Chapter 17.
Functions bs2_curve Aa thru Zz

Topic: Ignore

bs2_curve_accurate_derivs

Function: Spline Interface, Construction Geometry

Action: Gets the number of derivatives that bs2_curve_evaluate can calculate.

Prototype:
```
int bs2_curve_accurate_derivs (
    bs2_curve               // given curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Read–only

bs2_curve_add_knot

Function: Spline Interface, Construction Geometry

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

Prototype:

```
int bs2_curve_add_knot (
    bs2_curve bs2,           // given curve
    double new_knot_param,   // returned new knot
                             // parameter value
    int mult_req,            // returned new knot's
                             // multiplicity
    double knot_tol,         // returned knot
                             // tolerance
    const SPApar_pos&        // new knot surface
        new_knot_uv          // par_pos
        =*(SPApar_pos*)NULL_REF,
    const SPApar_vec&        // 2-space curve
        new_knot_deriv_below// deriv below new knot
        =*(SPApar_vec*)NULL_REF,
    const SPApar_vec&        // deriv above
        new_knot_deriv_above // new knot
        =*(SPApar_vec*)NULL_REF
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "baseutil/vector/param.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/sg_bs2c/sps2crtn.hxx"
```

Description: Add a knot to a B–spline at a given parameter value. 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. If the SPApar_pos for the new knot is supplied, the new pcurve is made to have its Bezier form and then the new knot is made to lie at this point. If the 2D curve derivative is supplied, the pcurve is forced to agree with this at the new knot. At present, this is only implemented for curves of degree 3 (which is most curves).

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs2c/sps2crtn.hxx

Effect: Changes model

bs2_curve_box

Function: Spline Interface, Construction Geometry

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

Prototype:
```
SPApar_box bs2_curve_box (
    bs2_curve cur,          // given curve
    double fitol            // given tolerance
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″baseutil/vector/param.hxx″
#include ″kernel/spline/bs2_crv/bs2curve.hxx″
#include ″kernel/spline/bs2_crv/bs2curve.hxx″
#include ″kernel/spline/bs2_crv/sp2crtn.hxx″
```

Description: Returns a box (presently a rectangular parallelepiped parallel to parameter-space axes) that completely contains the curve, with an additional allowance of the given tolerance all around. 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/bs2_crv/sp2crtn.hxx

Effect: Read–only

bs2_curve_closed

Function: Spline Interface, Construction Geometry

Action: Determines whether a spline is closed or open.

Prototype:
```
logical bs2_curve_closed (
    bs2_curve cur           // given curve
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″baseutil/logical.h″
#include ″kernel/spline/bs2_crv/bs2curve.hxx″
#include ″kernel/spline/bs2_crv/sp2crtn.hxx″
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Read–only

bs2_curve_connect

Function: Spline Interface, Construction Geometry

Action: Joins two 2D B-splines end to end.

Prototype:
```
bs2_curve bs2_curve_connect (
    bs2_curve crv1,          // first curve
    bs2_curve crv2           // second curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
```

Description: The two input curves will be deleted if successful. As with bs3_curve_connect, this routine takes care of compatibility of the curves, and of cleaning up: the two input splines are gone after the call, and the resulting spline is the return value.

It's dangerous to code:

```
s1 = bs2_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 code:

```
s3 = bs2_curve_connect(s1, s2)
```

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

This is similar to bs3_curve_connect, but not identical because the curve values have different meanings. In particular, it does not compare end points to decide which end to connect to which. Also, the second curve is not reparameterized to be true C1 with the first curve, because that's not appropriate in parameter space. As with bs2_curve_join, the second curve will be translated to match the first curve's end point. This also applies to the parameterizations.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Changes model

bs2_curve_construct

Function: Spline Interface, Construction Geometry

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

Prototype:
```
bs2_curve bs2_curve_construct (
    int nkts,                // number of knots
    const SPApar_pos* pverts,// parameter position
    double* knots,           // knots
    int mult                 // multiple
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/param.hxx"
#include "kernel/spline/bs2_crv/bs2curve.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). Same as bs3_curve_construct, but in two dimensions.

pverts 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 multiplicity three.

