NAME

PEXNURBSurface - Non-Uniform Rational B-spline Surface Primitive

SYNTAX

void PEXNURBSurface(Display **display*, XID *resource_id*, PEXOCRequestType *req_type*, int *rationality*, int *uorder*, int *vorder*, float **uknots*, float **vknots*, unsigned int *col_count*, unsigned int *row_count*, PEXArrayOfCoord *points*, unsigned int *curve_count*, PEXListOfTrimCurve **trim_curves*)

PARAMETERS

display	A pointer to a display structure returned by a successful **XOpenDisplay** call.
resource_id	The resource identifier of the renderer or structure.
req_type	The request type for the output command (**PEXOCRender, PEXOCStore, PEXOCRenderSingle** or **PEXOCStoreSingle**).
rationality	The type of B-spline surface, whether rational or non-rational **PEXRational** or **PEXNonRational**).
uorder	The order of the polynomial expression in the u direction.
vorder	The order of the polynomial expression in the v direction.
uknots	An array of floats specifying the B-spline curve knots in the u direction.
vknots	An array of floats specifying the B-spline curve knots in the v direction.
col_count	The number of columns in the points array (number of points in the u direction).
row_count	The number of rows in the points array (number of points in the v direction).
points	An array of points defining the B-spline surface.
curve_count	The number of trimming curves.
trim_curves	A pointer to a list of trimming curves.

RETURNS

None

DESCRIPTION

This function creates a non-uniform B-spline surface output primitive.

The surface is generated as a function of the parametric variables u and v. The u and v order must be positive integers and indicate the order of the surface in each of the u and v parameter dimensions. **PEXGetImpDepConstants**(3) can be called to determine the largest supported value for surface u and v order. The spline shape is specified using two lists of knots in the parametric coordinate space, plus an array of control points specified in modeling coordinates. The u and v knot sequences must each form a non-decreasing sequence of numbers. The column count indicates the number of control points in the u direction and the row count indicates the number of control points in the v direction. The control points are stored in the array in row-major order (i.e., the column number varies fastest as vertices are stored in the array) and the rows increase in the direction of increasing v. The number of knots in the u direction is the sum of the order in the u direction and column count. The number of knots in the v direction is the sum of the order in the v direction and row count. The number of control points in each direction must be at least as large as the corresponding order.

The minimum and maximum knot values define the range over which the B-spline surface is evaluated in the u or v parametric direction,.

If the rationality is **PEXRational,** the control point list must be specified in homogeneous (4D) modeling coordinates. If rationality is **PEXNonRational,** the control point list must be specified in non-homogeneous (3D) modeling coordinates.

In addition to the parametric bounds, a list of trimming loops may also be specified. Trimming loops serve to further restrict the region in parametric coordinate space over which the surface is evaluated. Each trimming loop is defined as a list of one or more B-spline trimming curves that are connected head-to-tail. Each trim curve is a completely specified NURB curve, i.e. it is rational or non-rational, has its own order, etc. The list must be explicitly closed so that the tail of the last curve joins the head of the first. Each trimming curve is parameterized independently. If there is floating point inaccuracy in closure or in head-to-tail connectivity between curves, closure or connectivity will be assumed. Trimming loops are defined in the parameter space of the surface and may not go outside the parameter space of the surface.

When no trimming loops are specified, the rectangular parameter limits of the surface are renderer as the edges of the surface based on the edge flag attribute.

Trimming loops define the region of the surface that is to be rendered based on the following two rules: (1) a point is in the portion of the surface to be rendered if any ray projected from it to infinity has an odd number of intersections with trimming loops, and (2) traveling in the direction of a trimming loop, the portion of the surface to be trimmed away should be on the right and the portion to be retained should be on the left. In other words, a loop defined in counter-clockwise order will cause the interior of the loop to be retained and the exterior to be trimmed away. A clockwise loop will cause the exterior of the loop to be retained and the interior to be trimmed away. If loops are nested, they must alternate in direction. In all cases, the outermost loop must be counter-clockwise. No trimming curve may intersect itself and no trimming loop may intersect itself or any other trimming loop. Trimming loops that do not obey these rules will result in implementation-dependent behavior.

Each trimming curve has a visibility flag that controls its visibility for the purposes of surface edge display. Depending on the surface edge attributes and the visibility flags associated with trimming curves, the curves in trimming loops may be drawn as surface edges.

All attributes affecting the representation of fill area sets also affect the representation of the non-unform B-spline surface primitive. In addition, the surface approximation is used to determine how to approximate the B-spline surface and the parametric surface characteristics are used to specify the appearance of the surface.

If either of the specified surface orders are not supported, the output primitive is stored in a structure, but when rendered, the primitive is ignored and has no visual effect.

Trimming curve specification must abide by the constraints of NURB curve (e.g. number of control points at least as large as the order, non-decreasing knot sequence, order plus number of controls points equals the number of knots). Also, the trim curve order must not be less than two.

DATA STRUCTURES

```
typedef union {
        PEXCoord2D    *point_2d;
        PEXCoord      *point;
        PEXCoord4D    *point_4d;
} PEXArrayOfCoord;    /* Pointer to array of points */

typedef struct {
        unsigned short       count;          /* number of curves */
        PEXTrimCurve         *curves;
} PEXListOfTrimCurve;

typedef struct {
        PEXSwitch            visibility;         /* True or False */
        unsigned char        reserved;
        unsigned short       order;
        PEXCoordType         rationality;        /* PEXRational or PEXNonRational */
        PEXEnumTypeIndex     approx_method;      /* see PEXGetEnumTypeInfo */
        float                tolerance;
        float                tmin, tmax;
        PEXListOfFloat       knots;
        unsigned short       count;              /* number of control points */
        PEXArrayOfCoord      control_points;
} PEXTrimCurve;

typedef  unsigned char    PEXSwitch;
typedef  unsigned short   PEXCoordType;
typedef  short            PEXEnumTypeIndex;

typedef struct {
        unsigned short       count;          /* number of floats */
        float                *floats;
} PEXListOfFloat;
```

See also **PEXlib.h.**

ERRORS

BadPEXOutputCommand
The output command contains an invalid value.

BadPEXRenderer
The specified renderer resource identifier is invalid.

BadPEXStructure

The specified structure resource identifier is invalid.

SEE ALSO

PEXSetInteriorStyle(3)
PEXSetInteriorStyleIndex(3)
PEXSetSurfaceColorIndex(3)
PEXSetSurfaceColor(3)
PEXSetReflectionAttributes(3)
PEXSetReflectionModel(3)
PEXSetSurfaceInterpMethod(3)
PEXSetBFInteriorStyle(3)
PEXSetBFInteriorStyleIndex(3)
PEXSetBFSurfaceColorIndex(3)
PEXSetBFSurfaceColor(3)
PEXSetBFReflectionAttributes(3)
PEXSetBFReflectionModel(3)
PEXSetBFSurfaceInterpMethod(3)
PEXSetSurfaceApprox(3)
PEXSetFacetCullingMode(3)
PEXSetFacetDistinguishFlag(3)
PEXSetPatternSize(3)
PEXSetPatternAttributes(3)
PEXSetPatternAttributes2D(3)
PEXSetInteriorBundleIndex(3)
PEXSetSurfaceEdgeFlag(3)
PEXSetSurfaceEdgeType(3)
PEXSetSurfaceEdgeWidth(3)
PEXSetSurfaceEdgeColor(3)
PEXSetSurfaceEdgeColorIndex(3)
PEXSetEdgeBundleIndex(3)
PEXSetParaSurfCharacteristics(3)
PEXGetImpDepConstants(3)