point3_list;


typedef struct {
  Pint                      order;        /* spline order                   */
  Pfloat_list               knots;        /* list of knot values            */
  Pint                      colr_type;    /* colour type                    */
  Pcolr_ctrl_point_list     colrs;        /* list of colour control points  */
} Pcurve_colr_spline;
```

```
typedef struct {
  Pint                  order;          /* spline order                  */
  Pfloat_list           knots;          /* knots                         */
  Pfloat                low_limit;      /* lower parameter range limit   */
  Pfloat                high_limit;     /* high parameter range limit    */
  Pctrl_point3_list     ctrl_points;    /* list of control points        */
} Pcurve_geom_spline;


typedef struct {
  Prationality      rationality;        /* control point rationality     */
  Pint_size         dims;               /* control point array u,v       */
                                        /* dimensions                    */
  Pint              points_dims;        /* control point dimensionality  */
  Pfloat            *ctrl_points;       /* array of control points       */
} Pdata_ctrl_point_array;


typedef struct {
  Pint              num_modes;          /* number of depth cue modes     */
  Pdepth_cue_mode   *modes;             /* list of depth cue modes       */
} Pdepth_cue_mode_list;


typedef struct {
  Pdepth_cue_mode   mode;               /* depth cue mode                */
  Pfloat            ref_planes[2];      /* depth cue reference planes    */
                                        /* (min, max)                    */
  Pfloat            scale[2];           /* depth cue scale factors       */
                                        /* (min, max)                    */
  Pgcolr            gcolr;              /* depth cue colour              */
} Pdepth_cue_rep;


typedef struct {
  Pdyn_mod          data_map;           /* data mapping representation   */
  Pdyn_mod          refl;               /* reflectance representation    */
  Pdyn_mod          param_surf;         /* parametric surface            */
                                        /* representation                */
  Pdyn_mod          light_source;       /* light source representation   */
  Pdyn_mod          depth_cue;          /* depth cue representation      */
  Pdyn_mod          colr_map;           /* colour mapping representation */
} Pdyn_ws_attrs_plus;
```

```
typedef struct {
  Pedge_flag        flag;                 /* edge flag                    */
  Pint              type;                 /* edgetype                     */
  Pfloat            width;                /* edgewidth scale factor       */
  Pgcolr            colr;                 /* edge colour                  */
} Pedge_bundle_plus;


typedef Pedge_flag Pedge_flag_pair[2];


typedef Pedge_flag Pedge_flag_triplet[3];


typedef struct {
  Pint              num_edge_flag_triplets;/* number of edge flag triplets */
  Pedge_flag_triplet*edge_flag_triplets;  /* list of edge flag triplets   */
} Pedge_flag_triplet_list;


typedef struct {
  Pint_size         dims;                 /* edge flag array dimensions   */
  Pedge_flag_pair   *edge_flags;          /* array of edge flag pairs     */
} Pedge_flag_array;


typedef struct {
  Pint              num_edges;            /* number of edges in list      */
  Pedge_flag        *edge_flags;          /* list of edge flags           */
} Pedge_flag_list;


typedef struct {
  Pint              num_lists;            /* number of edge flag lists    */
  Pedge_flag_list   *edge_flags;          /* list of edge flag lists      */
} Pedge_flag_set;


typedef struct {
  Pint              num_edge_flag_sets;   /* number of edge flag sets     */
  Pedge_flag_set    *edge_flag_sets;      /* list of edge flag sets       */
} Pedge_flag_set_list;


typedef struct {
  Pint              num_data_per_facet;   /* number of data values per-facet*/
  Pcolrv            *facet_colrv;         /* facet colour                 */
  Pvec3             *facet_norm;          /* facet normal                 */
  Pfloat            *facet_data;          /* list of facet data           */
} Pfacet;
```