The form of the bs2_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

bs2_curve_control_points

Function: Spline Interface, Construction Geometry

Action: Gets the number of control points and an array of control points for a 2D B–spline curve.

Prototype:
```
void bs2_curve_control_points (
    bs2_curve bs,           // input curve
    int& num_ctrlpts,       // number of control
                            // points output
    SPApar_pos*& ctrlpts    // output control point
                            // array
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/param.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/sg_bs2c/sps2crtn.hxx"
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs2c/sps2crtn.hxx

Effect: Read–only

bs2_curve_copy

Function: Spline Interface, Construction Geometry

Action: Creates an exact copy of the curve.

Prototype:
```
bs2_curve bs2_curve_copy (
    bs2_curve cur           // given curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
```

Description: Makes an exact copy of the curve. ACIS calls this routine only when a change is to be made to one copy of the curve duplication further. Ordinary duplication of ACIS parameter-space curves merely creates a new reference to the same underlying bs2_curve.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Changes model

bs2_curve_debug

Function: Spline Interface, Construction Geometry, Debugging

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

Prototype:
```
void bs2_curve_debug (
    bs2_curve cur,          // given curve
    char const* leader,     // leader string
    FILE* fp                // debug output file
        = debug_file_ptr
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
```

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

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Read–only

bs2_curve_delete

Function: Spline Interface, Construction Geometry

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

Prototype:
```
void bs2_curve_delete (
    bs2_curve& cur          // given curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
```

Description: If it does not prevent the operation of the standard C memory allocation mechanism, ACIS makes no assumptions about how the underlying surface package manages its storage space.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: System routine

bs2_curve_deriv

Function: Spline Interface, Construction Geometry

Action: Evaluates the first derivative of the curve with respect to the parameter at the given parameter value.

Prototype:
```
SPApar_vec bs2_curve_deriv (
    double param,           // given parameter value
    bs2_curve cur           // given curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/param.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
```

Description: Normally, this is implemented as a call to bs2_curve_eval.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Read–only

bs2_curve_eval

Function: Spline Interface, Construction Geometry

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

Prototype:
```
void bs2_curve_eval (
    double param,           // given parameter value
    bs2_curve cur,          // given curve
    SPApar_pos& x,          // returned parametric
                            // position
    SPApar_vec& xdot        // returned first
        =*(SPApar_vec*)NULL_REF,// derivative
    SPApar_vec& xdotdot     // returned second
        =*(SPApar_vec*)NULL_REF// derivative
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/param.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
```

Description: When that value is not returned and need not be evaluated, either of the two return values may be NULL references.

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: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Read–only

bs2_curve_evaluate

Function: Spline Interface, Construction Geometry

Action: Evaluates an arbitrary number of derivatives and selects the handedness of derivatives at discontinuities.

Prototype:

```
int bs2_curve_evaluate (
    double param,           // given parameter
    bs2_curve cur,          // given curve
    SPApar_pos& pos,        // returned parametric
                            // position
    SPApar_vec** deriv      // array of at least nd
        = NULL,             // pointers to locations
                            // into which derivatives
                            // are to be placed
    int nd                  // number of derivatives
        = 0,                // to be evaluated
    int index               // negative to evaluate
        = 0                 // the left–hand
                            // derivative at a knot,
                            // positive to evaluate
                            // the right–hand
                            // derivative, 0 for
                            // don't care
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "baseutil/vector/param.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
```

Description: Gives an arbitrary number of derivatives (up to a maximum returned by accurate_derivs), and selects the handedness of derivatives at discontinuities. This function returns the number of derivatives actually evaluated. Any derivatives requested but beyond the maximum are set to 0.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Read–only

bs2_curve_fit

Function: Spline Interface, Construction Geometry

Action: Fits a bs2_curve to a collection of point and direction data.

Prototype:
```
void bs2_curve_fit (
    intcurve_data const& int_pts,   // defining data
    bs2_curve& pea_bs               // returned curve
                                    // that fits data
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/spline/agspline/sg_bs2c/intdata.hxx″
#include ″kernel/spline/bs2_crv/bs2curve.hxx″
#include ″kernel/spline/sg_bs2c/sps2crtn.hxx″
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs2c/sps2crtn.hxx

Effect: Changes model

bs2_curve_for_curve_on_surf

Function: Spline Interface, Construction Geometry

Action: Creates a bs2_curve that approximates a segment of a bs3_curve lying on a surface.

