Chapter 22. Functions bs3_surface Ma thru Rz

Topic: Ignore

bs3_surface_make_con

Function: Spline Interface, Construction Geometry

Action: Creates a parametric surface coincident with the given cone.

Prototype:

```
bs3_surface bs3_surface_make_con (
    cone const& con,                // given cone
    SPAbox const& region_of_interest,// region of
                                    // interest
    double  = 0,                    // required
                                    // positional
                                    // fit
    double& actual_fit              // return actual
        =*(double*)NULL_REF,        // fit used
    SPApar_transf& pt               // return param.
        =*(SPApar_transf*)NULL_REF  // space mapping
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”baseutil/vector/box.hxx”
#include ”baseutil/vector/param.hxx”
#include ”kernel/kerngeom/surface/condef.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
```

Description: Creates a parametric surface coincident with the given (possibly elliptical) cone over at least the portion inside the box, within the specified positional tolerance, and has its normal in the same sense.

If the actual_fit argument is supplied, it returns the actual tolerance achieved, or exact zero if this is better than system positional tolerance. The parameterization of the resulting surface need not match in any way that of the original cone.

Optionally, the mapping from the old parameter bounds to the new parameter bounds can be returned by supplying the pt argument.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_srf/sp3srtn.hxx

Effect: Changes model

bs3_surface_make_pipe

Function: Spline Interface, Construction Geometry

Action: Creates a spline surface that is an approximation to a constant–radius circular pipe centered on the given spine curve.

Prototype:

```
bs3_surface bs3_surface_make_pipe (
    double radius,           // signed radius
    curve const& spine,      // spine curve
    curve const& zero,       // curve giving zero
                             // u direction
    SPAinterval const& u_range,// angle range for
                             // cross–sections
    double requested_fit     // required fit tolerance
        = 0,
    double& actual_fit       // returns the actual
        =*(double*)NULL_REF  // fit tolerance used
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”baseutil/vector/interval.hxx”
#include ”kernel/kerngeom/curve/curdef.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
```

Description: The second curve gives the direction at each point of the zero cross–section parameter (the u–direction), and the interval gives the parameter (angle) range required in the cross direction. The u–parameter increases clockwise around the spine direction for a positive radius (convex surface) and counterclockwise for a negative radius (concave surface).

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_srf/sp3srtn.hxx

Effect: Changes model

bs3_surface_make_pipe_boundary

Function: Spline Interface, Construction Geometry

Action: Creates a pipe surface given the spine, radius, and two boundary curves that bracket the zero u–parameter direction.

Prototype:

```
bs3_surface bs3_surface_make_pipe_boundary (
    double radius,            // signed radius
    curve const& spine,       // spine curve
    curve const& lowu,        // boundary curve on the
                              // low u side
    curve const& highu,       // boundary curve on the
                              // high u side
    double requested_fit,     // requested fit
                              // tolerance
    double& actual_fit        // returns the actual
        =*(double*)NULL_REF,  // fit tolerance used
    bs2_curve& lowp           // parameter space curve
        =*(bs2_curve*)NULL_REF,  // for low u boundary
    bs2_curve& highp          // parameter space curve
        =*(bs2_curve*)NULL_REF   // for high u
                              // boundary
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
#include ”kernel/kerngeom/curve/curdef.hxx”
#include ”kernel/spline/bs2_crv/bs2curve.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
```

Description: Optionally, this routine creates parameter–space curves corresponding to the given boundary curves as well.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_srf/sp3srtn.hxx

Effect: Changes model

bs3_surface_make_pla

Function: Spline Interface, Construction Geometry

Action: Creates a spline surface from the given plane.

Prototype:

```
bs3_surface bs3_surface_make_pla (
    plane const& pla,                // given plane
    SPAbox const& region_of_interest,// region of
                                     // interest
    double  = 0,                     // required
                                     // positional fit
    double& actual_fit               // return actual
        =*(double*)NULL_REF,         // fit used
    SPApar_transf& pt                // return param.
        =*(SPApar_transf*)NULL_REF   // space mapping
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”baseutil/vector/box.hxx”
#include ”baseutil/vector/param.hxx”
#include ”kernel/kerngeom/surface/pladef.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
```

Description: Creates a parametric surface coincident with the given plane over at least the portion inside the box, within the specified positional tolerance, and has its normal in the same sense.

If the actual_fit argument is supplied, it returns the actual tolerance achieved, or exact zero if this is better than system positional tolerance. The parameterization of the resulting surface need not match in any way that of the original plane.

