Chapter 18.

Functions bs3_curve Aa thru Lz

Topic: Ignore

bs3_curve_accurate_derivs

Function: Spline Interface, Construction Geometry

Action: Gets the number of derivatives that bs3_curve_evaluate can calculate.

Prototype:
```
int bs3_curve_accurate_derivs (
    bs3_curve               // given curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/bs3_crv/sp3crtn.hxx"
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_crv/sp3crtn.hxx

Effect: Read–only

bs3_curve_add_knot

Function: Spline Interface, Construction Geometry, Construction Geometry

Action: Adds a knot to a B-spline at a given parameter value.

Prototype:
```
int bs3_curve_add_knot (
    bs3_curve input,        // given curve, modified
                            // in place
    double new_par,         // returned new knot
                            // value
    int mult_req,           // returned new knot's
                            // multiplicity
    double knot_tol         // returned knot
                            // tolerance
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/sg_bs3c/sps3crtn.hxx"
```

Description: The routine returns the number of knots added. If the knot value to be added is an existing knot, the knot is inserted, provided the multiplicity of the current knot does not exceed the degree of the spline curve. The equality of the knots are tested using the tolerance given as input. The knot value must be within the parameter range of the input B-spline.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3c/sps3crtn.hxx

Effect: Changes model

bs3_curve_add_mult_ekn

Function: Spline Interface, Construction Geometry, Construction Geometry

Action: Adds multiple end knots to a spline curve.

Prototype:
```
void bs3_curve_add_mult_ekn (
    bs3_curve input         // given curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/sg_bs3c/sps3crtn.hxx"
```

Description: This routine is always called for periodic curves such that the B-spline control polygon always interpolates the start and end of the curve.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3c/sps3crtn.hxx

Effect: Changes model

bs3_curve_arc_3curve

Function: Spline Interface, Construction Geometry

Action: Gets the radius and center of an arc tangent to three curves.

Prototype:

```
int bs3_curve_arc_3curve (
    bs3_curve crv1,                 // first curve
    double& t1,                     // parameter of
                                    // tangency on first
                                    // curve
    bs3_curve crv2,                 // second curve
    double& t2,                     // parameter of
                                    // tangency on second
                                    // curve
    bs3_curve crv3,                 // third curve
    double& t3,                     // parameter of
                                    // tangency on third
                                    // curve
    const SPAunit_vector& normal,// normal to plane
                                    // of all curves
    double& radius,                 // returned arc
                                    // radius
    SPAposition& center             // returned arc
                                    // center
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "baseutil/vector/position.hxx"
#include "baseutil/vector/unitvec.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/sg_bs3c/sps3crtn.hxx"
```

Description: All three B-splines must lie in the same plane. There are potentially many circles lying tangent to the three curves. t1, t2, and t3 provide starting guesses for the parameter value at the point of tangency. Upon successful completion, each contains the actual parameter value at the point of tangency. This function returns an error code, which is 0 if there is no error.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3c/sps3crtn.hxx

Effect: Read–only

bs3_curve_arc_3curve_modified

Function: Spline Interface, Construction Geometry

Action: Gets the radius and center of an arc tangent to three curves.

Prototype:
```
int bs3_curve_arc_3curve_modified (
    bs3_curve crvs[3],      // array of bs3 curves
    SPAposition pos[3],     // array of positions
    double ts[3],           // array of tangency
                            // parameters
    const SPAunit_vector& normal,// normal to plane
                            // of all curves
    double& radius,         // radius
    SPAposition& center     // center
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/position.hxx"
#include "baseutil/vector/unitvec.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/sg_bs3c/sps3crtn.hxx"
```

Description: Gets the radius and center of an arc tangent to three curves. One or two of the curves may degenerate to a point. If a curve degenerates to a point, this is signified by a NULL curve pointer and a non–NULL position entry. It returns an error code which is 0 if no error.

All three B–splines must lie in the same plane. There are potentially many circles lying tangent to the three curves. t1, t2, and t3 provide starting guesses for the parameter value at the point of tangency. Upon successful completion, each contains the actual parameter value at the point of tangency. This function returns an error code, which is 0 if there is no error.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3c/sps3crtn.hxx

Effect: Changes model

bs3_curve_box

Function: Spline Interface, Construction Geometry

Action: Determines a box that encloses the curve with additional tolerances.

Prototype:
```
SPAbox bs3_curve_box (
    bs3_curve cur,          // given curve
    double fitol            // given fit tolerance
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/box.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/bs3_crv/sp3crtn.hxx"
```

Description: Returns a box (presently a rectangular parallelepiped parallel to object-space axes) that completely contains the curve, with an additional allowance of the given tolerance all round. The box will not be the smallest possible, but will be a compromise between a tight fit and fast evaluation.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_crv/sp3crtn.hxx

Effect: Changes model

bs3_curve_bs3_curve_int

Function: Spline Interface, Construction Geometry

Action: Intersects two curves.

Prototype:
```
curve_curve_int* bs3_curve_bs3_curve_int (
    bs3_curve cur1,         // first given curve
    bs3_curve cur2,         // second given curve
    double tol              // given fit tolerance
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernint/intcucu/intcucu.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/bs3_crv/sp3crtn.hxx"
```

Description: If the curves pass within the tolerance of each other, an intersection is produced. Intersections are returned in an undefined order.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_crv/sp3crtn.hxx

Effect: Changes model

bs3_curve_check

Function: Spline Interface, Construction Geometry

Action: Checks the curve for closure, continuity etc. and applies fixes if requested, and possible.

Prototype:
```
check_status_list* bs3_curve_check (
    bs3_curve bs3,              // given curve
    const intcurve& ic          // corresponding
        =*(intcurve*)NULL_REF,  // intcurve
    const                       // list of things
    check_status_list* check    // to be checked
        = NULL
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kerngeom/curve/intdef.hxx"
#include "kernel/kernint/d3_chk/chk_stat.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/d3_bs3/spd3rtn.hxx"
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: kernel