Prototype:
```
bs2_curve bs2_curve_for_curve_on_surf (
    bs3_curve ebs,          // object space curve
    double sparam,          // start parameter on
                            // curve
    double eparam,          // end parameter on curve
    bs3_surface srf,        // surface curve lies on
    double fit              // fit tolerance
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/spline/bs2_crv/bs2curve.hxx″
#include ″kernel/spline/bs3_crv/bs3curve.hxx″
#include ″kernel/spline/bs3_srf/bs3surf.hxx″
#include ″kernel/spline/sg_bs2c/sps2crtn.hxx″
```

Description: Given a bs3_surface and a bs3_curve known to lie on the surface (within some tolerance), calculate a bs2_curve that will approximate the curve between a start parameter and an end parameter to within a fit tolerance.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs2c/sps2crtn.hxx

Effect: Changes model

bs2_curve_from_ctrlpts

Function: Spline Interface, Construction Geometry

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

Prototype:
```
bs2_curve bs2_curve_from_ctrlpts (
    int degree,                  // degree
    logical rational,            // rational
    logical closed,              // closed
    logical periodic,            // periodic
    int num_ctrlpts,             // number of control
                                 // points
    const SPAposition* ctrlpts[],// control points, x
                                 // and y components
                                 // only are used
    const double weights[],      // weights
    double,                      // control point
                                 // tolerance
    int num_knots,               // number of knots
    const double knots[],        // knots
    double knot_tol              // knot tolerance
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/logical.h"
#include "baseutil/vector/position.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/sg_bs2c/sps2crtn.hxx"
```

Description: The control point tolerance and knot tolerance are used to determine when points or knots are the same. Only the *x* and *y*-components of the input positions are considered.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs2c/sps2crtn.hxx

Effect: Changes model

bs2_curve_interp

Function: Spline Interface, Construction Geometry

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

Prototype:
```
bs2_curve bs2_curve_interp (
    int npts,                   // number of points
    SPApar_pos const* pts,      // array of points
    SPApar_vec const& start_dir,// start derivative
    SPApar_vec const& end_dir,  // end derivative
    double fitol                // fit tolerance
        = 0,
    double& actual_tol          // returned actual
        =*(double*)NULL_REF     // fit tolerance used
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/param.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
```

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

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Changes model

bs2_curve_join

Function: Spline Interface, Construction Geometry

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

Prototype:
```
bs2_curve bs2_curve_join (
    bs2_curve first_cur,     // first curve
    bs2_curve last_cur       // second curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
```

Description: Appends the second curve to the end of the first, constructing a new combined curve (though destroying the originals). It is assumed that the ends to be joined match in position and direction, and that the parameterization is continuous.

ACIS uses this routine only to rejoin parts of a periodic curve that have been split by bs2_curve_split. They are rejoined in the opposite order so that the resultant curve starts and ends at the split point. A curve made this way is marked as closed by bs2_curve_join, but is not marked periodic.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Changes model

bs2_curve_init

Function: Spline Interface, Construction Geometry

Action: Initializes the bs2_curve interface and the underlying curve package.

Prototype:
```
void bs2_curve_init ();
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
```

Description: ACIS calls this function once internally; it should *not* be called more than once.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: System routine

bs2_curve_knottol

Function: Spline Interface, Construction Geometry

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

Prototype:
```
double bs2_curve_knottol ();
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs2_crv/sp2crtn.hxx”
```

Description: This routine is for the purpose of choosing between discontinuous “sided” derivatives.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Read–only

bs2_curve_knots

Function: Spline Interface, Construction Geometry

Action: Gets the number of knots and the knot vector for a 2D B-spline curve.