Optionally, the mapping from the old parameter bounds to the new parameter bounds can be returned by supplying the pt argument.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_srf/sp3srtn.hxx

Effect: Changes model

bs3_surface_make_sph

Function: Spline Interface, Construction Geometry

Action: Creates a spline surface from the given sphere.

Prototype:
```
bs3_surface bs3_surface_make_sph (
    sphere const& sph,              // given sphere
    SPAbox const& region_of_interest,// region of
                                    // interest
    double  = 0,                    // requested fit
                                    // tolerance
    double& actual_fit              // return actual
        =*(double*)NULL_REF,        // fit used
    SPApar_transf& pt               // return param.
        =*(SPApar_transf*)NULL_REF  // space mapping
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”baseutil/vector/box.hxx”
#include ”baseutil/vector/param.hxx”
#include ”kernel/kerngeom/surface/sphdef.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
```

Description: Creates a parametric surface coincident with the given sphere over at least the portion inside the box, within the specified positional tolerance, and has its normal in the same sense.

If the actual_fit argument is supplied, it returns the actual tolerance achieved, or exact zero if this is better than system positional tolerance. The parameterization of the resulting surface need not match in any way that of the original sphere.

Optionally, the mapping from the old parameter bounds to the new parameter bounds can be returned by supplying the pt argument.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_srf/sp3srtn.hxx

Effect: Changes model

bs3_surface_make_spl

Function: Spline Interface, Construction Geometry

Action: Creates a parametric surface coincident with the given spline.

Prototype:
```
bs3_surface bs3_surface_make_spl (
    spline const& spl,       // given spline
    SPAbox const&,           // region of interest
    double  = 0,             // requested fit
                             // tolerance
    double& actual_fit       // return actual
        =*(double*)NULL_REF, // fit used
    SPApar_transf&           // return parameter
        =*(SPApar_transf*)NULL_REF// space mapping
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”baseutil/vector/box.hxx”
#include ”baseutil/vector/param.hxx”
#include ”kernel/kerngeom/surface/spldef.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
```

Description: Creates a parametric surface coincident with the given spline over at least the portion inside the box, within the specified positional tolerance, and has its normal in the same sense.

If the actual_fit argument is supplied, it returns the actual tolerance achieved, or exact zero if this is better than system positional tolerance.

Optionally, the mapping from the old parameter bounds to the new parameter bounds can be returned by supplying the pt argument.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_srf/sp3srtn.hxx

Effect: Changes model

bs3_surface_make_sur

Function: Spline Interface, Construction Geometry

Action: Converts a portion of a general ACIS surface into a spline surface.

Prototype:

```
bs3_surface bs3_surface_make_sur (
    surface const& sur,              // given surface
    SPAbox const& region_of_interest,// region of
                                     // interest
    double requested_fit             // requested fit
        = 0,                         // tolerance
    double& actual_fit               // return actual
        =*(double*)NULL_REF,         // fit used
    SPApar_transf& pt                // return param.
        =*(SPApar_transf*)NULL_REF   // space mapping
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”baseutil/vector/box.hxx”
#include ”baseutil/vector/param.hxx”
#include ”kernel/kerngeom/surface/surdef.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
```

Description: Converts a general surface into a spline. There is no guarantee that the parameterization of the spline will have any obvious relationship to that of the original surface.

If the actual_fit argument is supplied, it returns the actual tolerance achieved, or exact zero if this is better than system positional tolerance. The parameterization of the resulting surface need not match in any way that of the original surface.

Optionally, the mapping from the old parameter bounds to the new parameter bounds can be returned by supplying the pt argument.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_srf/sp3srtn.hxx

Effect: Changes model

bs3_surface_make_tor

Function: Spline Interface, Construction Geometry

Action: Creates a spline surface from the given torus.

Prototype:
```
bs3_surface bs3_surface_make_tor (
    torus const& tor,               // given torus
    SPAbox const& region_of_interest,// region of
                                    // interest
    double  = 0,                    // requested fit
                                    // tolerance
    double& actual_fit              // return actual
        =*(double*)NULL_REF,        // fit used
    SPApar_transf& pt               // return param.
        =*(SPApar_transf*)NULL_REF  // space mapping
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”baseutil/vector/box.hxx”
#include ”baseutil/vector/param.hxx”
#include ”kernel/kerngeom/surface/tordef.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
```

Description: Creates a parametric surface coincident with the given torus over at least the portion inside the box, within the specified positional tolerance, and has its normal in the same sense.