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

Effect: Changes model

bs3_curve_check_smoothness

Function: Spline Interface, Construction Geometry

Action: Gets knot values where the curve is discontinuous in tangent direction or magnitude.

Prototype:
```
void bs3_curve_check_smoothness (
    bs3_curve bs,                // given B-spline
                                 // curve
    curve_irregularities*& cirr,// returned list of
                                 // C1/G1
                                 // discontinuities
    int& no_pts,                 // returned number of
                                 // points in list
    int continuity               // returned
        = 1                      // continuity, 1 =
                                 // tangent (default),
                                 // 2 = acceleration
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_crv/bs3curve.hxx”
#include ”kernel/spline/sg_bs3c/bs3ccont.hxx”
#include ”kernel/spline/sg_bs3c/sps3crtn.hxx”
```

Description: Returns a linked list that indicates knot values where the curve is discontinuous in tangent direction or magnitude. For C1/G1 continuity checking.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3c/sps3crtn.hxx

Effect: Read–only

bs3_curve_closed

Function: Spline Interface, Construction Geometry

Action: Determines if a curve is closed.

Prototype:
```
logical bs3_curve_closed (
    bs3_curve cur           // given curve
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”baseutil/logical.h”
#include ”kernel/spline/bs3_crv/bs3curve.hxx”
#include ”kernel/spline/bs3_crv/sp3crtn.hxx”
```

Description: Returns TRUE if the curve is closed, i.e., the two end points are within system tolerance of each other in object space.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_crv/sp3crtn.hxx

Effect: Read–only

bs3_curve_compat

Function: Spline Interface, Construction Geometry

Action: Sets two curves to be side–by–side compatible.

Prototype:
```
void bs3_curve_compat (
    bs3_curve b1,           // first given curve
    bs3_curve b2            // second given curve
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_crv/bs3curve.hxx”
#include ”kernel/spline/sg_bs3c/sps3crtn.hxx”
```

Description: This is not for making curves compatible to be connected end–to–end, but to make them compatible side–by–side, so that, for example, they could be used in the same surface, in the same direction. The resulting curves have the same degree, rationality, and knot vector.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3c/sps3crtn.hxx

Effect: Changes model

bs3_curve_conic_type

Function: Spline Interface

Action: Determines whether the given curve is a conic section.

Prototype:
```
bs_conic_type bs3_curve_conic_type (
    bs3_curve bs3c          // given curve
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_crv/bs3curve.hxx”
#include ”kernel/spline/sg_bs3c/sps3crtn.hxx”
```

Description: This API returns a value of the enumerated type bs_conic_type, that indicates what type of conic the curve is (ellipse, circle, parabola, hyperbola, or a line or polyline), or that it is not a conic section.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3c/sps3crtn.hxx

Effect: Read–only

bs3_curve_connect

Function: Spline Interface, Construction Geometry

Action: Joins two splines end to end in a C1 join.