```
typedef struct {
  Pint_size          dims;                /* facet array dimensions          */
  Pint               num_data_per_facet;  /* number of data values per-facet*/
  Pcolrv_ptr         facet_colrvs;        /* array of facet colour values    */
  Pvec3              *facet_norms;        /* array of facet normals          */
  Pfloat             *facet_data;         /* array of facet data lists       */
} Pfacet_array;


typedef struct {
  Pint               num_facets;          /* number of facets                */
  Pint               num_data_per_facet;  /* number of data values per-facet*/
  Pcolrv_ptr         facet_colrvs;        /* list of facet colour values     */
  Pvec3              *facet_norms;        /* list of facet normals           */
  Pfloat             *facet_data;         /* list of facet data lists        */
} Pfacet_list;


typedef struct {
  Pint_style         style;               /* interior style                  */
  Pint               style_ind;           /* interior style index            */
  Pgcolr             colr;                /* interior colour                 */
  Pint               shad_method;         /* shading method                  */
} Pint_bundle_plus;


typedef struct {
  Pint               num_int_styles;      /* number of interior styles       */
  Pint_style         int_styles[5];       /* list of available interior      */
                                          /* styles                          */
  Pint               num_hatch_styles;    /* number of available hatch       */
                                          /* styles                          */
  Pint_list          hatch_styles;        /* list of available hatch styles  */
  Pint_list          shad_methods;        /* list of available shading       */
                                          /* methods                         */
  Pint               num_pred_int_inds;   /* number of predefined interior   */
                                          /* indices                         */
} Pint_facs_plus;


typedef struct {
  Pint                type;               /* line type                       */
  Pfloat              width;              /* linewidth scale factor          */
  Pgcolr              colr;               /* polyline colour                 */
  Pint                shad_method;        /* polyline shading method         */
  Pcurve_approx_crit  approx_crit;        /* curve approximation criteria    */
                                          /* data record                     */
} Pline_bundle_plus;
```

```
typedef struct {
  Pint              num_types;          /* number of available linetypes  */
  Pint_list         types;              /* list of line types             */
  Pint              num_widths;         /* number of available line widths*/
  Pfloat            nom_width;          /* nominal linewidth              */
  Pfloat            min_width;          /* minimum linewidth              */
  Pfloat            max_width;          /* maximum linewidth              */
  Pint_list         shad_methods;       /* list of available line shading */
                                        /* methods                        */
  Pint              num_inds;           /* number of predefined bundle    */
                                        /* indices                        */
} Pline_facs_plus;


typedef struct {
  Pint              type;               /* marker type                    */
  Pfloat            size;               /* marker size scale factor       */
  Pgcolr            colr;               /* polymarker colour              */
} Pmarker_bundle_plus;


typedef struct {
  Psurf_approx_crit     approx_crit;    /* surface approximation criteria */
                                        /* data                           */
  Pparam_surf_chars     chars_data;     /* parametric surface             */
                                        /* characteristics data           */
} Pparam_surf_rep;


typedef struct {
  Pint                  colr_type;      /* colour type                    */
  Pcolrv_array          colrs;          /* array of colour values         */
} Ppat_rep_plus;


typedef struct {
  Pint              refl_model;         /* reflectance model              */
  Prefl_prop        refl_prop;          /* reflectance property data      */
} Prefl_rep;


typedef struct {
  Pint              num_sets;           /* number of fill area sets       */
  Pint              num_data_per_facet; /* number of data values per facet*/
  Pcolrv_ptr        facet_colrvs;       /* list of facet colours          */
  Pvec3             *facet_norms;       /* list of facet normals          */
  Pfloat            *facet_data;        /* list of facet data lists       */
} Pset_of_fill_area_sets_data;
```