If the actual_fit argument is supplied, it returns the actual tolerance achieved, or exact zero if this is better than system positional tolerance. The parameterization of the resulting surface need not match in any way that of the original torus.

Optionally, the mapping from the old parameter bounds to the new parameter bounds can be returned by supplying the pt argument.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_srf/sp3srtn.hxx

Effect: Changes model

bs3_surface_max_size_to_param_line

Function: Spline Interface, Construction Geometry

Action: Estimates the parameter of a near point on a surface, given the parameter of a control point of the surface.

Prototype:

```
double bs3_surface_max_size_to_param_line (
    bs3_surface bs3_surf,    // given surface
    const SPAposition& P,    // position
    double param_val,        // parameter value
    logical from_hi_edge,    // edge
    logical v_param_line     // v parameter line
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”baseutil/logical.h”
#include ”baseutil/vector/position.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/d3_bs3/spd3rtn.hxx”
```

Description: Returns an upper limit to the maximum distance from a point to that part of the bs3_surface whose vertices have parameter values between the given edge and the given value. If v_param_line is true, then the *u* values of the vertices' effective knot values are checked to make sure that they lie between the low/hi *u* edge (depending on whether from_hi is TRUE/FALSE) and the given (*u*) param value.

Errors: None

Limitations: This function will give poor estimates if the knot vector or control point distribution is skewed.

Library: kernel

Filename: kern/kernel/spline/d3_bs3/spd3rtn.hxx

Effect: Read–only

bs3_surface_mult_eku

Function: Spline Interface, Construction Geometry

Action: Determines if surface has multiple u end knots.

Prototype:

```
int bs3_surface_mult_eku (
    bs3_surface in_sur       // given surface
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/sg_bs3s/sps3srtn.hxx”
```

Description: If fully multiple end knots are at both the beginning and end of the surface, the function returns 1; otherwise returns 0. Linear is assumed to be multiple.

Errors: Returns 0 if the input is NULL.

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3s/sps3srtn.hxx

Effect: Read–only

bs3_surface_mult_ekv

Function: Spline Interface, Construction Geometry

Action: Determines if surface has multiple v end knots.

Prototype:
```
int bs3_surface_mult_ekv (
    bs3_surface in_sur       // given surface
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/sg_bs3s/sps3srtn.hxx”
```

Description: If fully multiple end knots are at both the beginning and end of the surface, the function returns 1; otherwise returns 0. Linear is assumed to be multiple.

Errors: Returns 0 if input is NULL.

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3s/sps3srtn.hxx

Effect: Read–only

bs3_surface_ncu

Function: Spline Interface, Construction Geometry

Action: Gets the number of control points in u_direction.

Prototype:
```
int bs3_surface_ncu (
    bs3_surface bs           // given surface
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/sg_bs3s/sps3srtn.hxx”
```

Description: Refer to Action.

Errors: Returns –1 if input is NULL.

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3s/sps3srtn.hxx

Effect: Read–only

bs3_surface_ncv

Function: Spline Interface, Construction Geometry

Action: Gets the number of control points in v_direction.

Prototype:

```
int bs3_surface_ncv (
    bs3_surface bs           // given surface
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/sg_bs3s/sps3srtn.hxx”
```

Description: Refer to Action.

Errors: Returns –1 if input is NULL.

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3s/sps3srtn.hxx

Effect: Read–only

bs3_surface_nku

Function: Spline Interface, Construction Geometry

Action: Gets the number of knots in the *u* direction for the given surface.

Prototype:

```
int bs3_surface_nku (
    bs3_surface bs          // input surface
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/sg_bs3s/sps3srtn.hxx”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3s/sps3srtn.hxx

Effect: Read–only

bs3_surface_nkv

Function: Spline Interface, Construction Geometry

Action: Gets the number of knots in the *v* direction for the given surface.

Prototype:

```
int bs3_surface_nkv (
    bs3_surface bs          // input surface
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/sg_bs3s/sps3srtn.hxx”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3s/sps3srtn.hxx

Effect: Read–only

bs3_surface_normal

Function: Spline Interface, Construction Geometry

Action: Evaluates the normal to a three–dimensional B–spline surface at a given (u,v).

Prototype:
```
SPAunit_vector bs3_surface_normal (
    SPApar_pos const& uv,    // given parameter point
    bs3_surface bs           // given surface
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”baseutil/vector/param.hxx”
#include ”baseutil/vector/unitvec.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
```

Description: This routine is often implemented as a call to bs3_surface_eval to obtain the first derivatives, returning the cross product, suitably normalized, but special action may be required at parameter singularities. At a surface singularity, like the apex of a cone, the routine returns a unit vector with all its components zero.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_srf/sp3srtn.hxx

Effect: Read–only

bs3_surface_nspans_u

Function: Spline Interface, Construction Geometry

Action: Gets number of simple patches in the u parameter direction forming a bs3_surface.