Prototype:
```
void bs2_curve_knots (
    bs2_curve bs,           // input curve
    int& num_knots,         // number of knots
    double*& knots          // knot vector
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs2_crv/bs2curve.hxx”
#include ”kernel/spline/sg_bs2c/sps2crtn.hxx”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs2c/sps2crtn.hxx

Effect: Read–only

bs2_curve_make_line

Function: Spline Interface, Construction Geometry

Action: Creates a straight line spline between two points.

Prototype:
```
bs2_curve bs2_curve_make_line (
    SPApar_pos const& start, // start point
    SPApar_pos const& end,   // end point
    double  = 0,             // requested fit
                             // tolerance
    double& actual_fit       // returned actual
         =*(double*)NULL_REF // fit used
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/param.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
```

Description: The resulting parameterization is not defined.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Changes model

bs2_curve_make_rho_conic

Function: Spline Interface, Construction Geometry

Action: Creates a conic curve given three distinct parametric positions and a *rho* value.

Prototype:
```
bs2_curve bs2_curve_make_rho_conic (
    SPApar_pos const& start,// start point
    SPApar_pos const& tan_int,// tangent intersection
                              // point
    SPApar_pos const& end,  // end point
    double rho              // rho value
        = 0.5,
    double  = 0,            // requested fit
                            // tolerance
    double& actual_fit      // returned actual
        =*(double*)NULL_REF  // fit used
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/param.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
```

Description: Given are the start and end points of the segment to be represented, the intersection point of the tangents at the start and end, and the *rho* value. This last determines the position of the mid-parameter point along the line joining the midpoint of the chord and the intersection of the tangents, and is simply the ratio of its distance from the chord midpoint to the total distance. A value of 0.5 gives a parabola, more gives a hyperbola, less an ellipse. The ends are assumed to be distinct.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Changes model

bs2_curve_open

Function: Spline Interface, Construction Geometry

Action: Determines if the spline is open or not.

Prototype:
```
logical bs2_curve_open (
    bs2_curve cur           // given curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/logical.h"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
```

Description: The end points of an open curve do not meet to form a closed loop.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Read–only

bs2_curve_par_trans

Function: Spline Interface, Construction Geometry, Transforms, Modifying Models

Action: Transforms the given bs2_curve in parameter space.

Prototype:
```
void bs2_curve_par_trans (
    bs2_curve bs,           // given curve
    SPApar_transf const& t  // transform
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/param.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
```

Description: A SPApar_transf consists of scaling and translation components. Thus, a bs2_curve can be scaled and translated in parameter space using this function.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Changes model

bs2_curve_period

Function: Spline Interface, Construction Geometry

Action: Gets the primary interval (the parametric period) of definition of a periodic curve.

Prototype:
```
double bs2_curve_period (
    bs2_curve cur           // given curve
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs2_crv/bs2curve.hxx”
#include ”kernel/spline/bs2_crv/sp2crtn.hxx”
```

Description: This function returns exactly zero for any non-periodic curve.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Read–only

bs2_curve_periodic

Function: Spline Interface, Construction Geometry

Action: Determines if the curve is smoothly closed in object space and may be considered an endless loop (periodic).

Prototype:
```
logical bs2_curve_periodic (
    bs2_curve cur           // given curve
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”baseutil/logical.h”
#include ”kernel/spline/bs2_crv/bs2curve.hxx”
#include ”kernel/spline/bs2_crv/sp2crtn.hxx”
```

Description: If the curve is periodic, the curve is defined for all parameter values.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Read–only

bs2_curve_position

Function: Spline Interface, Construction Geometry

Action: Evaluates the curve at the given parameter value.

Prototype:
```
SPApar_pos bs2_curve_position (
    double param,           // given parameter value
    bs2_curve cur           // given curve
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″baseutil/vector/param.hxx″
#include ″kernel/spline/bs2_crv/bs2curve.hxx″