```
typedef struct {
  Pint                      u_order;       /* u spline order                  */
  Pint                      v_order;       /* v spline order                  */
  Pfloat_list               u_knots;       /* list of u knot values           */
  Pfloat_list               v_knots;       /* list of v knot values           */
  Pint                      colr_type;     /* colour type                     */
  Pcolr_ctrl_point_array    colrs;         /* array of colour control points */
} Psurf_colr_spline;


typedef struct {
  Pint                      u_order;       /* u spline order                  */
  Pint                      v_order;       /* v spline order                  */
  Pfloat_list               u_knots;       /* list of u knot values           */
  Pfloat_list               v_knots;       /* list of v knot values           */
  Pdata_ctrl_point_array    data_points;   /* array of data control           */
                                           /* points                          */
} Psurf_data_spline;


typedef struct {
  Pint                num_data_spline;     /* number of data splines          */
  Psurf_data_spline   *data_splines;       /* list of data splines            */
} Psurf_data_spline_list;


typedef struct {
  Pint                    u_order;         /* u spline order                  */
  Pint                    v_order;         /* v spline order                  */
  Pfloat_list             u_knots;         /* u knots                         */
  Pfloat_list             v_knots;         /* v knots                         */
  Pctrl_point3_array      ctrl_points;     /* array of control points         */
} Psurf_geom_spline;


typedef struct {
  Pint                font;                /* text font                       */
  Ptext_prec          precision;           /* text precision                  */
  Pfloat              char_expan;          /* character expansion factor      */
  Pfloat              char_space;          /* character spacing               */
  Pgcolr              colr;                /* text colour                     */
} Ptext_bundle_plus;
```

```
typedef struct {
  Pint                  num_sets;           /* number of triangle sets        */
  Pint                  num_data_per_facet; /* number of data values per      */
                                            /* facet                          */
  Pcolrv_ptr            facet_colrvs;       /* list of facet colours          */
  Pvec3                 *facet_norms;       /* list of facet normals          */
  Pfloat                *facet_data;        /* list of facet data lists       */
} Ptri_set_data;


typedef struct {
  Pcurve_approx_crit  approx_crit;          /* trimming curve approx.         */
                                            /* criteria data record           */
  Pvis_flag           visibility_flag;      /* curve visibility flag          */
  Pint                order;                /* curve order                    */
  Pfloat_list         knots;                /* curve knot vector              */
  Pfloat              low_limit;            /* curve parameter range lower    */
                                            /* limit                          */
  Pfloat              high_limit;           /* curve parameter range upper    */
                                            /* limit                          */
  Pctrl_point_list    ctrl_points;          /* curve control points           */
} Ptrim_curve;


typedef struct {
  Pint              num_trim_curves;        /* number of trimming curves      */
  Ptrim_curve       *trim_curves;           /* list of trimming curves forming*/
                                            /* a closed loop                  */
} Ptrim_curve_loop;


typedef struct {
  Pint                  num_trim_loops;     /* number of trimming curve loops */
  Ptrim_curve_loop      *trim_loops;        /* list of trimming curve loops   */
} Ptrim_curve_loop_list;


typedef struct {
  Pint_size         dims;                   /* dimensions of vertex array     */
  Pint              num_data_per_vertex;    /* number of data values          */
                                            /* per-vertex                     */
  Ppoint            *vertex_points;         /* array of vertex points         */
  Pcolrv_ptr        vertex_colrvs;          /* array of vertex colours        */
  Pvec3             *vertex_norms;          /* array of vertex normals        */
  Pfloat            *vertex_data;           /* array of vertex data lists     */
} Pvertex_array;


typedef Pint Pvertex_ind_triplet[3];
```

```
typedef struct {
  Pint        num_vertex_ind_triplets;     /* number of vertex index          */
                                           /* triplets                        */
  Pvertex_ind_triplets*vertex_ind_triplets;/* list of vertex index triplets   */
} Pvertex_ind_triplet_list;


typedef struct {
  Pint              num_vertices;          /* number of vertices              */
  Pint              num_data_per_vertex;   /* number of data values           */
                                           /* per-vertex                      */
  Ppoint            *vertex_points;        /* list of vertex points           */
  Pcolrv_ptr        vertex_colrvs;         /* list of vertex colours          */
  Pvec3             *vertex_norms;         /* list of vertex normals          */
  Pfloat            *vertex_data;          /* list of vertex data lists       */
} Pvertex_list;


typedef struct {
  Pint              num_lists;             /* number of vertex lists          */
  Pvertex_list      *vertices;             /* list of vertex lists            */
} Pvertex_set;