Prototype:
```
int bs3_surface_nspans_u (
    bs3_surface sur          // given surface
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
```

Description: If a spline surface consists of a rectangular array of simple patches, each rectangular in parameter space, returns the number of such patches in the u parameter direction.

What is considered a “simple” patch is open to interpretation, but with a piecewise bipolynomial surface (such as a B–spline), each bipolynomial piece would be a reasonable choice.

Errors:	None
Limitations:	None
Library:	kernel
Filename:	kern/kernel/spline/bs3_srf/sp3srtn.hxx
Effect:	Read–only

bs3_surface_nspans_v

Function: Spline Interface, Construction Geometry

Action: Gets number of simple patches in the v parameter direction forming a bs3_surface.

Prototype:

```
int bs3_surface_nspans_v (
    bs3_surface sur          // given surface
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
```

Description: If a spline surface consists of a rectangular array of simple patches, each rectangular in parameter space, returns the number of such patches in the v parameter direction.

Errors:	None
Limitations:	None
Library:	kernel
Filename:	kern/kernel/spline/bs3_srf/sp3srtn.hxx
Effect:	Read–only

bs3_surface_offset

Function: Spline Interface, Construction Geometry

Action: Offsets the given surface by offset distance.

Prototype:

```
bs3_surface bs3_surface_offset (
    const bs3_surface orig_surf,// given surface
    double offset_dist,         // distance to offset
    double offset_tol           // approximate
                                // tolerance
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/sg_bs3s/sps3srtn.hxx”
```

Description: It is assumed that the offset does not create a degenerate surface.

A +ve distance means to offset in the direction of the surface normal, and a –ve distance means to offset in direction opposite to the surface normal.

The offset surface constructed will be at least with in the offset_tol tolerance value (0.001 is a good value).

Errors: If an error occurs, a NULL surface is returned.

Limitations: If the surface has any flat–spots (zero length normals) then the offset–algorithm fails and a NULL surface is returned.

If a too low offset_tol is given, a potential data explosion may happen in the result.

Library: kernel

Filename: kern/kernel/spline/sg_bs3s/sps3srtn.hxx

Effect: Changes model

bs3_surface_periodic_u

Function: Spline Interface, Construction Geometry

Action: Determines whether the surface is periodic in the *u*-parameter.

Prototype:

```
logical bs3_surface_periodic_u (
    bs3_surface bs           // given surface
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”baseutil/logical.h”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
```

Description: This routine returns TRUE if the parameter line on the surface corresponding to minimum *u*-parameter is geometrically identical to that for maximum *u*-parameter, the parameterizations are the same, and the normals are continuous across the boundary.

If this routine returns TRUE for a surface, any routine that expects a *uv* parameter value must be prepared to accept any *u* value, and to map it into the principle range of the periodic surface (by adding or subtracting a multiple of the period) before evaluating the surface. Any routine that returns an actual *uv* value "near" to a given one must return the actual *u* value close to the given one, even if it is outside the standard surface parameter range.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_srf/sp3srtn.hxx

Effect: Read–only

bs3_surface_periodic_v

Function: Spline Interface, Construction Geometry

Action: Determines whether the surface is periodic in the *v*-parameter.

Prototype:

```
logical bs3_surface_periodic_v (
    bs3_surface bs          // given surface
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”baseutil/logical.h”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
```

Description: This routine returns TRUE if the parameter line on the surface corresponding to minimum *v*-parameter is geometrically identical to that for maximum *v*-parameter, the parameterizations are the same, and the normals are continuous across the boundary.

If this routine returns TRUE for a surface, any routine that expects a *uv* parameter value must be prepared to accept any *v* value, and to map it into the principle range of the periodic surface (by adding or subtracting a multiple of the period) before evaluating the surface. In addition, any routine that returns an actual *uv* value "near" to a given one must return the actual *v* value close to the given one, even if it is outside the standard surface parameter range.

Errors:	None
Limitations:	None
Library:	kernel
Filename:	kern/kernel/spline/bs3_srf/sp3srtn.hxx
Effect:	Read–only

bs3_surface_period_u

Function: Spline Interface, Construction Geometry

Action: Gets the *u*-parameter period of a three-dimensional B-spline surface.