#include ″kernel/spline/bs2_crv/sp2crtn.hxx″
```

Description: Normally, this is implemented as a call to bs2_curve_eval.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Read–only

bs2_curve_range

Function: Spline Interface, Construction Geometry

Action: Gets the primary interval on which the curve is defined.

Prototype:
```
SPAinterval bs2_curve_range (
    bs2_curve cur           // given curve
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″baseutil/vector/interval.hxx″
#include ″kernel/spline/bs2_crv/bs2curve.hxx″
#include ″kernel/spline/bs2_crv/sp2crtn.hxx″
```

Description: In the open and non-periodic closed curves, the interval is also the actual domain.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Read–only

bs2_curve_reparam

Function: Spline Interface, Construction Geometry

Action: Reparameterizes the given curve in place so that its primary interval of definition is from the start to the given end parameters.

Prototype:
```
void bs2_curve_reparam (
    double start,           // start parameter
                            // desired
    double end,             // end parameter desired
    bs2_curve cur           // given curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
```

Description: The start to the given end parameters must be in increasing order.

Each new parameter value will be the appropriate linear function of the old parameter.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Changes model

bs2_curve_restore

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

Action: Restores a curve.

Prototype:
```
bs2_curve bs2_curve_restore ();
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
```

Description: Reads back a representation of a parametric curve written by bs2_curve_save and construct a duplicate of the original curve. Reading uses routines read_int, read_long, read_real, and read_string defined in kernutil/fileio/fileio.hxx.

The overloaded >> operator acts like bs2_curve_restore, but reads from a C++ style stream using stream operators and sets the result into the second argument. For example:

bs2_curve cur;

bs_2_3_spline_restore	Information to restore from SAT

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Changes model

bs2_curve_reverse

Function: Spline Interface, Construction Geometry

Action: Reverses the direction of the given curve, and negates the parameterization.

Prototype:
```
void bs2_curve_reverse (
    bs2_curve cur           // given curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
```

Description: Reverses the direction of the given curve, and negates the parameterization (so that the new primary definition interval is [–*b*, –*a*] if the original was [*a*, *b*]).

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Changes model

bs2_curve_same

Function: Spline Interface, Construction Geometry

Action: Determines whether two curves are the same.

Prototype:
```
logical bs2_curve_same (
    bs2_curve bs1,          // first curve
    bs2_curve bs2,          // second curve
    double tol              // parameter space
        = 0.0               // tolerance for equal
                            // control points
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/logical.h"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
```

Description: This routine checks that the two curves share the same form, and the same knot vectors and control points (within tolerance). The control points may (all) be shifted by any multiple of u_period or v_period. These values refer to the surface in whose parameter space this bs2_curve is embedded, and they default to zero. This routine is not to be used for coincidence testing, but only as a simple filter to discard obvious cases.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Read–only

bs2_curve_save

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

Action: Saves a curve to a file.

Prototype:
```
void bs2_curve_save (
    bs2_curve cur           // given curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
```

Description: Writes a representation of the parametric curve to some external medium, using routines write_int, write_long, write_real, and write_string, defined in ACIS file kernutil/fileio/fileio.hxx. It makes a format the allows reconstruction of the curve from the data by a single sequential pass.

The overloaded << operator acts like bs2_curve_save, but writes to a C++ style stream using stream operators. The output format need not necessarily be the same as for bs2_curve_save, but it is strongly recommended that it be so. For example:

bs2_curve cur;

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: System routine

bs2_curve_set_closed

Function: Spline Interface, Construction Geometry

Action: Marks a bs2_curve as being closed.

Prototype:
```
void bs2_curve_set_closed (
    bs2_curve crv           // given curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/sg_bs2c/sps2crtn.hxx"
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs2c/sps2crtn.hxx