Prototype:
```
bs3_curve bs3_curve_connect (
    bs3_curve bs1,          // first given curve
    bs3_curve bs2           // second given curve
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_crv/bs3curve.hxx”
#include ”kernel/spline/bs3_crv/sp3crtn.hxx”
```

Description: This routine takes care of compatibility of the curves, and of cleaning up. first_part and last_part are gone after the call, and the resulting spline is the return value. However, it’s dangerous to say

```
s1 = bs3_curve_connect ( s1, s2 );
```

because if it fails, it returns a NULL pointer and leaves s1 and s2 alone. Thus s1 still exists but the caller has just zeroed its pointer. So always say

```
s3 = bs3_curve_connect ( s1, s2 );
```

then either s3 is valid and s1 and s2 are gone, or vice versa.

The resulting spline has the same direction and parameterization as the first spline, s1. The second one might be reversed, and will likely be reparameterized. It gets attached to either the start or end of the first spline, wherever it is coincident. Its parameterization is adjusted so that its tangent magnitude matches that of the first spline, where they meet. This gives a C1 join, if the input splines were G1.

Errors: If there's a problem, the return value is NULL and the inputs are unchanged. The only problems might be if the splines have no common end points, or if either spline is null or corrupted somehow.

Limitations: Both splines must have the normal knot multiplicity of knots at the ends. This is not generalized to do both bs2_curves and bs3_curves; the dimension is 3.

Library: kernel

Filename: kern/kernel/spline/bs3_crv/sp3crtn.hxx

Effect: Changes model

bs3_curve_construct

Function: Spline Interface, Construction Geometry

Action: Creates a cubic curve which is supplied as B-spline vertexes and knot values.

Prototype:
```
bs3_curve bs3_curve_construct (
    int nkts,                    // number of knots
    const SPAposition* vertices,// vertices
    double* knots,               // list of knots
    int mult                     // multiplicity
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/position.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/d3_bs3/spd3rtn.hxx"
```

Description: The number of knots is given, and the multiplicities of internal knots (the end knots have multiplicity 3).

vertices is an array containing the three or two space vertices. It should contain mult*(nkts–2)+4 values.

knots is an array of nkts distinct knot values.

mult is the multiplicity of the internal knots. The end knots always get a multiplicity of three.

The form of the curve must be set after construction. The form of the underlying spline curve is set during the construction.

Errors: None

Limitations: None

Library: kernel

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

Effect: Changes model

bs3_curve_control_points

Function: Spline Interface, Construction Geometry

Action: Gets the number of control points and an array of control points for a 3D B–spline curve.

Prototype:
```
void bs3_curve_control_points (
    bs3_curve bs,           // input curve
    int& num_pts,           // number of control
                            // points output
    SPAposition*& ctrlpts   // output control point
                            // array
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/position.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/sg_bs3c/sps3crtn.hxx"
```

Description: This function creates an array of control points. It is up to the application to delete this array.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3c/sps3crtn.hxx

Effect: Read–only

bs3_curve_copy

Function: Spline Interface, Construction Geometry

Action: Creates an exact copy of the curve in free store.

Prototype:
```
bs3_curve bs3_curve_copy (
    bs3_curve cur           // given curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/bs3_crv/sp3crtn.hxx"
```

Description: ACIS calls this routine only when a change is to be made to one copy of the curve, so there is no advantage to be gained by deferring the duplication further. Ordinary duplication of ACIS intersection curves merely creates a new reference to the same underlying bs3_curve.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_crv/sp3crtn.hxx

Effect: System routine

bs3_curve_cub_fit_to_conic

Function: Spline Interface, Construction Geometry

Action: Creates a nonrational cubic B-spline fit to the given conic.

Prototype:
```
bs3_curve bs3_curve_cub_fit_to_conic (
    bs3_curve input,        // given curve
    double fit_tol          // fit tolerance
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/sg_bs3c/sps3crtn.hxx"
```

Description: The routine does not check to determine whether the given input spline is other than conic. If this routine is tried for any general rational, the routine works but the spline will be inaccurate; i.e., it is not guaranteed to be within the fit tolerance given as input. If an error occurs, this routine returns a NULL curve.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3c/sps3crtn.hxx

Effect: Changes model

bs3_curve_curvature

Function: Spline Interface, Construction Geometry

Action: Evaluates the curvature of the curve at the given parameter value.

Prototype:
```
SPAvector bs3_curve_curvature (
    double param,           // given parameter value
    bs3_curve cur           // given curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/vector.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/bs3_crv/sp3crtn.hxx"
```

Description: The direction of the vector is from the point on the curve towards the center of curvature, and the magnitude is the curvature. This is not a true Euclidean vector, because it does not transform correctly under scaling.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_crv/sp3crtn.hxx

Effect: Read–only

bs3_curve_cylinder

Function: Spline Interface, Construction Geometry

Action: Gets a cylinder enclosing a bs3_curve.