Prototype:
```
double bs3_surface_period_u (
    bs3_surface bs            // given surface
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
```

Description: If the given surface is not periodic, the routine returns 0.

Errors:	None
Limitations:	None
Library:	kernel
Filename:	kern/kernel/spline/bs3_srf/sp3srtn.hxx
Effect:	Read–only

bs3_surface_period_v

Function: Spline Interface, Construction Geometry

Action: Gets the *v*-parameter period of a three-dimensional B-spline surface.

Prototype:
```
double bs3_surface_period_v (
    bs3_surface bs            // given surface
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
```

Description:	If the given surface is not periodic, the routine returns 0.
Errors:	None
Limitations:	None
Library:	kernel
Filename:	kern/kernel/spline/bs3_srf/sp3srtn.hxx
Effect:	Read–only

bs3_surface_perp

Function: Spline Interface, Construction Geometry

Action: Gets the intersection with the surface and the normal to the surface of a perpendicular dropped from a point to the surface.

Prototype:

```
void bs3_surface_perp (
    SPAposition const& point,// given point
    bs3_surface bs,          // given surface
    SPAposition& foot,       // returned foot of
                             // perpendicular
    SPAunit_vector& norm,    // returned normal
    SPApar_pos const& uv_guess// returned guess uv if
        =*(SPApar_pos*)NULL_REF,// known
    SPApar_pos& uv_actual    // returned actual uv
        =*(SPApar_pos*)NULL_REF// used
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”baseutil/vector/param.hxx”
#include ”baseutil/vector/position.hxx”
#include ”baseutil/vector/unitvec.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
```

Description: If an initial guess of the parameter values for the foot of the perpendicular is given, it is assumed to be close to the desired position, allowing faster processing.

For an open surface, there may be no perpendicular at all if the given point is beyond the boundary. In this case, a perpendicular will be dropped to a boundary edge, even though this is not perpendicular to the surface itself. If there is no perpendicular to the edges, then one of the corners will be returned as the foot. The normal direction will always be the normal to the surface at the foot.

Errors: Returns without setting any arguments if the input surface is NULL.

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_srf/sp3srtn.hxx

Effect: Read–only

bs3_surface_planar

Function: Spline Interface, Construction Geometry

Action: Determines if a bs3_surface is planar.

Prototype:
```
logical bs3_surface_planar (
    bs3_surface bs,          // surface to check
    SPAunit_vector& pln_norm// normal to the plane
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”baseutil/logical.h”
#include ”baseutil/vector/unitvec.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/sg_bs3s/sps3srtn.hxx”
```

Description: Returns TRUE if the surface is planar.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3s/sps3srtn.hxx

Effect: Read–only

bs3_surface_poles_u

Function: Spline Interface, Construction Geometry

Action: Gets a flag indicating the existence of poles (singularities) in the *u* direction.

Prototype:
```
int bs3_surface_poles_u (
    bs3_surface bs           // input surface
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/sg_bs3s/sps3srtn.hxx”
```

Description: Determines if the input surface has any poles in the *u* direction and returns an indicator flag:

0	Surface is not singular at either end
1	Surface is singular at *u*-start
2	Surface is singular at *u*-end
3	Surface is singular at *u*-start and *u*-end

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3s/sps3srtn.hxx

Effect: Read–only

bs3_surface_poles_v

Function: Spline Interface, Construction Geometry

Action: Gets a flag indicating the existence of poles (singularities) in the *v* direction.

Prototype:

```
int bs3_surface_poles_v (
    bs3_surface bs          // input surface
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/sg_bs3s/sps3srtn.hxx”
```

Description: Determines if the input surface has any poles in the *v* direction and returns an indicator flag:

0	Surface is not singular at either end
1	Surface is singular at *v*-start
2	Surface is singular at *v*-end
3	Surface is singular at *v*-start and *v*-end

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3s/sps3srtn.hxx

Effect: Read–only

bs3_surface_position

Function: Spline Interface, Construction Geometry

Action: Evaluates and returns a position on a three-dimensional B-spline surface at a given *uv*.