typedef struct {
  Pint_size         dims;                  /* dimensions of vertex array      */
  Pint              num_data_per_vertex;   /* number of data values           */
                                           /* per-vertex                      */
  Ppoint3           *vertex_points;        /* array of vertex points          */
  Pcolrv_ptr        vertex_colrvs;         /* array of vertex colours         */
  Pvec3             *vertex_norms;         /* array of vertex normals         */
  Pfloat            *vertex_data;          /* array of vertex data lists      */
} Pvertex3_array;


typedef struct {
  Pint              num_vertices;          /* number of vertices              */
  Pint              num_data_per_vertex;   /* number of data values           */
                                           /* per-vertex                      */
  Ppoint3           *vertex_points;        /* list of vertex points           */
  Pcolrv_ptr        vertex_colrvs;         /* list of vertex colours          */
  Pvec3             *vertex_norms;         /* list of vertex normals          */
  Pfloat            *vertex_data;          /* list of vertex data lists       */
} Pvertex3_list;


typedef struct {
  Pint              num_lists;             /* number of vertex lists          */
  Pvertex3_list     *vertices;             /* list of vertex lists            */
} Pvertex3_set;
```

```
typedef struct {
  Pint              data_map_rep;         /* max. # of data mapping table   */
                                          /* entries                        */
  Pint              refl_rep;             /* max. # of reflectance table    */
                                          /* entries                        */
  Pint              param_surf_rep;       /* max. # of param. surface table */
                                          /* entries                        */
  Pint              light_source_rep;     /* max. # of light source table   */
                                          /* entries                        */
  Pint              depth_cue_rep;        /* max. # of depth cue table      */
                                          /* entries                        */
  Pint              colr_map_rep;         /* max. # of colour mapping table */
                                          /* entries                        */
} Pws_st_tables_plus;
```

Clause A.2, modify the PHIGS `Pelem_data` *type definition by adding the specified PHIGS PLUS fields as defined below:*

```
typedef union {
  /* start of PHIGS element data */
  ...,
  /* end of PHIGS element data */
  /* start of PHIGS PLUS element data */
  struct Pelem_polyline_set3_colr {
    Pint              colr_type;              /* colour type                    */
    Pvertex3_set      vertex_data;            /* list of vertex lists           */
  } polyline_set3_colr;
  struct Pelem_fill_area_set3_data {
    Pint              colr_type;              /* colour type                    */
    Pfacet            *facet_data;            /* facet data                     */
    Pedge_flag_set    *edge_flags;            /* list of edge flag lists        */
    Pvertex3_set      vertex_data;            /* list of vertex lists           */
  } fill_area_set3_data;
  struct Pelem_fill_area_set_data {
    Pint              colr_type;              /* colour type                    */
    Pfacet            *facet_data;            /* facet data                     */
    Pedge_flag_set    *edge_flags;            /* list of edge flag lists        */
    Pvertex_set       vertex_data;            /* list of vertex lists           */
  } fill_area_set_data;
  struct Pelem_cell_array3_plus {
    Pparal            paral;                  /* parallelogram P, Q, R          */
    Ppat_rep_plus     pattern;                /* pattern colour array           */
  } cell_array3_plus;
  struct Pelem_set_of_fill_area_sets3_data {
    Pint                  colr_type;          /* colour type                    */
    Pset_of_fill_area_sets_data *set_data;    /* fill area sets data            */
    Pedge_flag_set_list   *edge_flags;        /* list of edge flag sets         */
    Pvertex3_list         vertex_data;        /* list of vertices               */
    Pcontour_set_list     contours;           /* list of contour sets           */
  } set_of_fill_area_sets3_data;
  struct Pelem_set_of_fill_area_sets_data {
    Pint                  colr_type;          /* colour type                    */
    Pset_of_fill_area_sets_data *set_data;    /* fill area sets data            */
    Pedge_flag_set_list   *edge_flags;        /* list of edge flag sets         */
    Pvertex_list          vertex_data;        /* list of vertices               */
    Pcontour_set_list     contours;           /* list of contour sets           */
  } set_of_fill_area_sets_data;
  struct Pelem_tri_set3_data {
    Pint                      colr_type;      /* colour type                    */
    Ptri_set_data             *set_data;      /* triangle set data              */
    Pedge_flag_triplet_list *edge_flags;      /* list of triangle edge flag     */
                                              /* triplets                       */
    Pvertex3_list             vertex_data;    /* list of vertices               */
    Pvertex_ind_triplet_list triplets;        /* list of triangle vertex index  */
                                              /* triplets                       */
```