Prototype:
```
void bs3_curve_cylinder (
    bs3_curve cu,           // given curve
    SPAposition& root,      // returned cylinder root
                            // point
    SPAunit_vector& axis,   // returned cylinder axis
    double& rad             // returned cylinder
radius
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/position.hxx"
#include "baseutil/vector/unitvec.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/d3_bs3/spd3rtn.hxx"
```

Description: The cylinder axis is parallel to the line between its end points, and its radius is the minimum required to enclose all the B-spline control points.

The method is:

First, set the axis to the line from *P1* to *Pn* (these being the first and last vertexes in the B-spline hull).

Second, find the vertex furthest from this line. Move the axis parallel to itself, halfway to this vertex.

Third, find the vertex furthest from the new axis. Use the distance to this vertex as the cylinder radius.

Errors: None

Limitations: None

Library: kernel

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

Effect: Changes model

bs3_curve_debug

Function: Spline Interface, Construction Geometry, Debugging

Action: Gets a readable representation of the curve and writes it to a file.

Prototype:
```
void bs3_curve_debug (
    bs3_curve cur,          // given curve
    char const* leader,     // leader string
    FILE* fp                // output file
        = debug_file_ptr
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/bs3_crv/sp3crtn.hxx"
```

Description: If there is more than one text line (as is almost certain), all lines but the first will start with the leader string. Do not use a newline to terminate the last line.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_crv/sp3crtn.hxx

Effect: System routine

bs3_curve_degree

Function: Spline Interface, Construction Geometry

Action: Gets the degree of a bs3_curve.

Prototype:
```
int bs3_curve_degree (
    bs3_curve bs            // bs3_curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/sg_bs3c/sps3crtn.hxx"
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3c/sps3crtn.hxx

Effect: Read–only

bs3_curve_degree_elevate

Function: Spline Interface, Construction Geometry

Action: Raises the degree of a B-spline curve in place by one.

Prototype:
```
void bs3_curve_degree_elevate (
    bs3_curve& input        // given curve, modified
                            // in place
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/sg_bs3c/sps3crtn.hxx"
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3c/sps3crtn.hxx

Effect: Changes model

bs3_curve_delete

Function: Spline Interface, Construction Geometry

Action: Deletes storage occupied by the given curve that is no longer required.

Prototype:
```
void bs3_curve_delete (
    bs3_curve& cur          // given curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/bs3_crv/sp3crtn.hxx"
```

Description: ACIS makes no assumptions about how the underlying surface package manages its storage space if it doesn't incapacitate the standard C memory allocation mechanism.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_crv/sp3crtn.hxx

Effect: System routine

bs3_curve_deriv

Function: Spline Interface, Construction Geometry

Action: Evaluates the parametric derivative (direction and magnitude) of a given 3D B–spline curve at a given parameter value.

Prototype:
```
SPAvector bs3_curve_deriv (
    double param,           // given parameter value
    bs3_curve cur           // given curve
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”baseutil/vector/vector.hxx”
#include ”kernel/spline/bs3_crv/bs3curve.hxx”
#include ”kernel/spline/bs3_crv/sp3crtn.hxx”
```

Description: Normally, this is implemented as a call to bs3_curve_eval.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_crv/sp3crtn.hxx

Effect: Read–only

bs3_curve_end

Function: Spline Interface, Construction Geometry

Action: Gets the end point of the given spline curve.

Prototype:
```
SPAposition bs3_curve_end (
    bs3_curve bs            // given curve
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”baseutil/vector/position.hxx”
#include ”kernel/spline/bs3_crv/bs3curve.hxx”
#include ”kernel/spline/sg_bs3c/sps3crtn.hxx”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3c/sps3crtn.hxx

Effect: Read–only

bs3_curve_end_tangent

Function: Spline Interface, Construction Geometry

Action: Gets the normalized tangent to the given spline at the end.

Prototype:
```
SPAunit_vector bs3_curve_end_tangent (
    bs3_curve bs            // given curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/unitvec.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/sg_bs3c/sps3crtn.hxx"
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3c/sps3crtn.hxx

Effect: Read–only

bs3_curve_estimate_param

Function: Spline Interface, Construction Geometry

Action: Estimates the parameter of the nearest point or a curve to the given point.