Prototype:
```
SPAposition bs3_surface_position (
    SPApar_pos const& uv,    // given parameter point
    bs3_surface bs           // given surface
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”baseutil/vector/param.hxx”
#include ”baseutil/vector/position.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_srf/sp3srtn.hxx

Effect: Read–only

bs3_surface_prin_curv

Function: Spline Interface, Construction Geometry

Action: Evaluates the principal axes of curvature and corresponding curvatures of a three-dimensional B-spline surface at a given *uv*.

Prototype:

```
void bs3_surface_prin_curv (
    SPApar_pos const& uv,    // given parameter point
    bs3_surface bs,          // given surface
    SPAunit_vector& u1,      // returned first
                             // principal axis
    double& c1,              // returned first
                             // curvature
    SPAunit_vector& u2,      // returned second
                             // principal axis
    double& c2               // returned second
                             // curvature
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”baseutil/vector/param.hxx”
#include ”baseutil/vector/unitvec.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
```

Description: The sign and order of the principal axes, are not significant. The sign of the curvature value indicates whether the surface is convex or concave with respect to the normal direction, which is considered to point out from the region bounded by the surface. A convex surface (one that curves back from the outward normal direction) has positive curvature, a concave one has negative curvature.

Errors: Returns without setting anything if the input surface is null.

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_srf/sp3srtn.hxx

Effect: Read–only

bs3_surface_proc

Function: Spline Interface, Construction Geometry

Action: Creates an approximate spline surface to a procedurally defined surface.

Prototype:

```
bs3_surface bs3_surface_proc (
    SPAinterval& u_interval,// parameter range in u
                            // over which approximate
                            // surface is constructed
    SPAinterval& v_interval,// parameter range in v
                            // over which approximate
                            // surface is constructed
    pt_eval_fn_t pt_fn,     // model space point
                            // evaluator
    vec_eval_fn_t du_fn,    // ds/du evaluator
    vec_eval_fn_t dv_fn,    // ds/dv evaluator
    vec_eval_fn_t duv_fn,   // ds2/dudv evaluator
    void* data,             // data to be passed to
                            // evaluators
    double res              // fit resolution
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "baseutil/vector/interval.hxx"
#include "kernel/spline/bs3_srf/bs3surf.hxx"
#include "kernel/spline/sg_bs3s/sps3srtn.hxx"
```

Description: The surface is defined over a parametric interval in *u* and *v*. Functions are supplied by the caller to calculate points on the surface, tangent vectors, and second derivatives. This information is evaluated and used to define a surface that is a fit to the evaluated data. If an error occurs, a NULL surface is returned.

The following are the type definitions of the evaluation functions used in bs3_surface_proc.

The pt_fn evaluator function for the surface must return a point on the surface at a given (*u*,*v*) value:

```
typedef logical
    (*pt_eval_fn_t) (       // return of FALSE =>
                            // evaluation failed
    double u,               // u param of evaluation
                            // point
    double v,               // v param of evaluation
                            // point
    void* data,             // data you passed to
                            // bs3_surface_proc
    SPAposition& pt         // OUT: model space
                            // position
                            // at given uv location
    );
```

The du_fn evaluator function for the surface must return a *u*–partial on the surface at a given (*u*,*v*) value. The dv_fn evaluator function for the surface must return a *v*–partial on the surface at a given (*u*,*v*) value. The duv_fn evaluator function for the surface must return a *uv*–partial on the surface at a given (*u*,*v*) value. All have the form:

```
typedef logical
    (*vec_eval_fn_t) (          // return of FALSE =>
                                // evaluation failed
    double u,                   // u param of evaluation
                                // point
    double v,                   // v param of evaluation
                                // point
    void* data,                 // data you passed to
                                // bs3_surface_proc
    SPAvector& vec);            // OUT: vector at given uv
                                // location
    );
```

Errors: None

Limitations: Do not give too small a value for res, because this can result in data explosion of the approximate surface. A res value of 0.01 seems to work best.

Library: kernel

Filename: kern/kernel/spline/sg_bs3s/sps3srtn.hxx

Effect: Changes model

bs3_surface_range

Function: Spline Interface, Construction Geometry

Action: Gets the range of both parameter values that defines the surface as the bounds of a rectangular box in parameter space.

Prototype:

```
SPApar_box bs3_surface_range (
    bs3_surface bs            // given surface
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”baseutil/vector/param.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
```

Description:	If the surface is periodic in one or both directions, it is defined for all parameter values in the periodic direction. This function returns a standard range over which the surface is traversed exactly once.
Errors:	None
Limitations:	None
Library:	kernel
Filename:	kern/kernel/spline/bs3_srf/sp3srtn.hxx
Effect:	Read–only

bs3_surface_range_u

Function: Spline Interface, Construction Geometry

Action: Gets the *u* parameter range a three-dimensional B-spline surface.

Prototype:
```
SPAinterval bs3_surface_range_u (
    bs3_surface bs          // given surface
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”baseutil/vector/interval.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
```

Description:	If the surface is periodic in the *u* parameter, it is deemed to be defined for all *u*, but this routine returns a standard range over which the surface is described exactly once. The length of this interval is the value returned by bs3_surface_period_u for such a surface.
Errors:	None
Limitations:	None
Library:	kernel
Filename:	kern/kernel/spline/bs3_srf/sp3srtn.hxx
Effect:	Read–only

bs3_surface_range_v

Function: Spline Interface, Construction Geometry

Action: Gets the *v* parameter range of a three-dimensional B-spline surface.

Prototype:

```
SPAinterval bs3_surface_range_v (
    bs3_surface bs          // given surface
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”baseutil/vector/interval.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
```

Description: If the surface is periodic in the *v* parameter, it is deemed to be defined for all *v*, but this routine returns a standard range over which the surface is described exactly once. The length of this interval is the value returned by bs3_surface_period_v for such a surface.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_srf/sp3srtn.hxx

Effect: Read–only

bs3_surface_rational_u

Function: Spline Interface, Construction Geometry

Action: Determines if a surface is rational in *u*.

Prototype:

```
logical bs3_surface_rational_u (
    bs3_surface bs          // given surface
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”baseutil/logical.h”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/sg_bs3s/sps3srtn.hxx”
```

Description: Returns TRUE if the surface is rational in u_direction logical, otherwise FALSE.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3s/sps3srtn.hxx

Effect: Read–only

bs3_surface_rational_v

Function: Spline Interface, Construction Geometry

Action: Determines if a surface is rational in *v*.

Prototype:
```
logical bs3_surface_rational_v (
    bs3_surface bs           // given surface
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”baseutil/logical.h”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/sg_bs3s/sps3srtn.hxx”
```

Description: Returns TRUE if the surface is rational in v_direction logical, otherwise FALSE.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3s/sps3srtn.hxx

Effect: Read–only

bs3_surface_remove_extra_knots

Function: Spline Interface, Construction Geometry

Action: Deletes knots where multiplicities are greater than the degree in both *u* and *v*.

Prototype:
```
void bs3_surface_remove_extra_knots (
    bs3_surface bs,          // given surface
    double knot_tol          // knot tolerance
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
```

Description: This function searches for knots multiplicities greater than the degree in both the *u* and *v* directions. The appropriate knots and control points are removed from the data structure.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_srf/sp3srtn.hxx

Effect: System routine

bs3_surface_reparam_u

Function: Spline Interface, Construction Geometry

Action: Reparameterizes the surface in *u*.

Prototype:
```
void bs3_surface_reparam_u (
    double start,            // start u parameter
                             // desired
    double end,              // end u parameter
                             // desired
    bs3_surface bs           // given surface
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
```

Description: Reparameterizes the given surface in place using a linear transformation in the *u* direction so that its primary interval of definition in the *u* direction is from the start to the end parameters given (which must be in increasing order).

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_srf/sp3srtn.hxx

Effect: Changes model

bs3_surface_reparam_v

Function: Spline Interface, Construction Geometry

Action: Reparameterizes the surface in *v*.

Prototype:

```
void bs3_surface_reparam_v (
    double start,           // start v parameter
                            // desired
    double end,             // end v parameter
                            // desired
    bs3_surface bs          // given surface
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
```

Description: Reparameterizes a B-spline surface using linear transformation in the *v*-direction, to achieve given start and end values. The new parameter value is the appropriate linear function of the old.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_srf/sp3srtn.hxx

Effect: Changes model

bs3_surface_restore

Function: Spline Interface, Construction Geometry, SAT Save and Restore

Action: Restores a saved surface.

Prototype:

```
bs3_surface bs3_surface_restore ();
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
```

Description: Reads back a representation of a parametric surface as written by bs3_surface_save, and creates a duplicate of the original surface.

The overloaded >> operator behaves like bs3_surface_restore, except that it reads from a C++ style stream using stream operators, and sets the result into the second argument.

For example: bs3_surface surf;

Reading uses routines read_int, read_long, read_real, and read_string that are defined in kernutil/fileio/fileio.hxx.