```
} tri_set3_data;
struct Pelem_tri_set_data {
  Pint                    colr_type;     /* colour type                    */
  Ptri_set_data           *set_data;     /* triangle set data              */
  Pedge_flag_triplet_list *edge_flags;   /* list of triangle edge flag     */
                                         /* triplets                       */
  Pvertex_list            vertex_data;   /* list of vertices               */
  Pvertex_ind_triplet_list triplets;     /* list of triangle vertex index  */
                                         /* triplets                       */
} tri_set_data;
struct Pelem_tri_strip3_data {
  Pint                  colr_type;       /* colour type                    */
  Pfacet_list           *facet_data;     /* list of facet data             */
  Pedge_flag_list       *edge_flags;     /* list of edge flags             */
  Pvertex3_list         vertex_data;     /* list of vertex data            */
} tri_strip3_data;
struct Pelem_tri_strip_data {
  Pint                  colr_type;       /* colour type                    */
  Pfacet_list           *facet_data;     /* list of facet data             */
  Pedge_flag_list       *edge_flags;     /* list of edge flags             */
  Pvertex_list          vertex_data;     /* list of vertex data            */
} tri_strip_data;
struct Pelem_quad_mesh3_data {
  Pint                  colr_type;       /* colour type                    */
  Pfacet_array          *facet_data;     /* array of facet data            */
  Pedge_flag_array      *edge_flags;     /* array of edge flags            */
  Pvertex3_array        vertex_data;     /* array of vertex data           */
} quad_mesh3_data;
struct Pelem_quad_mesh_data {
  Pint                  colr_type;       /* colour type                    */
  Pfacet_array          *facet_data;     /* array of facet data            */
  Pedge_flag_array      *edge_flags;     /* array of edge flags            */
  Pvertex_array         vertex_data;     /* array of vertex data           */
} quad_mesh_data;
struct Pelem_non_uniform_b_spline_curve3 {
  Pcurve_geom_spline    geom_spline;     /* geometry spline                */
} non_uniform_b_spline_curve3;
struct Pelem_non_uniform_b_spline_curve3_colr {
  Pcurve_geom_spline    geom_spline;     /* geometry spline                */
  Pcurve_colr_spline    *colr_spline;    /* colour spline                  */
} non_uniform_b_spline_curve3_colr;
struct Pelem_non_uniform_b_spline_surf3 {
  Psurf_geom_spline     geom_spline;     /* geometry spline                */
  Ptrim_curve_loop_list *trim_loops;     /* list of trimming curve loops   */
} non_uniform_b_spline_surf3;
struct Pelem_non_uniform_b_spline_surf3_data {
  Psurf_geom_spline     geom_spline;     /* geometry spline                */
  Ptrim_curve_loop_list *trim_loops;     /* list of trimming curve loops   */
  Psurf_colr_spline     *colr_spline;    /* colour spline                  */
  Psurf_data_spline_list *data_spline;   /* data spline                    */
} non_uniform_b_spline_surf3_data;
```

```
  Pgcolr                colr;                /* general colour                */
  Pfacet_disting_mode   facet_disting_mode;  /* facet distinguishing mode     */
  Pfacet_cull_mode      facet_cull_mode;     /* facet culling mode            */
  Pdata_map_rep         data_map_method;     /* data mapping method           */
  Prefl_prop            refl_prop;           /* reflectance property data     */
  struct Pelem_light_source_st {
    Pint_list               act_list;        /* activation list               */
    Pint_list               deact_list;      /* deactivation list             */
  } light_source_st;
  Pcurve_approx_crit  curve_approx_crit;     /* curve approximation criteria  */
                                             /* data record                   */
  Psurf_approx_crit   surf_approx_crit;      /* surface approximation criteria*/
                                             /* data record                   */
  Pparam_surf_chars   param_surf_chars;      /* parametric surface            */
                                             /* characteristics data record   */
  /* end of PHIGS PLUS element data */
} Pelem_data;
```

A.3 External functions

Clause A.3, change the following PHIGS external functions:

```
/* INQUIRE GENERALIZED DRAWING PRIMITIVE 3 */
extern void pinq_gdp3 (
  Pint                  ws_type,             /* workstation type              */
  Pint                  gdp,                 /* GDP function identifier       */
  Pint                  *err_ind,            /* OUT error indicator           */
  Pint                  *num_attr,           /* OUT number of attributes used */
  Pattrs                attr[7]              /* OUT list of attributes used   */
);

/* INQUIRE GENERALIZED DRAWING PRIMITIVE */
extern void pinq_gdp (
  Pint                  ws_type,             /* workstation type              */
  Pint                  gdp,                 /* GDP function identifier       */
  Pint                  *err_ind,            /* OUT error indicator           */
  Pint                  *num_attr,           /* OUT number of attributes used */
  Pattrs                attr[7]              /* OUT list of attributes used   */
);
```

Clause A.3, add the following PHIGS PLUS external functions after PHIGS external functions:

```
/* POLYLINE SET 3 WITH COLOUR */
extern void ppolyline_set3_colr (
  Pint                  colr_type,           /* colour type                   */
  const Pvertex3_set    *vertex_data         /* list of vertex lists          */
);
```

```
/* FILL AREA SET 3 WITH DATA */
extern void pfill_area_set3_data (
  Pint                    colr_type,      /* colour type                    */
  const Pfacet            *facet_data,    /* facet data                     */
  const Pedge_flag_set    *edge_flags,    /* list of edge flag lists        */
  const Pvertex3_set      *vertex_data    /* list of vertex lists           */
);

/* FILL AREA SET WITH DATA */
extern void pfill_area_set_data (
  Pint                    colr_type,      /* colour type                    */
  const Pfacet            *facet_data,    /* facet data                     */
  const Pedge_flag_set    *edge_flags,    /* list of edge flag lists        */
  const Pvertex_set       *vertex_data    /* list of vertex lists           */
);

/* CELL ARRAY 3 PLUS */
extern void pcell_array3_plus (
  const Pparal            *paral,         /* parallelogram P, Q, R          */
  const Ppat_rep_plus     *pat            /* pattern colour array           */
);

/* SET OF FILL AREA SETS 3 WITH DATA */
extern void pset_of_fill_area_sets3_data (
  Pint                      colr_type,    /* colour type                    */
  const Pset_of_fill_area_sets_data *set_data,/*fill area set data          */
  const Pedge_flag_set_list  *edge_flags, /* list of edge flag sets         */
  const Pvertex3_list        *vertex_data,/* list of vertices               */
  const Pcontour_set_list    *contours    /* list of contour sets           */
);

/* SET OF FILL AREA SETS WITH DATA */
extern void pset_of_fill_area_sets_data (
  Pint                      colr_type,    /* colour type                    */
  const Pset_of_fill_area_sets_data  *set_data,/*fill area set data          */
  const Pedge_flag_set_list  *edge_flags, /* list of edge flag sets         */
  const Pvertex_list         *vertex_data,/* list of vertices               */
  const Pcontour_set_list    *contours    /* list of contour sets           */
);

/* TRIANGLE SET 3 WITH DATA */
extern void ptri_set3_data (
  Pint                     colr_type,     /* colour type                    */
  const Ptri_set_data      *set_data,     /* triangle set data              */
  const Pedge_flag_triplet_list *edge_flags,/*list of triangle edge flag     */
                                          /* triplets                       */
  const Pvertex3_list      *vertex_data,  /* list of vertices               */
  const Pvertex_ind_triplet_list *triplets /* list of triangle vertex index */
                                          /* triplets                       */
);
```