Effect: Read–only

bs2_curve_set_ctrlpt

Function: Spline Interface, Construction Geometry

Action: Sets the position of one control point.

Prototype:
```
void bs2_curve_set_ctrlpt (
    bs2_curve curv,          // bs2_curve to modify
    int index,               // index of target
                             // control point
    SPApar_pos const& pos,   // uv location copied
                             // into control point
    double weight            // weight to which
                             // control point is
                             // assigned, only used if
                             // curve is rational
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/param.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
```

Description: Checks that curv has a control point at the given index. If it does, it copies the *uv* values of pos into control point's data structure. When curv is rational, it also copies the weight value into the control point's data structure.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Changes model

bs2_curve_set_ctrlpts

Function: Spline Interface, Construction Geometry

Action: Sets the positions, and optionally weights, of all control points on a curve.

Prototype:
```
void bs2_curve_set_ctrlpts (
    bs2_curve curv,          // bs2_curve to modify
    const SPApar_pos pos[],  // [uv0,uv1...] to copy
                             // sized:[cpt_count]
    const double weight[]    // weight to which
                             // control point is
                             // assigned, only used if
                             // curve is rational
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”baseutil/vector/param.hxx”
#include ”kernel/spline/bs2_crv/bs2curve.hxx”
#include ”kernel/spline/bs2_crv/sp2crtn.hxx”
```

Description: Iterates through all control points on curv. Copies the values of pos into the corresponding control points. If curv is rational, it also copies the weight values.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Changes model

bs2_curve_set_end_prms

Function: Spline Interface, Construction Geometry

Action: Sets the values of the start and/or end parameters of a spline.

Prototype:
```
logical bs2_curve_set_end_prms (
    double* t0,              // new start parameter
    double* t1,              // new end parameter
    bs2_curve cur            // curve to modify
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”baseutil/logical.h”
#include ”kernel/spline/bs2_crv/bs2curve.hxx”
#include ”kernel/spline/bs2_crv/sp2crtn.hxx”
```

Description: This function adjusts the start and/or end parameter values of a spline. The parameter values are passed in as pointers so that they may be set individually. A NULL pointer indicates no change at that end.

Errors: Knot sequences must be strictly increasing, so the new start value must be less than the second knot value, and the new end value must be greater than the penultimate knot value. If this condition is violated at either end, the function returns FALSE and the curve is left unchanged.

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Changes model

bs2_curve_set_form

Function: Spline Interface, Construction Geometry

Action: Sets a bs2_curve form variable to the same value as a corresponding bs3_curve form variable.

Prototype:
```
void bs2_curve_set_form (
    bs3_curve_form cv2_form,// form variable for
                            // bs3_curve that
                            // corresponds to given
                            // bs2_curve
    bs2_curve cv2           // bs2_curve for which
                            // the form variable is
                            // to be set
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/sg_bs2c/sps2crtn.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
```

Description: This function receives a form variable for the bs3_curve that the bs2_curve corresponds to and sets the bs2_curve form variable to the bs3_curve form variable.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs2c/sps2crtn.hxx

Effect: Changes model

bs2_curve_set_open

Function: Spline Interface, Construction Geometry

Action: Sets a curve's form to be open.

Prototype:
```
void bs2_curve_set_open (
    bs2_curve crv           // given curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/sg_bs2c/sps2crtn.hxx"
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs2c/sps2crtn.hxx

Effect: Changes model

bs2_curve_set_periodic

Function: Spline Interface, Construction Geometry

Action: Marks a bs2_curve as being periodic.

Prototype:
```
void bs2_curve_set_periodic (
    bs2_curve crv           // given curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/sg_bs2c/sps2crtn.hxx"
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs2c/sps2crtn.hxx

Effect: Read–only

bs2_curve_shift

Function: Spline Interface, Construction Geometry

Action: Reparameterizes the given curve in place by adding the given shift value to its parameter values.