```
if (restore_version_number < SPLINE_VERSION)
    if (read_int() == –1)
        // First check that there is a surface to read.
    read_int                        stype
    read_int                        save_dim
    read_int                        u degree
    read_int                        v degree
    read_int                        save nu span
    read_int                        save nv span
    read_int                        rat u
    read_int                        rat v
    read_int                        form u
    read_int                        form v
    read_int                        pole u
    read_int                        pole v
else
    // New style header. There are keywords instead of numbers
    // where appropriate, and redundant values are missing.
    read_id                         id string
    if (strcmp( id_string, type_nullbs ) == 0)
        // return NULL;
    else if (strcmp( id_string, type_nubs ) == 0 )
        // rational = FALSE;
    else if (strcmp( id_string, type_nurbs ) == 0 )
        // rational = TRUE;
    else
        // sys_error( UNKNOWN_BS_SURFACE );
    read_int                        u degree
    read_int                        v degree
    if (rational)
        read_id                     id string for rational_u or
                                    rational_v
    if (restore_version_number < CONSISTENT_VERSION)
        read_id                     id string for formu
        read_id                     id string for formv
        read_id                     id string for poleu
        read_id                     id string for polev
    else
        read_enum                   Read enumeration bs3_surf_form
                                    for form_map for form u
        read_enum                   Read enumeration bs3_surf_form
                                    for form_map for form v
        read_enum                   Read enumeration sing_map for
```

```
                                        pole u
          read_enum                     Read enumeration sing_map for
                                        pole v
// Read the knots and multiplicities, allocating space for
// the knot values as we go, and accumulating the total of
// knots and multiplicities.
read_int                                Number of knots in u
if (restore_version_number >= SPLINE_VERSION)
     read_int                           Number of knots in v
for (int i = 0; i < n_uknots; i++)
     read_real                          u knot
     read_int                           u multiplicity
if (restore_version_number < SPLINE_VERSION)
     read_int                           Number of knots in v
for (i = 0; i < n_vknots; i++)
     read_real                          v knot
     read_int                           v multiplicity
// Finally read the control point values.
for (row_start = bs–>node0;
     row_start != NULL;
     row_start = row_start–>vnext)
     for (ag_snode *this_node = row_start;
          this_node != NULL;
          this_node = this_node–>unext)
          for (i = 0; i < dimh; i++)
               read_real                node Pw weight
```

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_srf/sp3srtn.hxx

Effect: System routine

bs3_surface_revolve_curve

Function: Spline Interface, Construction Geometry

Action: Creates a spline surface by revolving a three-dimensional B-spline curve about an axis.

Prototype:
```
bs3_surface bs3_surface_revolve_curve (
    bs3_curve gen,           // generator curve
    straight const& axis,    // axis
    double start_ang         // start angle
        = 0.0,
    double stop_ang          // stop angle
        = 0.0,
    double                   // requested fit
         = 0,                // tolerance
    double& actual_fit       // returned actual fit
        =*(double*)NULL_REF  // tolerance used
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kerngeom/curve/strdef.hxx”
#include ”kernel/spline/bs3_crv/bs3curve.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/bs3_srf/sp3srtn.hxx”
```

Description: The curve need not be planar. The generating curve defines the zero angular position, from that the start and stop angles are calculated, clockwise around the given axis. If the angles are equal, the curve is swept around a full circle. The result will match the true surface of revolution within the specified positional precision. If the actual precision achieved is better than system positional accuracy (for example if the surface package supports rational quadratic polynomial surfaces) then the actual tolerance will be returned as exactly zero.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_srf/sp3srtn.hxx

Effect: Changes model

bs3_surface_ruled

Function: Spline Interface, Construction Geometry

Action: Creates a ruled surface between two curves.

Prototype:
```
bs3_surface bs3_surface_ruled (
    const bs3_curve& crv1,   // first curve
    const bs3_curve& crv2    // second curve
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_crv/bs3curve.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/sg_bs3s/sps3srtn.hxx”
```

Description: This function can accept incompatible bs3_curves, meaning their knot vectors, rationality, or degree can be different.

Errors: If an error occurs, a NULL surface is returned.

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3s/sps3srtn.hxx

Effect: Changes model

bs3_surface_rule_from_pt

Function: Spline Interface, Construction Geometry

Action: Creates a triangular ruled surface from a point to a curve.

Prototype:

```
bs3_surface bs3_surface_rule_from_pt (
    const SPAposition& p,    // point on boundary of
                             // surface
    const bs3_curve crv      // boundary curve
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”baseutil/vector/position.hxx”
#include ”kernel/spline/bs3_crv/bs3curve.hxx”
#include ”kernel/spline/bs3_srf/bs3surf.hxx”
#include ”kernel/spline/sg_bs3s/sps3srtn.hxx”
```

Description: Refer to Action.

Errors: If an error occurs, a NULL surface is returned.

Limitations: The point must not be on the curve.

Library: kernel

Filename: kern/kernel/spline/sg_bs3s/sps3srtn.hxx

Effect: Changes model