Prototype:
```
double bs3_curve_estimate_param (
    const SPAposition& P,   // given position
                            // on curve
    bs3_curve cu            // given curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/position.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/d3_bs3/spd3rtn.hxx"
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: kernel

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

Effect: Read–only

bs3_curve_eval

Function: Spline Interface, Construction Geometry

Action: Evaluates the curve and its parametric derivatives at the given parameter value.

Prototype:
```
void bs3_curve_eval (
    double param,           // given parameter
    bs3_curve cur,          // given curve
    SPAposition& x,         // returned position
    SPAvector& xdot         // returned first
        =*(SPAvector*)NULL_REF,// derivative
    SPAvector& xdotdot      // returned second
        =*(SPAvector*)NULL_REF// derivative
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/position.hxx"
#include "baseutil/vector/vector.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/bs3_crv/sp3crtn.hxx"
```

Description: The number of derivatives evaluated depends upon the last two arguments. If the last argument is not NULL, two derivatives will be evaluated. If the last argument is NULL and the next–to–last argument is not NULL, one derivative will be calculated. If both are NULL, no derivatives will be calculated.

Errors: None

Limitations: This routine ignores the possibility of discontinuities. At a discontinuity, values will be returned for either the left or the right side, with no current guarantee of which.

Library: kernel

Filename: kern/kernel/spline/bs3_crv/sp3crtn.hxx

Effect: Read–only

bs3_curve_evaluate

Function: Spline Interface, Construction Geometry

Action: Gets an arbitrary number of derivatives of a curve.

Prototype:
```
int bs3_curve_evaluate (
    double param,           // given parameter
    bs3_curve cur,          // given curve
    SPAposition& pos,       // returned position
    SPAvector* const*       // returned pointer to
        = NULL,             // derivatives
    int nd                  // number of derivatives
        = 0,                // to be evaluated
    int index               // –ve to evaluate the
        = 0                 // left–hand derivative
                            // at a knot, +ve to
                            // evaluate the
                            // right–hand derivative,
                            // 0 for ”don’t care”.
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”baseutil/vector/position.hxx”
#include ”baseutil/vector/vector.hxx”
#include ”kernel/spline/bs3_crv/bs3curve.hxx”
#include ”kernel/spline/bs3_crv/sp3crtn.hxx”
```

Description: General evaluator, giving an arbitrary number of derivatives (up to a maximum returned by accurate_derivs), and selection of the handedness of derivatives at discontinuities. This routine returns the number of derivatives actually evaluated. Any derivatives requested but beyond the maximum are set to 0.0.

deriv must point to an array of locations into which the calculated derivatives are placed. It must contain at least nd pointers, but any pointer may be NULL to indicate that that derivative is not required.

nd specifies the number of derivatives that should be calculated.

index is negative to evaluate the left-hand derivative at a knot and positive to evaluate the right-hand derivative at a knot, or 0 for don’t care about discontinuities.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_crv/sp3crtn.hxx

Effect: Read–only

bs3_curve_extrema

Function: Spline Interface, Construction Geometry

Action: Determines the extreme points (maxima and minima) of a parametric curve with respect to a given direction in object space.

Prototype:
```
curve_extremum* bs3_curve_extrema (
    bs3_curve cur,              // given curve
    SPAunit_vector const& dir   // given direction
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/unitvec.hxx"
#include "kernel/kerngeom/curve/curdef.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/bs3_crv/sp3crtn.hxx"
```

Description: The return value is a linked list of curve_extremum objects, each of which contains the parameter value of an extremum, together with a classification of whether it is a maximum or a minimum. The list is returned in increasing order of parameter.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_crv/sp3crtn.hxx

Effect: Read–only

bs3_curve_facet

Function: Spline Interface, Construction Geometry

Action: Gets an array of points and parameter values that form the linear approximation to a curve within a specified tolerance.

Prototype:
```
void bs3_curve_facet (
    bs3_curve bs,           // given curve
    double s,               // starting parameter
    double e,               // ending parameter
    double tol,             // tolerance
    int nmax,               // max number of points
    int& npts,              // returned number of
                            // points generated, set
                            // to nmax+1 if nmax
                            // exceeded
    SPAposition pts[],      // returned array of
                            // points on the curve
    double t[]              // returned parameter
                            // values at points
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/position.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/sg_bs3c/sps3crtn.hxx"