Prototype:
```
void bs2_curve_shift (
    double delta,           // parameter shift
                            // desired
    bs2_curve cur           // returned given curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Changes model

bs2_curve_split

Function: Spline Interface, Construction Geometry

Action: Splits a given parametric curve into two sections at a given parameter value.

Prototype:
```
bs2_curve bs2_curve_split (
    bs2_curve& cur,         // given curve
    double param,           // given parameter value
    SPApar_pos const& split_pt// given position
        =*(SPApar_pos*)NULL_REF,
    SPApar_dir const&       // given direction
        =*(SPApar_dir*)NULL_REF
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″baseutil/vector/param.hxx″
#include ″kernel/spline/bs2_crv/bs2curve.hxx″
#include ″kernel/spline/bs2_crv/sp2crtn.hxx″
```

Description: If a parameter-space point is given (which need not be exactly at the point defined by the curve and parameter, but is assumed to be close), the appropriate end point of each resulting spline is shifted to lie exactly on the given position.

No attempt is made to match tangent directions at the cut.

If the B-spline is open, this routine creates a new spline for the initial portion of the curve and returns it as its value. If the B-spline is closed, it takes the portion before the given parameter value and tacks on the end, but marks the spline as open. In this case, the routine returns NULL.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Changes model

bs2_curve_straddles_bs3_surface_knots

Function: Spline Interface, Construction Geometry

Action: Determines if the convex hull of the curve straddles any knots.

Prototype:
```
logical bs2_curve_straddles_bs3_surface_knots (
    bs2_curve bs2_cur,       // given bs2_curve
    bs3_surface bs3_sur,     // given bs3_surface
    logical v_direction      // v direction
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″baseutil/logical.h″
#include ″kernel/spline/bs2_crv/bs2curve.hxx″
#include ″kernel/spline/bs3_srf/bs3surf.hxx″
#include ″kernel/spline/d3_bs3/spd3rtn.hxx″
```

Description: The method used is equivalent to checking if the convex hull of the curve straddles any knots. It determines if the given bs2_curve straddles any of the knot lines of the given bs3_surface, meaning that the curve inherits the continuity degree of the B-spline surface (otherwise its infinitely continuous). It doesn’t matter if occasionally it does straddle some knots when it doesn’t, but the converse error is not acceptable.

Errors: None

Limitations: None

Library: kernel

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

Effect: Read–only

bs2_curve_subset

Function: Spline Interface, Construction Geometry

Action: Creates a curve that is a subset of a given curve.

Prototype:
```
bs2_curve bs2_curve_subset (
    bs2_curve old_bs,           // given curve
    SPAinterval const& new_range,// required bounds
    double = 0,                 // required fit
                                // tolerance
    double& actual_fit          // returned actual
        =*(double*)NULL_REF     // fit tolerance used
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/interval.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
```

Description: Creates a new curve that is a subset of the given curve. The subset is the overlap in parameter space of the given curve and a given interval. A curve periodic in one or both parameter directions is rolled around if need be to cover the required range. It is very unlikely that the tolerance arguments will be needed, but they are included for completeness.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Changes model

bs2_curve_tangent

Function: Spline Interface, Construction Geometry

Action: Evaluates the curve direction at the given parameter value.

Prototype:
```
SPApar_dir bs2_curve_tangent (
    double param,           // given parameter value
    bs2_curve cur           // given curve
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”baseutil/vector/param.hxx”
#include ”kernel/spline/bs2_crv/bs2curve.hxx”
#include ”kernel/spline/bs2_crv/sp2crtn.hxx”
```

Description: Normally, this is implemented as a call to bs2_curve_eval.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Read–only

bs2_curve_to_array

Function: Spline Interface, Construction Geometry

Action: Gets the dimension, degree, control points, weights, and knots for a 2D B-spline curve, and determines if the curve is rational.

Prototype:
```
void bs2_curve_to_array (
    bs2_curve bs,           // given curve
    int& dim,               // returned dimension
    int& deg,               // returned degree
    int& rat,               // returned rational
                            // value
    int& num_ctrlpts,       // returned number of
                            // control points
    SPAposition*& ctrlpts,  // returned control
                            // points, uses
                            // x and y
    double*& weights,       // returned weights
    int& num_knots,         // returned number of
                            // knots
    double*& knots          // returned knots
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/position.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/sg_bs2c/sps2crtn.hxx"
#include "baseutil/logical.h"
```

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

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs2c/sps2crtn.hxx

Effect: Read–only

bs2_curve_to_bs3_curve

Function: Spline Interface, Construction Geometry

Action: Resolves the AG references in the skin code.

Prototype:
```
bs3_curve bs2_curve_to_bs3_curve (
    bs2_curve in_cur        // given curve
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs3_crv/bs3curve.hxx"
#include "kernel/spline/sg_bs2c/sps2crtn.hxx"
```

Description: Not recommended for general use. This routine transfers the ag_spline data from the bs2_curve structure to the bs3_structure.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs2c/sps2crtn.hxx

Effect: Changes model

bs2_curve_trans

Function: Spline Interface, Construction Geometry, Transforms, Modifying Models

Action: Transforms the given curve in place.

Prototype:
```
void bs2_curve_trans (
    bs2_curve,              // given curve
    SPAtransf const&        // transform
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "baseutil/vector/transf.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
```

Description: A SPAtransf consists of a 3x3 matrix with unit determinant, giving an affine transformation, an overall scaling factor and a translation vector. The three logical flags relating to the matrix are:

rotate indicates whether the matrix is anything other than the identity.

reflect indicates whether the determinant is –1.

shear is set if the matrix is not orthogonal.

This function is only provided for completeness. Because both curve and surface parameterizations are transformation-independent, there is nothing for it to do.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Changes model

bs2_curve_u_param_line

Function: Spline Interface, Construction Geometry

Action: Creates a parameter-space curve along a *u* parameter line of a spline surface.

Prototype:
```
bs2_curve bs2_curve_u_param_line (
    bs3_surface surf,       // given surface
    double v                // v parameter of surface
    );
```