```

Description: Two arrays are input to the routine, one for curve positions and one for the parameter values at these positions. The dimension of these arrays is specified by the maximum number of points to return.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3c/sps3crtn.hxx

Effect: Read–only

bs3_curve_fillet_curves

Function: Spline Interface, Construction Geometry

Action: Creates a fillet curve between two curves.

Prototype:
```
bs3_curve bs3_curve_fillet_curves (
    bs3_curve crv1,          // first given curve
    double radius1,          // first given radius
    double& t1,              // guess for start
                             // parameter on
                             // first curve
    bs3_curve crv2,          // second given curve
    double radius2,          // second given radius
    double& t2,              // guess for start
                             // parameter on
                             // second curve
    const SPAunit_vector&    // normal to plane of
        normal               // both curves
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/unitvec.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/sg_bs3c/sps3crtn.hxx"
```

Description: If an error occurs, this routine returns a NULL curve.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3c/sps3crtn.hxx

Effect: Changes model

bs3_curve_from_bs2

Function: Spline Interface, Construction Geometry

Action: Converts a two-dimensional parameter-space curve into a three-dimensional curve.

Prototype:
```
bs3_curve bs3_curve_from_bs2 (
    bs2_curve curin          // given curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/bs3_crv/sp3crtn.hxx"
```

Description: Converts a two-dimensional parameter-space curve into a three-dimensional curve by treating the *u* parameter as the *x*-coordinate, the *v* parameter as the *y*-coordinate, and adding a zero as the *z*-coordinate. If an error occurs, this routine returns a NULL curve.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_crv/sp3crtn.hxx

Effect: Changes model

bs3_curve_from_ctrlpts

Function: Spline Interface, Construction Geometry

Action: Creates a 3D B-spline curve specified as a sequence of control points, weights, and an associated knot vector.

Prototype:
```
bs3_curve bs3_curve_from_ctrlpts (
    int degree,              // degree
    logical rational,        // rational
    logical closed,          // closed
    logical periodic,        // periodic
    int num_ctrlpts,         // number of control
                             // points
    const SPAposition        // control points
        ctrlpts[],           // array
    const double weights[],  // weights array
    double ctrlpt_tol,       // control point
                             // tolerance
    int num_knots,           // number of knots
    const double knots[],    // knots array
    double knot_tol,         // knot tolerance
    const int& dimension     // control point
        =*(int*)NULL_REF     // dimension specified in
                             // ctrlpts array.
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/logical.h"
#include "baseutil/vector/position.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/sg_bs3c/sps3crtn.hxx"
```

Description: The control point tolerance and knot tolerance are used to determine when control points or knots are the same. If an error occurs, a NULL curve is returned.

Allowed values for control point dimension are 1, 2 or 3. Default value is 3 corresponding to x, y and z.

Note *Control dimension does NOT depend on rational or non–rational.*

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3c/sps3crtn.hxx

Effect: Changes model

bs3_curve_hermite_interp

Function: Spline Interface, Construction Geometry

Action: Interpolates a cubic B–spline curve (Hermite interpolation) from points and tangents.

Prototype:
```
bs3_curve bs3_curve_hermite_interp (
    int no_pts,              // number of points to
                             // interpolate
    SPAposition* pts,        // array of points
    SPAvector* tans,         // array of tangents
    double* knots            // array of knots
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/position.hxx"
#include "baseutil/vector/vector.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/sg_bs3c/sps3crtn.hxx"
```

Description: Each array is of size no_pts.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3c/sps3crtn.hxx

Effect: Changes model

bs3_curve_init

Function: Spline Interface, Construction Geometry

Action: Initializes the bs3_curve interface and the underlying curve package.

Prototype:
```
void bs3_curve_init ();
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs3_crv/sp3crtn.hxx"
```

Description: ACIS calls this routine once; it should *not* be called again.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_crv/sp3crtn.hxx

Effect: System routine

bs3_curve_int

Function: Spline Interface, Construction Geometry

Action: Determines the intersection of two curves.

Prototype:
```
curve_curve_int* bs3_curve_int (
    bs3_curve& curv1,        // first given curve
    bs3_curve& curv2         // second given curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernint/intcucu/intcucu.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/sg_bs3c/sps3crtn.hxx"
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3c/sps3crtn.hxx

Effect: Changes model

bs3_curve_intcurve_invert

Function: Spline Interface, Construction Geometry

Action: Inverts a bs3_curve, taking an intcurve into account.

Prototype:
```
double bs3_curve_intcurve_invert (
    const SPAposition& pos,     // point to invert
    const intcurve& this_int,   // associated
                                // intcurve
    const SPAparameter& param_guess// guess parameter
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/param.hxx"
#include "baseutil/vector/position.hxx"
#include "kernel/kerngeom/curve/intdef.hxx"
#include "kernel/spline/sg_bs3c/sps3crtn.hxx"
```

Description: This produces an inversion based on an intcurve and not solely on the bs3_curve and returns the actual parameter value.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3c/sps3crtn.hxx

Effect: Changes model

bs3_curve_interp

Function: Spline Interface, Construction Geometry

Action: Creates a curve that interpolates or fits to the given tolerance the given points, with the given tangent directions at the two end points.

Prototype:
```
bs3_curve bs3_curve_interp (
    int npts,                // number of points to
                             // interpolate
    SPAposition const* pts,  // points to interpolate
                             // or fit
    SPAunit_vector const&    // start direction
        start_dir,           // vector
    SPAunit_vector const&    // end direction
        end_dir,             // vector
    double fitol,            // fit tolerance
    double& actual_tol       // returned actual
        =*(double*)NULL_REF, // tolerance used
    logical periodic         // make periodic if no
        = FALSE              // end conditions and
                             // matching end points
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/position.hxx"
#include "baseutil/vector/unitvec.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/bs3_crv/sp3crtn.hxx"
#include "baseutil/logical.h"
```

Description: If an end direction is a NULL reference or has zero length, then a *natural* boundary condition is used, i.e., the second derivative at that end is set to zero.

If any of the start or end conditions are set and are non–zero, or the last point to interpolate is more than SPAresabs from the first point, the periodic flag is ignored.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_crv/sp3crtn.hxx

Effect: Changes model

bs3_curve_interp_knots

Function: Spline Interface, Construction Geometry

Action: Creates a cubic curve which interpolates or fits to an array of points, with given start and end directions.

Prototype:
```
bs3_curve bs3_curve_interp_knots (
    int npt,                // number of points to
                            // interpolate
    SPAposition const* pos[],// points to interpolate
    double knots[],         // knot vector array
    const SPAvector& sdir,  // start derivative
    const SPAvector& edir   // end derivative
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/position.hxx"
#include "baseutil/vector/vector.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/d3_bs3/spd3rtn.hxx"
```

Description: If a direction is a NULL reference or zero length, then a *natural* boundary condition is used, that is zero second derivative at the appropriate end.

Errors: None

Limitations: The number of knots in the knot vector must be (npt + degree + 1). Since this is a cubic (degree is 3), the number of knots must be npt + 4.

Library: kernel

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

Effect: Changes model

bs3_curve_invert

Function: Spline Interface, Construction Geometry

Action: Determines the parameter value of a near–point to the given point on the curve.

Prototype:
```
double bs3_curve_invert (
    SPAposition const& pos, // given point
    double,                 // given tolerance
    bs3_curve cur,          // given curve
    SPAparameter const&     // guess parameter
        param_guess         // guess parameter
        =*(SPAparameter*)NULL_REF
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/param.hxx"
#include "baseutil/vector/position.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/bs3_crv/sp3crtn.hxx"
```

Description: If a guess for the parameter value is supplied, it is assumed to be a good one, so that a close local minimum of distance may be determined. If there is no guess, it is assumed that there is at most one point of minimum distance within the specified tolerance. If the point is near one end of the curve, then there may be no algebraic minimum of the distance function, but the end point itself may be the near point.

Errors: Returns an error if point is not within given tolerance of curve.

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_crv/sp3crtn.hxx

Effect: Read–only

bs3_curve_join

Function: Spline Interface, Construction Geometry

Action: Creates a new curve by appending the second curve to the end of the first.

Prototype:
```
bs3_curve bs3_curve_join (
    bs3_curve first_part,   // first curve
    bs3_curve last_part     // second curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/bs3_crv/sp3crtn.hxx"
```

Description: Appends the second curve to the end of the first, constructing a new combined curve (though destroying the originals). If two curves are input, the first curve is modified to be the combined curve and returned; the second curve is deleted. If one curve is NULL, the other curve is returned as the combined curve.

Errors: None

Limitations: This routine does not check for compatibility. It is commonly used after bs3_curve_split(), where the curves were originally the same curve. Compatible means that the splines have the same degree and rationality, and the end of the first is coincident with the start of the second. Note that this is not what is provided by bs3_curve_compat(). That routine results in two curves that are compatible side–by–side, e.g. that could be used as two curves of the same surface, in the same direction.