```
bs2_curve bs2_curve_u_param_line (
    SPAinterval u_range,     // parametric curve range
    bs2_curve_form form,     // form of required
                             // curve; should
                             // correspond to form of
                             // surface (object–space
                             // meaning rather than
                             // parameter–space)
    double v                 // v parameter of surface
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "baseutil/vector/interval.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
#include "kernel/spline/bs3_srf/bs3surf.hxx"
```

Description: Creates a parameter-space curve along a *u* parameter line (i.e., one with constant *v*-parameter) of a spline surface.

In the first prototype, the surface is used only to obtain the *u*–parameter range. In the second prototype, the range and form are explicitly specified.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Changes model

bs2_curve_v_param_line

Function: Spline Interface, Construction Geometry

Action: Creates a parameter line space curve along a *v* parameter line of a spline surface.

Prototype:

```
bs2_curve bs2_curve_v_param_line (
    bs3_surface surf,        // given surface
    double u                 // u parameter of surface
    );
```

```
bs2_curve bs2_curve_v_param_line (
    SPAinterval v_range,     // parametric curve range
    bs2_curve_form form,     // form of required
                             // curve; should
                             // correspond to form of
                             // surface (object–space
                             // meaning rather than
                             // parameter–space)
    double u                 // u parameter of surface
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "baseutil/vector/interval.hxx"
#include "kernel/spline/bs2_crv/bs2curve.hxx"
#include "kernel/spline/bs2_crv/sp2crtn.hxx"
#include "kernel/spline/bs3_srf/bs3surf.hxx"
```

Description: Creates a parameter line space curve along a *v* parameter line (i.e., one with constant *u* parameter) of a spline surface.

The *v*-parameter of the surface varies for this function, while the *u*-parameter is fixed. The parameterization of the resulting curve is undefined, though it has the same sense as the surface *v*-parameter, but would normally be the same as the surface *v*-parameter. The curve will be open, closed, or periodic according to whether the surface was open, closed, or periodic in the *v*-direction.

The first of the two prototypes shown is the preferred way to create the curve.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/bs2_crv/sp2crtn.hxx

Effect: Changes model

bs2_curve_weights

Function: Spline Interface, Construction Geometry

Action: Gets the number of weights and an array of weights values for a rational, 2D B-spline curve.

Prototype:
```
void bs2_curve_weights (
    bs2_curve bs,           // input curve
    int& num_weights,       // number of weights
    double*& weights        // array of weights
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/spline/bs2_crv/bs2curve.hxx”
#include ”kernel/spline/sg_bs2c/sps2crtn.hxx”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/spline/sg_bs2c/sps2crtn.hxx

Effect: Read–only

bs2_radius_is_zero

Function: Blending

Action: Determines whether or not the spline radius is zero.

Prototype:
```
logical bs2_radius_is_zero (
    bs2_curve bs2,          // spline curve
    double tol              // tolerance
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/sg_husk/vrbln/var_rad.hxx”
#include ”kernel/spline/bs2_crv/bs2curve.hxx”
#include ”baseutil/logical.h”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: kernel

Filename: kern/kernel/sg_husk/vrbln/var_rad.hxx

Effect: Read–only