Library: kernel

Filename: kern/kernel/spline/bs3_crv/sp3crtn.hxx

Effect: Changes model

bs3_curve_knots

Function: Spline Interface, Construction Geometry

Action: Gets the number of knots and knot values for a 3D B–spline curve.

Prototype:
```
void bs3_curve_knots (
    bs3_curve bs,           // given curve
    int& num_knots,         // number of knots
    double*& knots          // knots
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_crv/bs3curve.hxx”
#include ”kernel/spline/sg_bs3c/sps3crtn.hxx”
```

Description: This function creates arrays of knot points. It is up to the application to delete these arrays.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3c/sps3crtn.hxx

Effect: Read–only

bs3_curve_knottol

Function: Spline Interface, Construction Geometry

Action: Gets the parametric criterion used to determine whether a given parameter is a knot.

Prototype:
```
double bs3_curve_knottol ();
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_crv/sp3crtn.hxx”
```

Description: For the purposes of choosing between discontinuous “sided” derivatives.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_crv/sp3crtn.hxx

Effect: Read–only

bs3_curve_knot_mult

Function: Spline Interface, Construction Geometry

Action: Determines the multiplicity of a B–spline curve knot.

Prototype:
```
int bs3_curve_knot_mult (
    bs3_curve input,         // given curve
    double knot_value,       // knot at which
                             // multiplicity is to be
                             // determined
    double knot_tol          // knot tolerance
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs3_crv/bs3curve.hxx”
#include ”kernel/spline/sg_bs3c/sps3crtn.hxx”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3c/sps3crtn.hxx

Effect: Read–only

bs3_curve_length

Function: Spline Interface, Construction Geometry

Action: Determines the arc length of a three–dimensional B–spline curve between given parameter bounds.

Prototype:
```
double bs3_curve_length (
    bs3_curve cur,          // given curve
    SPAinterval const&      // optional
        cur_range           // parametric bounds
        =*(SPAinterval*)NULL_REF,
    logical approx_OK       // TRUE to get
        = FALSE             // approximate length
                            // quickly (length of
                            // control polygon)
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/logical.h"
#include "baseutil/vector/interval.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/bs3_crv/sp3crtn.hxx"
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_crv/sp3crtn.hxx

Effect: Read–only

bs3_curve_length_param

Function: Spline Interface, Construction Geometry

Action: Determines the parameter value of the point at a given arc length from the given parameter value.

Prototype:
```
double bs3_curve_length_param (
    bs3_curve cur,          // given curve
    double start,           // parameter to datum
                            // point
    double length           // arc length, possibly
                            // the negative of the
                            // desired point
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/bs3_crv/sp3crtn.hxx"
```

Description: This routine acts as an inverse to bs3_curve_param_length.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs3_crv/sp3crtn.hxx

Effect: Read–only

bs3_curve_line_tan_2crv

Function: Spline Interface, Construction Geometry

Action: Determines the positions for creating a line tangent to two bs3_curves.

Prototype:
```
logical bs3_curve_line_tan_2crv (
    bs3_curve crv1,          // first given curve
    double t1,               // first curve parameter
                             // guess
    bs3_curve crv2,          // second given curve
    double t2,               // second curve parameter
                             // guess
    const SPAunit_vector&    // returned normal to
        normal,              // plane of curve
    SPAposition& pt1,        // returned first end of
                             // tangent line
    SPAposition& pt2         // returned second end of
                             // tangent line
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/logical.h"
#include "baseutil/vector/position.hxx"
#include "baseutil/vector/unitvec.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/sg_bs3c/sps3crtn.hxx"
```

Description: Returns TRUE if the points were computed, or FALSE if no solution was found.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3c/sps3crtn.hxx

Effect: Read–only

bs3_curve_line_tan_pt_crv

Function: Spline Interface, Construction Geometry

Action: Determines all positions on a bs3_curve where a line from a given position is tangent to the curve.

Prototype:
```
logical bs3_curve_line_tan_pt_crv (
    const SPAposition& point,// start point
    bs3_curve crv,           // given curve
    const SPAunit_vector&    // curve normal
        normal,              // curve normal
    int nmax,                // maximum number of
                             // points to find
    int& num_pts,            // returned number of
                             // tangency points found
    SPAposition tanpts[]     // returned tangency
                             // points
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/logical.h"
#include "baseutil/vector/position.hxx"
#include "baseutil/vector/unitvec.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/sg_bs3c/sps3crtn.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/sg_bs3c/sps3crtn.hxx"
```

Description: Returns TRUE if the points were computed; FALSE if no solution was found.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs3c/sps3crtn.hxx

Effect: Read–only